LAMPS Working Group
Internet-Draft
Updates: 4210, 6712 (if approved)
Intended status: Standards Track
Expires: March 12, 2021

H. Brockhaus
Siemens

September 8, 2020

CMP Updates
draft-ietf-lamps-cmp-updates-04

Abstract

This document contains a set of updates to the base syntax and transport of Certificate Management Protocol (CMP) version 2. This document updates RFC 4210 and RFC 6712.

Specifically, the CMP services updated in this document comprise the enabling of using EnvelopedData instead of EncryptedValue, the definition of extended key usages to identify certificates of CMP endpoints on certification and registration authorities, and adds an HTTP URI discovery mechanism and extend the URI structure.

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 March 12, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
1.1. Convention and Terminology 3
2. Updates to RFC 4210 - Certificate Management Protocol (CMP) . 3
2.1. New Section 1.1. - Changes since RFC 4210 3
2.2. New Section 4.5 - Extended Key Usage 4
2.3. Replace Section 5.1.3.4 - Multiple Protection 6
2.4. Replace Section 5.2.2. - Encrypted Values 7
2.5. Update Section 5.3.4. - Certification Response 9
2.6. Replace Section 5.3.19.9. - Revocation Passphrase 9
2.7. Update Section 5.3.19. - PKI General Message Content . . 10
2.8. Update Section 5.3.22 - Polling Request and Response . . 11
2.9. Update Section 9 - IANA Considerations 12
2.10. Update Appendix B - The Use of Revocation Passphrase . . 13
2.11. Update Appendix C - Request Message Behavioral Clarifications 13
2.12. Update Appendix D.4. - Initial Registration/Certification (Basic Authenticated Scheme) 14
3. Updates to RFC 6712 - HTTP Transfer for the Certificate Management Protocol (CMP) 14
3.1. New Section 1.1. - Changes since RFC 6712 14
3.2. New Section 3.6. - HTTP Request-URI 15
3.3. Update Section 6. - IANA Considerations 16
4. IANA Considerations 16
5. Security Considerations 17
6. Acknowledgements 17
7. References 17
7.1. Normative References 17
7.2. Informative References 18
Appendix A. ASN.1 Modules 19
A.1. 1988 ASN.1 Module 19
A.2. 2002 ASN.1 Module 31
Appendix B. History of changes 43
Author's Address 45

1. Introduction

While using CMP [RFC4210] in industrial and IoT environments and developing the Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile] some limitations were identified in the original CMP specification. This document updates

RFC 4210 [RFC4210] and RFC 6712 [RFC6712] to overcome these limitations.

In general, this document aims to improve the crypto agility of CMP to be flexible to react on future advances in cryptography.

This document also introduces new extended key usages to identify CMP endpoints on registration and certification authorities.

1.1. 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 uses 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], and RFC 5280 [RFC5280]. 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.

KGA: Key generation authority, which generates key pairs on behalf of an EE. The KGA could be co-located with an RA or a CA.

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

2. Updates to RFC 4210 - Certificate Management Protocol (CMP)

2.1. New Section 1.1. - Changes since RFC 4210

The following subsection describes feature updates to RFC 4210 [RFC4210]. They are always related to the base specification. Hence references to the original sections in RFC 4210 [RFC4210] are used whenever possible.

Insert this section at the end of the current Section 1.

1.1 Changes since RFC 4210

The following updates are made in [thisRFC]:

- Add new extended key usages for different CMP server types, e.g. registration authority and certification authority, to express the authorization of the entity identified in the certificate containing the respective extended key usage extension to act as the indicated PKI management entity.

- Extend the description of multiple protection to cover additional use cases, e.g., batch processing of messages.

- Offering EnvelopedData as the preferred choice next to EncryptedValue to extend crypto agility in CMP. Note that according to RFC 4211 [RFC4211] section 2.1.9 the use of the EncryptedValue structure has been deprecated in favor of the EnvelopedData structure. RFC 4211 [RFC4211] offers the EncryptedKey structure, a choice of EncryptedValue and EnvelopedData for migration to EnvelopedData. For reasons of completeness and consistency the exchange of EncryptedValue is performed for all usages in RFC 4210 [RFC4210]. This includes the protection of centrally generated private keys, encryption of certificates, and revocation passphrases.

- Extend the usage of polling also to p10cr messages.

< TBD: The specification of algorithm profiles seed to be moved to a separate document. >

2.2. New Section 4.5 - Extended Key Usage

The following subsection describes new extended key usages for different CMP server types specified in RFC 4210 [RFC4210].

Insert this section at the end of the current Section 4.

4.5 Extended Key Usage

The Extended Key Usage (EKU) extension indicates the purposes for which the certified public key may be used. It therefore restricts the use of a certificate to specific applications.

A CA may want to delegate parts of their duties to other PKI management entities. The mechanism to prove this delegation explained in this section offers zero-touch means to check the authorization of such delegation. Such delegation could also be expressed by other means, e.g., explicit configuration.

To offer automatic validation means for the delegation of a role by a CA, the certificates used by PKI management entities for CMP message protection or signed data for central key generation MUST be issued by the delegating CA and MUST contain the respective EKUs. This proves the authorization of this entity by the delegating CA to act as the PKI management entity as described below.

The ASN.1 to define these EKUs is:

```
id-kp OBJECT IDENTIFIER ::=
   { iso(1) identified-organization(3) dod(6) internet(1)
     security(5) mechanisms(5) pkix(7) kp(3) }

id-kp-cmcCA OBJECT IDENTIFIER ::= { id-kp 27 }
id-kp-cmcRA OBJECT IDENTIFIER ::= { id-kp 28 }
id-kp-cmKGA OBJECT IDENTIFIER ::= { id-kp 32 }
```

Note: RFC 6402 section 2.10 [RFC6402] specifies OIDs for a CMC CA and a CMC RA. As the functionality of a CA and RA is not specific to whether use CMC or CMP as certificate management protocol, the same OIDs SHALL be used for a CMP CA and a CMP RA.

< TBD: The Description of the OIDs for id-kp-cmcCA and id-kp-cmcRA needs to be extended to avoid confusion as they currently only refer to CMC. >

The description of the PKI management entity for each of the EKUs is as follows:

CMP CA: CMP Certification Authorities are CMP endpoints on CA equipment as described in section 3.1.1.2. The key used in the context of CMP management operations, especially CMP message protection, need not be the same key that signs the certificates. It is necessary, however, to ensure that the entity acting as CMP CA is authorized to do so. Therefore, the CMP CA MUST do one of the following,

* use the CA private key on the CMP endpoint, or

* explicitly designate this authority to another entity.

For automatic validation of such delegation it MUST be indicated by the id-kp-cmcCA extended key usage. This extended key usage MUST be placed into the certificate used on the CA equipment and the CA that delegates this role MUST issue the CMP CA certificate.

Note: Using a separate key pair for protecting CMP management operations at the CA decreases the number of operations of the private key used to sign certificates.

CMP RA: CMP Registration Authorities are CMP endpoints on RA equipment as described in Section 3.1.1.3. A CMP RA is identified by the id-kp-cmcRA extended key usage. This extended key usage is placed into RA certificates. The CA that delegated this role is identified by the CA that issued the CMP RA certificate.

CMP KGA: CMP Key Generation Authorities are identified by the id-kp-cmKGA extended key usage. Though the CMP KGA knows the private key it generated on behalf of the end entity. This is a very sensitive service and needs specific authorization. This authorization is either with the CA certificate itself, or indicated by placing the id-kp-cmKGA extended key usage into the CMP RA or CMP CA certificate used to authenticate the origin of the private key, and to express the authorization to offer this service.

Note: In device PKIs, especially those issuing IDevID certificates, CA may have very long validity (including the GeneralizedTime value 99991231235959Z to indicate a not well-defined expiration date as specified in IEEE 802.1AR Section 8.5 [IEEE802.1AR] and RFC 5280 Section 4.1.2.5 [RFC5280]). Such validity periods SHOULD NOT be used for protection of CMP messages. Certificates for delegated CMP message protection (CMP CA, CMP RA, CMP KGA) MUST NOT use indefinite expiration date.

2.3. Replace Section 5.1.3.4 - Multiple Protection

Section 5.1.3.4 of RFC 4210 [RFC4210] describes the nested message. This document opens the usage of nested messages also for batch transport of PKI messages between different PKI management entities.

Replace the text of the section with the following text.

In cases where an end entity sends a protected PKI message to an RA, the RA MAY forward that message to a CA, adding its own protection (which MAY be a MAC or a signature, depending on the information and certificates shared between the RA and the CA). There are different use cases for such multi protected messages.

- The RA confirms the validation and authorization of a message and forwards the original message unchanged.

- The RA collects several messages and forwards them in a batch. This can for instance be used to bridge an off-line connection between two PKI management entities. In communication to the CA request messages and in communication from the CA response or announcement messages will be collected in such batch.

- The RA modifies the message(s) in some way (e.g., add or modify particular field values or add new extensions) before forwarding them, then it MAY create its own desired PKIBody. In case the changes made by the RA to PKIMessage breaks the POP, the RA MUST either set the POP RAVerified or include the original PKIMessage from the EE in the generalInfo field of PKIHeader of the nested message (to force the CA to check POP on the original message). The infoType to be used in this situation is {id-it 15} (see Section 5.3.19 for the value of id-it) and the infoValue is PKIMessages (contents MUST be in the same order as the requests in PKIBody). For simplicity reasons, if batching is used in combination with inclusion of the original PKIMessage in the generalInfo field, all messages in the batch MUST be of the same type (e.g., ir).

These use cases are accomplished by nesting the messages sent by the PKI entity within a new PKI message. The structure used is as follows.

```
NestedMessageContent ::= PKIMessages
```

(The use of PKIMessages, a SEQUENCE OF PKIMessage, lets the RA batch the requests of several EEs in a single new message.)

2.4. Replace Section 5.2.2. - Encrypted Values

Section 5.2.2 of RFC 4210 [RFC4210] describes the usage of EncryptedValue to transport encrypted data. This document extends the encryption of data to preferably use EnvelopedData.

Replace the text of the section with the following text.

Where encrypted data (restricted, in this specification, to be either private keys, certificates, or passwords) are sent in PKI messages, the EncryptedKey data structure is used.

```
EncryptedKey ::= CHOICE {
   encryptedValue        EncryptedValue, -- deprecated
   envelopedData     [0] EnvelopedData }
```

See CRMF [RFC4211] for EncryptedKey and EncryptedValue syntax and for EnvelopedData syntax see CMS [RFC5652]. Using the EncryptedKey data

structure, the choice to either use EncryptedValue (for backward compatibility only) or EnvelopedData is offered. The use of the EncryptedValue structure has been deprecated in favor of the EnvelopedData structure. Therefore, it is recommended to use EnvelopedData.

Note: As we reuse the EncryptedKey structure defined in CRMF [RFC4211], the update is backward compatible. Using the new syntax with the untagged default choice EncryptedValue is bitwise compatible with the old syntax.

The EncryptedKey data structure is used in CMP to either transport a private key, certificate or revocation passphrase in encrypted form.

EnvelopedData is used as follows:

- Contains only one recepientInfo structure because the content is encrypted only for one recipient.

- Contains a private key in the AsymmetricKeyPackage structure as defined in RFC 5958 [RFC5958] wrapped in a SignedData structure as specified in CMS section 5 [RFC5652] signed by the Key Generation Authority.

- Contains a certificate or revocation passphrase directly in the encryptedContent field.

Note: To ensure explicit control of the encoding of the private key according to the specific algorithm the new key pair in an asymmetric key package structure as specified in [RFC5958].

The content of the EnvelopedData structure, as specified in CMS section 6 [RFC5652], MUST be encrypted using a newly generated symmetric content-encryption key. This content-encryption key MUST be securely provided to the recipient using one of three key management techniques.

The choice of the key management technique to be used by the sender depends on the credential available for the recipient:

- Recipient's certificate that contains a key usage extension asserting keyAgreement: The content-encryption key will be protected using the key agreement key management technique, as specified in CMS section 6.2.2 [RFC5652].

- Recipient's certificate that contains a key usage extension asserting keyEncipherment: The content-encryption key will be

protected using the key transport key management technique, as specified in CMS section 6.2.1 [RFC5652].

- Jointly shared secret: The content-encryption key will be protected using the password-based key management technique, as specified in CMS section 6.2.4 [RFC5652].

2.5. Update Section 5.3.4. - Certification Response

Section 5.3.4 of RFC 4210 [RFC4210] describes the Certification Response. This document updates the syntax by using the parent structure EncryptedKey instead of EncryptedValue as described in Section 2.1 above.

Replace the ASN.1 syntax of CertifiedKeyPair and CertOrEncCert with the following text.

```
CertifiedKeyPair ::= SEQUENCE {
    certOrEncCert       CertOrEncCert,
    privateKey      [0] EncryptedKey        OPTIONAL,
    -- see [CRMF] for comment on encoding
    publicationInfo [1] PKIPublicationInfo  OPTIONAL
}

CertOrEncCert ::= CHOICE {
    certificate     [0] Certificate,
    encryptedCert   [1] EncryptedKey
}
```

Add the following paragraphs to the end of the section.

The use of EncryptedKey is described in section 5.2.2.

2.6. Replace Section 5.3.19.9. - Revocation Passphrase

Section 5.3.19.9 of RFC 4210 [RFC4210] describes the provisioning of a revocation passphrase for authenticating a later revocation request. This document updates the handling by using the parent structure EncryptedKey instead of EncryptedValue to transport this information as described in Section 2.1 above.

Replace the text of the section with the following text.

This MAY be used by the EE to send a passphrase to a CA/RA for the purpose of authenticating a later revocation request (in the case that the appropriate signing private key is no longer available to authenticate the request). See Appendix B for further details on the use of this mechanism.

GenMsg: {id-it 12}, EncryptedKey
GenRep: {id-it 12}, < absent >

The use of EncryptedKey is described in section 5.2.2.

2.7. Update Section 5.3.19. - PKI General Message Content

The following subsections describes IDs for new examples InfoTypeAndValue to be used in general messages content specified in RFC 4210 [RFC4210].

Section 5.3.19 of RFC 4210 [RFC4210] describes various PKI general messages and the respective OIDs. This document adds three additional subsection to introduce the new IDs id-it-caCerts, id-it-rootCaKeyUpdate, and id-it-certReqTemplate to the support messages as defined in Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile] Section 4.4.

Add these new subsections at the end of this section with the following text.

2.3.19.14 CA Certificates

This MAY be used by the client to get the latest CA intermediate and issuing CA certificates.

GenMsg: {id-it 17}, < absent >
GenRep: {id-it 17}, CaCertsValue | < absent >

5.3.19.15. Root CA Certificates Update

This MAY be used by the client to get an update of an existing root CA Certificate.

GenMsg: {id-it 18}, < absent >
GenRep: {id-it 18}, RootCaKeyUpdateValue | < absent >

Note: In contrast to CAKeyUpdAnnContent, this type offers omitting newWithOld and oldWithNew in the GenRep message, depending on the needs of the EE.

5.3.19.16. Certificate Request Template

This MAY be used by the client to get a template with parameters for a future certificate request operation.

GenMsg: {id-it 19}, < absent >
GenRep: {id-it 19}, CertReqTemplateValue | < absent >

2.8. Update Section 5.3.22 - Polling Request and Response

Section 5.3.22 of RFC 4210 [RFC4210] describes when and how polling messages are used. This document adds the polling mechanism also to outstanding p10cr transactions.

Replace all paragraphs in front of the state machine diagram with the following text.

This pair of messages is intended to handle scenarios in which the client needs to poll the server in order to determine the status of an outstanding ir, cr, p10cr, or kur transaction (i.e., when the "waiting" PKIStatus has been received).

```
PollReqContent ::= SEQUENCE OF SEQUENCE {
    certReqId    INTEGER }

PollRepContent ::= SEQUENCE OF SEQUENCE {
    certReqId    INTEGER,
    checkAfter   INTEGER,  -- time in seconds
    reason       PKIFreeText OPTIONAL }
```

The following clauses describe when polling messages are used, and how they are used. It is assumed that multiple certConf messages can be sent during transactions. There will be one sent in response to each ip, cp, or kup that contains a CertStatus for an issued certificate.

1 In response to an ip, cp, or kup message, an EE will send a certConf for all issued certificates and, following the ack, a pollReq for all pending certificates.

2 In response to a pollReq, a CA/RA will return an ip, cp, or kup if one or more of the pending certificates is ready; otherwise, it will return a pollRep.

3 If the EE receives a pollRep, it will wait for at least as long as the checkAfter value before sending another pollReq.

4 If an ip, cp, or kup is received in response to a pollReq, then it will be treated in the same way as the initial response.

Note: A p10cr message contains exactly one CertificationRequestInfo data structure as specified in PKCS#10 [RFC2986] but no certificate request number. Therefore, the certReqId MUST be set to 0 in all following messages of this transaction.

2.9. Update Section 9 - IANA Considerations

Section 9 of RFC 4210 [RFC4210] contains the IANA Considerations of that document. As this document defines a new and updates two existing Extended Key Usages, the IANA Considerations need to be updated accordingly.

Add the following paragraphs between the first and second paragraph of the section.

Within the SMI-numbers registry "SMI Security for PKIX Extended Key Purpose Identifiers (1.3.6.1.5.5.7.3)" (see https://www.iana.org/assignments/smi-numbers/smi-numbers.xhtml#smi-numbers-1.3.6.1.5.5.7.3) as defined in RFC 7299 [RFC7299] three changes have been performed.

Two existing entries have been updated to also point to this document:

Decimal	Description	References
27	id-kp-cmcCA	[RFC6402][thisRFC]
28	id-kp-cmcRA	[RFC6402][thisRFC]

One new entry has been added:

Decimal	Description	References
32	id-kp-cmKGA	[thisRFC]

Within the SMI-numbers registry "SMI Security for PKIX CMP Information Types (1.3.6.1.5.5.7.4)" (see https://www.iana.org/assignments/smi-numbers/smi-numbers.xhtml#smi-numbers-1.3.6.1.5.5.7.4) as defined in RFC 7299 [RFC7299] three changes have been performed.

Three new entry have been added:

Decimal	Description	References
17	id-it-caCerts	[thisRFC]
18	id-it-rootCaKeyUpdate	[thisRFC]
19	id-it-certReqTemplate	[thisRFC]

2.10. Update Appendix B - The Use of Revocation Passphrase

Appendix B of RFC 4210 [RFC4210] describes the usage of the revocation passphrase. As this document updates RFC 4210 [RFC4210] to utilize the parent structure EncryptedKey instead of EncryptedValue as described in Section 2.1 above, the description is updated accordingly.

Replace the first bullet point of this section with the following text.

- The OID and value specified in Section 5.3.19.9 of RFC 4210 [RFC4210] MAY be sent in a GenMsg message at any time, or MAY be sent in the generalInfo field of the PKIHeader of any PKIMessage at any time. (In particular, the EncryptedKey as described in section 5.2.2 may be sent in the header of the certConf message that confirms acceptance of certificates requested in an initialization request or certificate request message.) This conveys a revocation passphrase chosen by the entity (i.e., for use of EnvelopedData this is in the decrypted bytes of encryptedContent field and for use of EncryptedValue this is in the decrypted bytes of the encValue field) to the relevant CA/RA; furthermore, the transfer is accomplished with appropriate confidentiality characteristics.

Replace the third bullet point of this section with the following text.

- When using EnvelopedData the localKeyId attribute as specified in RFC 2985 [RFC2985] and when using EncryptedValue the valueHint field MAY contain a key identifier (chosen by the entity, along with the passphrase itself) to assist in later retrieval of the correct passphrase (e.g., when the revocation request is constructed by the entity and received by the CA/RA).

2.11. Update Appendix C - Request Message Behavioral Clarifications

Appendix C of RFC 4210 [RFC4210] provides clarifications to the request message behavior. As this document updates RFC 4210 [RFC4210] to utilize the parent structure EncryptedKey instead of EncryptedValue as described in Section 2.1 above, the description is updated accordingly.

Replace the note coming after the ASN.1 syntax of POPOPrivKey of this section with the following text.

```
-- **********
-- *  the type of "thisMessage" is given as BIT STRING in RFC 4211
-- *  [RFC4211]; it should be "EncryptedKey" (in accordance with
-- *  Section 5.2.2 of this specification). Therefore, this document
-- *  makes the behavioral clarification of specifying that the
-- *  contents of "thisMessage" MUST be encoded either as
-- *  "EnvelopedData" or "EncryptedValue" (only for backward
-- *  compatibility) and then wrapped in a BIT STRING.  This allows
-- *  the necessary conveyance and protection of the private key
-- *  while maintaining bits-on-the-wire compatibility with RFC 4211
-- *  [RFC4211].
-- **********
```

2.12. Update Appendix D.4. - Initial Registration/Certification (Basic Authenticated Scheme)

Appendix D.4 of RFC 4210 [RFC4210] provides the initial registration/certification scheme. This scheme shall continue to use EncryptedValue for backward compatibility reasons.

Replace the comment after the privateKey field of crc[1].certifiedKeyPair in the syntax of the Initialization Response message with the following text.

```
-- see Appendix C, Request Message Behavioral Clarifications
-- for backward compatibility reasons, use EncryptedValue
```

3. Updates to RFC 6712 - HTTP Transfer for the Certificate Management Protocol (CMP)

3.1. New Section 1.1. - Changes since RFC 6712

The following subsection describes feature updates to RFC 6712 [RFC6712]. They are always related to the base specification. Hence references to the original sections in RFC 6712 [RFC6712] are used whenever possible.

Insert this section at the end of the current Section 1.

1.1 Changes since RFC 6712

The following updates are made in draft-ietf-lamps-cmp-updates:

o Add an HTTP URI discovery mechanism and extend the URI structure.

3.2. New Section 3.6. - HTTP Request-URI

Section 3.6 of RFC 6712 [RFC6712] specifies the used HTTP URIs. This document adds a discovery mechanism and extends the URIs.

Replace the text of the section with the following text.

Each CMP server on a PKI management entity supporting HTTP or HTTPS transport MUST support the use of the path-prefix of '/.well-known/' as defined in RFC 8515 [RFC8515] and the registered name of 'cmp' to ease interworking in a multi-vendor environment.

The CMP client needs to be configured with sufficient information to form the CMP server URI. This is at least the authority portion of the URI, e.g., 'www.example.com:80', or the full operational path of the PKI management entity. Additional arbitrary label, e.g., 'profileLabel' and 'operationLabel', may be configured as a separate component or as part of the full operational path to provide further information. The 'profileLabel' may support addressing multiple CAs or certificate profiles and the 'operationLabel' may support addressing PKI management operation specific endpoints. 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/operationLabel

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

4 http://www.example.com/.well-known/cmp/profileLabel/operationLabel

The discovery of supported endpoints as defined above will provide the information to the CMP client 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 CMP client will get a list of potential endpoints supported by the PKI management entity.

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

< TBD: It needs to be discussed if the discovery should be performed using GET on "/.well-known/cmp" or GET on "/.well-known" only. >

The following provides an illustrative example for a PKI management entity supporting various PKI management operations for various certificate profiles and CAs.

Detailed message description:

REQ: GET /.well-known/cmp

RES: Content
</cmp/certprofile1/operation1>;ct=pkixcmp
</cmp/certprofile2/operation1>;ct=pkixcmp
</cmp/certprofile3/operation1>;ct=pkixcmp
</cmp/certprofile1/operation2>;ct=pkixcmp
</cmp/certprofile2/operation2>;ct=pkixcmp
</cmp/certprofile3/operation2>;ct=pkixcmp
</cmp/ca1/operation3>;ct=pkixcmp
</cmp/ca2/operation3>;ct=pkixcmp

3.3. Update Section 6. - IANA Considerations

Section 6 of RFC 6712 [RFC6712] contains the IANA Considerations of that document. As this document defines a new well-known URI, the IANA Considerations need to be updated accordingly.

Add the following text between the first and second paragraph of the section.

Within the well-known URI registry (see https://www.iana.org/assignments/well-known-uris/well-known-uris.xhtml#well-known-uris-1) as defined in RFC 8515 [RFC8515] the following change has been performed.

One new name entry has been added:

URI suffix	Change controller
cmp	IETF

4. IANA Considerations

This document contains an update to the IANA Consideration sections to be added to [RFC4210] and [RFC6712].

< TBD: The existing description and information of id-kp-cmcRA and id-kp-cmcCA need to be updated to reflect their extended usage. >

5. Security Considerations

No changes are made to the existing security considerations of RFC 4210 [RFC4210] and RFC 6712 [RFC6712].

6. Acknowledgements

Special thank goes to Jim Schaad for his guidance and the inspiration on structuring and writing this document I got from [RFC6402] that updates CMC. Special thank also goes also to Russ Housley and Tomas Gustavsson for reviewing and providing valuable suggestions on the approvement of this document.

I also like to thank all reviewers of this document for their valuable feedback.

7. References

7.1. Normative References

[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>.

[RFC2985] Nystrom, M. and B. Kaliski, "PKCS #9: Selected Object Classes and Attribute Types Version 2.0", RFC 2985, DOI 10.17487/RFC2985, November 2000, <https://www.rfc-editor.org/info/rfc2985>.

[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>.

[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>.

[RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, DOI 10.17487/RFC5912, June 2010, <https://www.rfc-editor.org/info/rfc5912>.

[RFC5958] Turner, S., "Asymmetric Key Packages", RFC 5958, DOI 10.17487/RFC5958, August 2010, <https://www.rfc-editor.org/info/rfc5958>.

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

[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>.

[RFC7299] Housley, R., "Object Identifier Registry for the PKIX Working Group", RFC 7299, DOI 10.17487/RFC7299, July 2014, <https://www.rfc-editor.org/info/rfc7299>.

[RFC8515] Jethanandani, M. and M. Reina Ortega, "URN Namespace for ETSI Documents", RFC 8515, DOI 10.17487/RFC8515, February 2019, <https://www.rfc-editor.org/info/rfc8515>.

7.2. Informative References

[I-D.ietf-lamps-lightweight-cmp-profile]
Brockhaus, H., Fries, S., and D. Oheimb, "Lightweight CMP Profile", draft-ietf-lamps-lightweight-cmp-profile-02 (work in progress), July 2020.

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

Appendix A. ASN.1 Modules

A.1. 1988 ASN.1 Module

This section contains the updated ASN.1 module for [RFC4210]. This module replaces the module in Appendix F of that document. Although a 2002 ASN.1 module is provided, this remains the normative module as per the policy of the PKIX working group.

```
PKIXCMP {iso(1) identified-organization(3)
      dod(6) internet(1) security(5) mechanisms(5) pkix(7)
      id-mod(0) id-mod-cmp2000(16)}

DEFINITIONS EXPLICIT TAGS ::=

BEGIN

-- EXPORTS ALL --

IMPORTS

    Certificate, CertificateList, Extensions, AlgorithmIdentifier,
    UTF8String, id-kp -- if required; otherwise, comment out
           FROM PKIX1Explicit88 {iso(1) identified-organization(3)
           dod(6) internet(1) security(5) mechanisms(5) pkix(7)
           id-mod(0) id-pkix1-explicit-88(1)}

    GeneralName, KeyIdentifier
           FROM PKIX1Implicit88 {iso(1) identified-organization(3)
           dod(6) internet(1) security(5) mechanisms(5) pkix(7)
           id-mod(0) id-pkix1-implicit-88(2)}

    CertTemplate, PKIPublicationInfo, EncryptedKey, EncryptedValue,
    CertId, CertReqMessages
           FROM PKIXCRMF-2005 {iso(1) identified-organization(3)
           dod(6) internet(1) security(5) mechanisms(5) pkix(7)
           id-mod(0) id-mod-crmf2005(36)}
    -- The import of EncryptedKey is added due to the updates made
    -- in CMP Updates [thisRFC]

    -- see also the behavioral clarifications to CRMF codified in
    -- Appendix C of this specification
    CertificationRequest
           FROM PKCS-10 {iso(1) member-body(2)
                        us(840) rsadsi(113549)
                        pkcs(1) pkcs-10(10) modules(1) pkcs-10(1)}
    -- (specified in RFC 2986 with 1993 ASN.1 syntax and IMPLICIT
    -- tags).  Alternatively, implementers may directly include
```

```
      -- the [PKCS10] syntax in this module

      localKeyId
             FROM PKCS-9 {iso(1) member-body(2) us(840) rsadsi(113549)
                          pkcs(1) pkcs-9(9) modules(0) pkcs-9(1)}
      -- The import of localKeyId is added due to the updates made in
      -- CMP Updates [thisRFC]

      EnvelopedData, SignedData
             FROM CryptographicMessageSyntax2004 { iso(1)
             member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
             smime(16) modules(0) cms-2004(24) }
      -- The import of EnvelopedData and SignedData is added due to
      -- the updates made in CMP Updates [thisRFC]

      ;

   -- the rest of the module contains locally-defined OIDs and
   -- constructs

   CMPCertificate ::= CHOICE {
       x509v3PKCert        Certificate
   }
   -- This syntax, while bits-on-the-wire compatible with the
   -- standard X.509 definition of "Certificate", allows the
   -- possibility of future certificate types (such as X.509
   -- attribute certificates, WAP WTLS certificates, or other kinds
   -- of certificates) within this certificate management protocol,
   -- should a need ever arise to support such generality.  Those
   -- implementations that do not foresee a need to ever support
   -- other certificate types MAY, if they wish, comment out the
   -- above structure and "un-comment" the following one prior to
   -- compiling this ASN.1 module.  (Note that interoperability
   -- with implementations that don't do this will be unaffected by
   -- this change.)

   -- CMPCertificate ::= Certificate

   PKIMessage ::= SEQUENCE {
       header           PKIHeader,
       body             PKIBody,
       protection   [0] PKIProtection OPTIONAL,
       extraCerts   [1] SEQUENCE SIZE (1..MAX) OF CMPCertificate
                        OPTIONAL
   }

   PKIMessages ::= SEQUENCE SIZE (1..MAX) OF PKIMessage
```

```
   PKIHeader ::= SEQUENCE {
       pvno                INTEGER     { cmp1999(1), cmp2000(2) },
       sender              GeneralName,
       -- identifies the sender
       recipient           GeneralName,
       -- identifies the intended recipient
       messageTime     [0] GeneralizedTime         OPTIONAL,
       -- time of production of this message (used when sender
       -- believes that the transport will be "suitable"; i.e.,
       -- that the time will still be meaningful upon receipt)
       protectionAlg   [1] AlgorithmIdentifier     OPTIONAL,
       -- algorithm used for calculation of protection bits
       senderKID       [2] KeyIdentifier           OPTIONAL,
       recipKID        [3] KeyIdentifier           OPTIONAL,
       -- to identify specific keys used for protection
       transactionID   [4] OCTET STRING            OPTIONAL,
       -- identifies the transaction; i.e., this will be the same in
       -- corresponding request, response, certConf, and PKIConf
       -- messages
       senderNonce     [5] OCTET STRING            OPTIONAL,
       recipNonce      [6] OCTET STRING            OPTIONAL,
       -- nonces used to provide replay protection, senderNonce
       -- is inserted by the creator of this message; recipNonce
       -- is a nonce previously inserted in a related message by
       -- the intended recipient of this message
       freeText        [7] PKIFreeText             OPTIONAL,
       -- this may be used to indicate context-specific instructions
       -- (this field is intended for human consumption)
       generalInfo     [8] SEQUENCE SIZE (1..MAX) OF
                               InfoTypeAndValue     OPTIONAL
       -- this may be used to convey context-specific information
       -- (this field not primarily intended for human consumption)
   }

   PKIFreeText ::= SEQUENCE SIZE (1..MAX) OF UTF8String
       -- text encoded as UTF-8 String [RFC3629] (note: each
       -- UTF8String MAY include an [RFC3066] language tag
       -- to indicate the language of the contained text
       -- see [RFC2482] for details)

   PKIBody ::= CHOICE {       -- message-specific body elements
       ir       [0]  CertReqMessages,        --Initialization Request
       ip       [1]  CertRepMessage,         --Initialization Response
       cr       [2]  CertReqMessages,        --Certification Request
       cp       [3]  CertRepMessage,         --Certification Response
       p10cr    [4]  CertificationRequest,   --imported from [PKCS10]
       popdecc  [5]  POPODecKeyChallContent, --pop Challenge
       popdecr  [6]  POPODecKeyRespContent,  --pop Response
```

```
        kur      [7]  CertReqMessages,        --Key Update Request
        kup      [8]  CertRepMessage,         --Key Update Response
        krr      [9]  CertReqMessages,        --Key Recovery Request
        krp      [10] KeyRecRepContent,       --Key Recovery Response
        rr       [11] RevReqContent,          --Revocation Request
        rp       [12] RevRepContent,          --Revocation Response
        ccr      [13] CertReqMessages,        --Cross-Cert. Request
        ccp      [14] CertRepMessage,         --Cross-Cert. Response
        ckuann   [15] CAKeyUpdAnnContent,     --CA Key Update Ann.
        cann     [16] CertAnnContent,         --Certificate Ann.
        rann     [17] RevAnnContent,          --Revocation Ann.
        crlann   [18] CRLAnnContent,          --CRL Announcement
        pkiconf  [19] PKIConfirmContent,      --Confirmation
        nested   [20] NestedMessageContent,   --Nested Message
        genm     [21] GenMsgContent,          --General Message
        genp     [22] GenRepContent,          --General Response
        error    [23] ErrorMsgContent,        --Error Message
        certConf [24] CertConfirmContent,     --Certificate confirm
        pollReq  [25] PollReqContent,         --Polling request
        pollRep  [26] PollRepContent          --Polling response
    }

    PKIProtection ::= BIT STRING

    ProtectedPart ::= SEQUENCE {
        header    PKIHeader,
        body      PKIBody
    }

    id-PasswordBasedMac OBJECT IDENTIFIER ::= {1 2 840 113533 7 66 13}
    PBMParameter ::= SEQUENCE {
        salt                OCTET STRING,
        -- note:  implementations MAY wish to limit acceptable sizes
        -- of this string to values appropriate for their environment
        -- in order to reduce the risk of denial-of-service attacks
        owf                 AlgorithmIdentifier,
        -- AlgId for a One-Way Function (SHA-1 recommended)
        iterationCount      INTEGER,
        -- number of times the OWF is applied
        -- note:  implementations MAY wish to limit acceptable sizes
        -- of this integer to values appropriate for their environment
        -- in order to reduce the risk of denial-of-service attacks
        mac                 AlgorithmIdentifier
        -- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
    }   -- or HMAC [RFC2104, RFC2202])

    id-DHBasedMac OBJECT IDENTIFIER ::= {1 2 840 113533 7 66 30}
    DHBMParameter ::= SEQUENCE {
```

```
        owf                 AlgorithmIdentifier,
        -- AlgId for a One-Way Function (SHA-1 recommended)
        mac                 AlgorithmIdentifier
        -- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
    }   -- or HMAC [RFC2104, RFC2202])


    NestedMessageContent ::= PKIMessages

    PKIStatus ::= INTEGER {
        accepted               (0),
        -- you got exactly what you asked for
        grantedWithMods        (1),
        -- you got something like what you asked for; the
        -- requester is responsible for ascertaining the differences
        rejection              (2),
        -- you don't get it, more information elsewhere in the message
        waiting                (3),
        -- the request body part has not yet been processed; expect to
        -- hear more later (note: proper handling of this status
        -- response MAY use the polling req/rep PKIMessages specified
        -- in Section 5.3.22; alternatively, polling in the underlying
        -- transport layer MAY have some utility in this regard)
        revocationWarning      (4),
        -- this message contains a warning that a revocation is
        -- imminent
        revocationNotification (5),
        -- notification that a revocation has occurred
        keyUpdateWarning       (6)
        -- update already done for the oldCertId specified in
        -- CertReqMsg
    }

    PKIFailureInfo ::= BIT STRING {
    -- since we can fail in more than one way!
    -- More codes may be added in the future if/when required.
        badAlg              (0),
        -- unrecognized or unsupported Algorithm Identifier
        badMessageCheck     (1),
        -- integrity check failed (e.g., signature did not verify)
        badRequest          (2),
        -- transaction not permitted or supported
        badTime             (3),
        -- messageTime was not sufficiently close to the system time,
        -- as defined by local policy
        badCertId           (4),
        -- no certificate could be found matching the provided criteria
        badDataFormat       (5),
```

```
        -- the data submitted has the wrong format
        wrongAuthority      (6),
        -- the authority indicated in the request is different from the
        -- one creating the response token
        incorrectData       (7),
        -- the requester's data is incorrect (for notary services)
        missingTimeStamp    (8),
        -- when the timestamp is missing but should be there
        -- (by policy)
        badPOP              (9),
        -- the proof-of-possession failed
        certRevoked         (10),
           -- the certificate has already been revoked
        certConfirmed       (11),
           -- the certificate has already been confirmed
        wrongIntegrity      (12),
           -- invalid integrity, password based instead of signature or
           -- vice versa
        badRecipientNonce   (13),
           -- invalid recipient nonce, either missing or wrong value
        timeNotAvailable    (14),
           -- the TSA's time source is not available
        unacceptedPolicy    (15),
           -- the requested TSA policy is not supported by the TSA.
        unacceptedExtension (16),
           -- the requested extension is not supported by the TSA.
        addInfoNotAvailable (17),
           -- the additional information requested could not be
           -- understood or is not available
        badSenderNonce      (18),
           -- invalid sender nonce, either missing or wrong size
        badCertTemplate     (19),
           -- invalid cert. template or missing mandatory information
        signerNotTrusted    (20),
           -- signer of the message unknown or not trusted
        transactionIdInUse  (21),
           -- the transaction identifier is already in use
        unsupportedVersion  (22),
           -- the version of the message is not supported
        notAuthorized       (23),
           -- the sender was not authorized to make the preceding
           -- request or perform the preceding action
        systemUnavail       (24),
        -- the request cannot be handled due to system unavailability
        systemFailure       (25),
        -- the request cannot be handled due to system failure
        duplicateCertReq    (26)
        -- certificate cannot be issued because a duplicate
```

```
        -- certificate already exists
    }

    PKIStatusInfo ::= SEQUENCE {
        status        PKIStatus,
        statusString  PKIFreeText     OPTIONAL,
        failInfo      PKIFailureInfo  OPTIONAL
    }

    OOBCert ::= CMPCertificate

    OOBCertHash ::= SEQUENCE {
        hashAlg     [0] AlgorithmIdentifier     OPTIONAL,
        certId      [1] CertId                  OPTIONAL,
        hashVal         BIT STRING
        -- hashVal is calculated over the DER encoding of the
        -- self-signed certificate with the identifier certID.
    }

    POPODecKeyChallContent ::= SEQUENCE OF Challenge
    -- One Challenge per encryption key certification request (in the
    -- same order as these requests appear in CertReqMessages).

    Challenge ::= SEQUENCE {
        owf                 AlgorithmIdentifier  OPTIONAL,
        -- MUST be present in the first Challenge; MAY be omitted in
        -- any subsequent Challenge in POPODecKeyChallContent (if
        -- omitted, then the owf used in the immediately preceding
        -- Challenge is to be used).
        witness             OCTET STRING,
        -- the result of applying the one-way function (owf) to a
        -- randomly-generated INTEGER, A.  [Note that a different
        -- INTEGER MUST be used for each Challenge.]
        challenge           OCTET STRING
        -- the encryption (under the public key for which the cert.
        -- request is being made) of Rand, where Rand is specified as
        --   Rand ::= SEQUENCE {
        --      int      INTEGER,
        --       - the randomly-generated INTEGER A (above)
        --      sender   GeneralName
        --       - the sender's name (as included in PKIHeader)
        --   }
    }

    POPODecKeyRespContent ::= SEQUENCE OF INTEGER
    -- One INTEGER per encryption key certification request (in the
    -- same order as these requests appear in CertReqMessages).  The
    -- retrieved INTEGER A (above) is returned to the sender of the
```

```
   -- corresponding Challenge.

   CertRepMessage ::= SEQUENCE {
       caPubs       [1] SEQUENCE SIZE (1..MAX) OF CMPCertificate
                        OPTIONAL,
       response         SEQUENCE OF CertResponse
   }

   CertResponse ::= SEQUENCE {
       certReqId           INTEGER,
       -- to match this response with corresponding request (a value
       -- of -1 is to be used if certReqId is not specified in the
       -- corresponding request)
       status              PKIStatusInfo,
       certifiedKeyPair    CertifiedKeyPair    OPTIONAL,
       rspInfo             OCTET STRING        OPTIONAL
       -- analogous to the id-regInfo-utf8Pairs string defined
       -- for regInfo in CertReqMsg [CRMF]
   }

   CertifiedKeyPair ::= SEQUENCE {
       certOrEncCert       CertOrEncCert,
       privateKey      [0] EncryptedKey        OPTIONAL,
       -- see [CRMF] for comment on encoding
       -- Changed from Encrypted Value to EncryptedKey as a CHOICE of
       -- EncryptedValue and EnvelopedData due to the changes made in
       -- CMP Updates [thisRFC]
       -- Using the choice EncryptedValue is bit-compatible to the
       -- syntax without this change
       publicationInfo [1] PKIPublicationInfo  OPTIONAL
   }

   CertOrEncCert ::= CHOICE {
       certificate     [0] CMPCertificate,
       encryptedCert   [1] EncryptedKey
       -- Changed from Encrypted Value to EncryptedKey as a CHOICE of
       -- EncryptedValue and EnvelopedData due to the changes made in
       -- CMP Updates [thisRFC]
       -- Using the choice EncryptedValue is bit-compatible to the
       -- syntax without this change
   }

   KeyRecRepContent ::= SEQUENCE {
       status                  PKIStatusInfo,
       newSigCert          [0] CMPCertificate OPTIONAL,
       caCerts             [1] SEQUENCE SIZE (1..MAX) OF
                                       CMPCertificate OPTIONAL,
       keyPairHist         [2] SEQUENCE SIZE (1..MAX) OF
```

```
                                        CertifiedKeyPair OPTIONAL
   }

   RevReqContent ::= SEQUENCE OF RevDetails

   RevDetails ::= SEQUENCE {
       certDetails         CertTemplate,
       -- allows requester to specify as much as they can about
       -- the cert. for which revocation is requested
       -- (e.g., for cases in which serialNumber is not available)
       crlEntryDetails     Extensions       OPTIONAL
       -- requested crlEntryExtensions
   }

   RevRepContent ::= SEQUENCE {
       status       SEQUENCE SIZE (1..MAX) OF PKIStatusInfo,
       -- in same order as was sent in RevReqContent
       revCerts [0] SEQUENCE SIZE (1..MAX) OF CertId
                                        OPTIONAL,
       -- IDs for which revocation was requested
       -- (same order as status)
       crls     [1] SEQUENCE SIZE (1..MAX) OF CertificateList
                                        OPTIONAL
       -- the resulting CRLs (there may be more than one)
   }

   CAKeyUpdAnnContent ::= SEQUENCE {
       oldWithNew   CMPCertificate, -- old pub signed with new priv
       newWithOld   CMPCertificate, -- new pub signed with old priv
       newWithNew   CMPCertificate  -- new pub signed with new priv
   }

   CertAnnContent ::= CMPCertificate

   RevAnnContent ::= SEQUENCE {
       status              PKIStatus,
       certId              CertId,
       willBeRevokedAt     GeneralizedTime,
       badSinceDate        GeneralizedTime,
       crlDetails          Extensions  OPTIONAL
       -- extra CRL details (e.g., crl number, reason, location, etc.)
   }

   CRLAnnContent ::= SEQUENCE OF CertificateList

   CertConfirmContent ::= SEQUENCE OF CertStatus

   CertStatus ::= SEQUENCE {
```

```
      certHash    OCTET STRING,
      -- the hash of the certificate, using the same hash algorithm
      -- as is used to create and verify the certificate signature
      certReqId   INTEGER,
      -- to match this confirmation with the corresponding req/rep
      statusInfo  PKIStatusInfo OPTIONAL
   }

   PKIConfirmContent ::= NULL

   -- Added in CMP Updates [thisRFC]

   RootCaKeyUpdateContent ::= SEQUENCE {
      newWithNew       CMPCertificate
      -- new root CA certificate
      newWithOld   [0] CMPCertificate OPTIONAL,
      -- X.509 certificate containing the new public root CA key
      -- signed with the old private root CA key
      oldWithNew   [1] CMPCertificate OPTIONAL
      -- old root CA key certificate
      }

   -- Added in CMP Updates [thisRFC]

   CertReqTemplateContent ::= SEQUENCE {
      certTemplate           CertTemplate,
      -- prefilled certTemplate structure elements
      rsaKeyLen              INTEGER OPTIONAL
      -- Any reasonable RSA key length, if subjectPublicKeyInfo
      -- of the certTemplate has the OID rsaEncryption.
      }

   InfoTypeAndValue ::= SEQUENCE {
       infoType               OBJECT IDENTIFIER,
       infoValue              ANY DEFINED BY infoType  OPTIONAL
   }
   -- Example InfoTypeAndValue contents include, but are not limited
   -- to, the following (un-comment in this ASN.1 module and use as
   -- appropriate for a given environment):
   --
   --   id-it-caProtEncCert    OBJECT IDENTIFIER ::= {id-it 1}
   --      CAProtEncCertValue      ::= CMPCertificate
   --   id-it-signKeyPairTypes OBJECT IDENTIFIER ::= {id-it 2}
   --      SignKeyPairTypesValue   ::= SEQUENCE OF AlgorithmIdentifier
   --   id-it-encKeyPairTypes  OBJECT IDENTIFIER ::= {id-it 3}
   --      EncKeyPairTypesValue    ::= SEQUENCE OF AlgorithmIdentifier
   --   id-it-preferredSymmAlg OBJECT IDENTIFIER ::= {id-it 4}
   --      PreferredSymmAlgValue   ::= AlgorithmIdentifier
```

```
   --   id-it-caKeyUpdateInfo  OBJECT IDENTIFIER ::= {id-it 5}
   --      CAKeyUpdateInfoValue    ::= CAKeyUpdAnnContent
   --   id-it-currentCRL       OBJECT IDENTIFIER ::= {id-it 6}
   --      CurrentCRLValue         ::= CertificateList
   --   id-it-unsupportedOIDs  OBJECT IDENTIFIER ::= {id-it 7}
   --      UnsupportedOIDsValue    ::= SEQUENCE OF OBJECT IDENTIFIER
   --   id-it-keyPairParamReq  OBJECT IDENTIFIER ::= {id-it 10}
   --      KeyPairParamReqValue    ::= OBJECT IDENTIFIER
   --   id-it-keyPairParamRep  OBJECT IDENTIFIER ::= {id-it 11}
   --      KeyPairParamRepValue    ::= AlgorithmIdentifer
   --   id-it-revPassphrase    OBJECT IDENTIFIER ::= {id-it 12}
   --      RevPassphraseValue      ::= EncryptedKey
   --      -- Changed from Encrypted Value to EncryptedKey as a CHOICE
   --      -- of EncryptedValue and EnvelopedData due to the changes
   --      -- made in CMP Updates [thisRFC]
   --      -- Using the choice EncryptedValue is bit-compatible to the
   --      -- syntax without this change
   --   id-it-implicitConfirm  OBJECT IDENTIFIER ::= {id-it 13}
   --      ImplicitConfirmValue    ::= NULL
   --   id-it-confirmWaitTime  OBJECT IDENTIFIER ::= {id-it 14}
   --      ConfirmWaitTimeValue    ::= GeneralizedTime
   --   id-it-origPKIMessage   OBJECT IDENTIFIER ::= {id-it 15}
   --      OrigPKIMessageValue     ::= PKIMessages
   --   id-it-suppLangTags     OBJECT IDENTIFIER ::= {id-it 16}
   --      SuppLangTagsValue       ::= SEQUENCE OF UTF8String
   --   id-it-caCerts OBJECT IDENTIFIER ::= { id-it 17}
   --      CaCertsValue ::= SEQUENCE OF CMPCertificate
   --      -- id-it-caCerts added in CMP Updates [thisRFC]
   --   id-it-rootCaKeyUpdate OBJECT IDENTIFIER ::= { id-it 18}
   --      RootCaKeyUpdateValue ::= RootCaKeyUpdateContent
   --      -- id-it-rootCaKeyUpdate added in CMP Updates [thisRFC]
   --   id-it-certReqTemplate OBJECT IDENTIFIER ::= { id-it 19}
   --      CertReqTemplateValue ::= CertReqTemplateContent
   --      -- id-it-certReqTemplate added in CMP Updates [thisRFC]
   --
   -- where
   --
   --   id-pkix OBJECT IDENTIFIER ::= {
   --      iso(1) identified-organization(3)
   --      dod(6) internet(1) security(5) mechanisms(5) pkix(7)}
   -- and
   --   id-it   OBJECT IDENTIFIER ::= {id-pkix 4}
   --
   --
   -- This construct MAY also be used to define new PKIX Certificate
   -- Management Protocol request and response messages, or general-
   -- purpose (e.g., announcement) messages for future needs or for
   -- specific environments.
```

```
   GenMsgContent ::= SEQUENCE OF InfoTypeAndValue

   -- May be sent by EE, RA, or CA (depending on message content).
   -- The OPTIONAL infoValue parameter of InfoTypeAndValue will
   -- typically be omitted for some of the examples given above.
   -- The receiver is free to ignore any contained OBJ. IDs that it
   -- does not recognize. If sent from EE to CA, the empty set
   -- indicates that the CA may send
   -- any/all information that it wishes.

   GenRepContent ::= SEQUENCE OF InfoTypeAndValue
   -- Receiver MAY ignore any contained OIDs that it does not
   -- recognize.

   ErrorMsgContent ::= SEQUENCE {
       pKIStatusInfo          PKIStatusInfo,
       errorCode              INTEGER           OPTIONAL,
       -- implementation-specific error codes
       errorDetails           PKIFreeText       OPTIONAL
       -- implementation-specific error details
   }

   PollReqContent ::= SEQUENCE OF SEQUENCE {
       certReqId              INTEGER
   }

   PollRepContent ::= SEQUENCE OF SEQUENCE {
       certReqId              INTEGER,
       checkAfter             INTEGER,  -- time in seconds
       reason                 PKIFreeText OPTIONAL
   }

   --
   -- Extended Key Usage extension for PKI entities used in CMP
   -- operations, added due to the changes made in
   -- CMP Updates [thisRFC]
   -- The EKUs for the CA and RA are reused from CMC as defined in
   -- [RFC6402]
   --

   -- id-kp-cmcCA OBJECT IDENTIFIER ::= { id-kp 27 }
   -- id-kp-cmcRA OBJECT IDENTIFIER ::= { id-kp 28 }
   id-kp-cmKGA OBJECT IDENTIFIER ::= { id-kp 32 }

   END -- of CMP module
```

A.2. 2002 ASN.1 Module

This section contains the updated 2002 ASN.1 module for [RFC5912]. This module replaces the module in Section 9 of that document. The module contains those changes that were done to update to 2002 ASN.1 standard done in [RFC5912] as well as changes made for this document.

< TBD: Dose this document then also updates [RFC5912]? >

< In case the working group sees a need to provide this ASN.1 module in 2015 syntax, please let me know. >

```
PKIXCMP-2009
    { iso(1) identified-organization(3) dod(6) internet(1)
    security(5) mechanisms(5) pkix(7) id-mod(0)
    id-mod-cmp2000-02(50) } DEFINITIONS EXPLICIT TAGS ::=
BEGIN
IMPORTS

AttributeSet{}, Extensions{}, EXTENSION, ATTRIBUTE
FROM PKIX-CommonTypes-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57)}

AlgorithmIdentifier{}, SIGNATURE-ALGORITHM, ALGORITHM,
    DIGEST-ALGORITHM, MAC-ALGORITHM
FROM AlgorithmInformation-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0)
    id-mod-algorithmInformation-02(58)}

Certificate, CertificateList, id-kp
FROM PKIX1Explicit-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-explicit-02(51)}

GeneralName, KeyIdentifier
FROM PKIX1Implicit-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-implicit-02(59)}

CertTemplate, PKIPublicationInfo, EncryptedKey, EncryptedValue,
    CertId,CertReqMessages
FROM PKIXCRMF-2009
    { iso(1) identified-organization(3) dod(6) internet(1)
    security(5) mechanisms(5) pkix(7) id-mod(0)
    id-mod-crmf2005-02(55) }
-- see also the behavioral clarifications to CRMF codified in
```

```
   -- Appendix C of this specification

   CertificationRequest
   FROM PKCS-10
       {iso(1) identified-organization(3) dod(6) internet(1) security(5)
       mechanisms(5) pkix(7) id-mod(0) id-mod-pkcs10-2009(69)}
   -- (specified in RFC 2986 with 1993 ASN.1 syntax and IMPLICIT
   -- tags).  Alternatively, implementers may directly include
   -- the [PKCS10] syntax in this module

   localKeyId
   FROM PKCS-9
       {iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
       modules(0) pkcs-9(1)}
       -- The import of localKeyId is added due to the updates made in
       -- CMP Updates [thisRFC]

   EnvelopedData, SignedData
   FROM CryptographicMessageSyntax-2009
       {iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
       smime(16) modules(0) id-mod-cms-2004-02(41)}
       -- The import of EnvelopedData and SignedData is added due to
       -- the updates made in CMP Updates [thisRFC]
   ;

   -- the rest of the module contains locally defined OIDs and
   -- constructs

   CMPCertificate ::= CHOICE { x509v3PKCert Certificate, ... }
   -- This syntax, while bits-on-the-wire compatible with the
   -- standard X.509 definition of "Certificate", allows the
   -- possibility of future certificate types (such as X.509
   -- attribute certificates, WAP WTLS certificates, or other kinds
   -- of certificates) within this certificate management protocol,
   -- should a need ever arise to support such generality.  Those
   -- implementations that do not foresee a need to ever support
   -- other certificate types MAY, if they wish, comment out the
   -- above structure and "uncomment" the following one prior to
   -- compiling this ASN.1 module.  (Note that interoperability
   -- with implementations that don't do this will be unaffected by
   -- this change.)

   -- CMPCertificate ::= Certificate

   PKIMessage ::= SEQUENCE {
       header           PKIHeader,
       body             PKIBody,
       protection   [0] PKIProtection OPTIONAL,
```

```
        extraCerts   [1] SEQUENCE SIZE (1..MAX) OF CMPCertificate
                      OPTIONAL }

   PKIMessages ::= SEQUENCE SIZE (1..MAX) OF PKIMessage

   PKIHeader ::= SEQUENCE {
       pvno                INTEGER     { cmp1999(1), cmp2000(2) },
       sender              GeneralName,
       -- identifies the sender
       recipient           GeneralName,
       -- identifies the intended recipient
       messageTime     [0] GeneralizedTime         OPTIONAL,
       -- time of production of this message (used when sender
       -- believes that the transport will be "suitable"; i.e.,
       -- that the time will still be meaningful upon receipt)
       protectionAlg   [1] AlgorithmIdentifier{ALGORITHM, {...}}
                           OPTIONAL,
       -- algorithm used for calculation of protection bits
       senderKID       [2] KeyIdentifier           OPTIONAL,
       recipKID        [3] KeyIdentifier           OPTIONAL,
       -- to identify specific keys used for protection
       transactionID   [4] OCTET STRING            OPTIONAL,
       -- identifies the transaction; i.e., this will be the same in
       -- corresponding request, response, certConf, and PKIConf
       -- messages
       senderNonce     [5] OCTET STRING            OPTIONAL,
       recipNonce      [6] OCTET STRING            OPTIONAL,
       -- nonces used to provide replay protection, senderNonce
       -- is inserted by the creator of this message; recipNonce
       -- is a nonce previously inserted in a related message by
       -- the intended recipient of this message
       freeText        [7] PKIFreeText             OPTIONAL,
       -- this may be used to indicate context-specific instructions
       -- (this field is intended for human consumption)
       generalInfo     [8] SEQUENCE SIZE (1..MAX) OF
                           InfoTypeAndValue     OPTIONAL
       -- this may be used to convey context-specific information
       -- (this field not primarily intended for human consumption)
   }

   PKIFreeText ::= SEQUENCE SIZE (1..MAX) OF UTF8String
       -- text encoded as UTF-8 String [RFC3629] (note: each
       -- UTF8String MAY include an [RFC3066] language tag
       -- to indicate the language of the contained text;
       -- see [RFC2482] for details)

   PKIBody ::= CHOICE {       -- message-specific body elements
       ir       [0]  CertReqMessages,        --Initialization Request
```

```
        ip       [1]  CertRepMessage,         --Initialization Response
        cr       [2]  CertReqMessages,        --Certification Request
        cp       [3]  CertRepMessage,         --Certification Response
        p10cr    [4]  CertificationRequest,   --imported from [PKCS10]
        popdecc  [5]  POPODecKeyChallContent, --pop Challenge
        popdecr  [6]  POPODecKeyRespContent,  --pop Response
        kur      [7]  CertReqMessages,        --Key Update Request
        kup      [8]  CertRepMessage,         --Key Update Response
        krr      [9]  CertReqMessages,        --Key Recovery Request
        krp      [10] KeyRecRepContent,       --Key Recovery Response
        rr       [11] RevReqContent,          --Revocation Request
        rp       [12] RevRepContent,          --Revocation Response
        ccr      [13] CertReqMessages,        --Cross-Cert. Request
        ccp      [14] CertRepMessage,         --Cross-Cert. Response
        ckuann   [15] CAKeyUpdAnnContent,     --CA Key Update Ann.
        cann     [16] CertAnnContent,         --Certificate Ann.
        rann     [17] RevAnnContent,          --Revocation Ann.
        crlann   [18] CRLAnnContent,          --CRL Announcement
        pkiconf  [19] PKIConfirmContent,      --Confirmation
        nested   [20] NestedMessageContent,   --Nested Message
        genm     [21] GenMsgContent,          --General Message
        genp     [22] GenRepContent,          --General Response
        error    [23] ErrorMsgContent,        --Error Message
        certConf [24] CertConfirmContent,     --Certificate confirm
        pollReq  [25] PollReqContent,         --Polling request
        pollRep  [26] PollRepContent          --Polling response
    }

    PKIProtection ::= BIT STRING

    ProtectedPart ::= SEQUENCE {
        header    PKIHeader,
        body      PKIBody }

    id-PasswordBasedMac OBJECT IDENTIFIER ::= { iso(1) member-body(2)
        usa(840) nt(113533) nsn(7) algorithms(66) 13 }
    PBMParameter ::= SEQUENCE {
        salt                OCTET STRING,
        -- note:  implementations MAY wish to limit acceptable sizes
        -- of this string to values appropriate for their environment
        -- in order to reduce the risk of denial-of-service attacks
        owf                 AlgorithmIdentifier{DIGEST-ALGORITHM, {...}},
        -- AlgId for a One-Way Function (SHA-1 recommended)
        iterationCount      INTEGER,
        -- number of times the OWF is applied
        -- note:  implementations MAY wish to limit acceptable sizes
        -- of this integer to values appropriate for their environment
        -- in order to reduce the risk of denial-of-service attacks
```

```
        mac                 AlgorithmIdentifier{MAC-ALGORITHM, {...}}
        -- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
        -- or HMAC [RFC2104, RFC2202])
    }

    id-DHBasedMac OBJECT IDENTIFIER ::= { iso(1) member-body(2)
        usa(840) nt(113533) nsn(7) algorithms(66) 30 }
    DHBMParameter ::= SEQUENCE {
        owf                 AlgorithmIdentifier{DIGEST-ALGORITHM, {...}},
        -- AlgId for a One-Way Function (SHA-1 recommended)
        mac                 AlgorithmIdentifier{MAC-ALGORITHM, {...}}
        -- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
        -- or HMAC [RFC2104, RFC2202])
    }

    PKIStatus ::= INTEGER {
        accepted              (0),
        -- you got exactly what you asked for
        grantedWithMods       (1),
        -- you got something like what you asked for; the
        -- requester is responsible for ascertaining the differences
        rejection             (2),
        -- you don't get it, more information elsewhere in the message
        waiting               (3),
        -- the request body part has not yet been processed; expect to
        -- hear more later (note: proper handling of this status
        -- response MAY use the polling req/rep PKIMessages specified
        -- in Section 5.3.22; alternatively, polling in the underlying
        -- transport layer MAY have some utility in this regard)
        revocationWarning     (4),
        -- this message contains a warning that a revocation is
        -- imminent
        revocationNotification (5),
        -- notification that a revocation has occurred
        keyUpdateWarning      (6)
        -- update already done for the oldCertId specified in
        -- CertReqMsg
    }

    PKIFailureInfo ::= BIT STRING {
    -- since we can fail in more than one way!
    -- More codes may be added in the future if/when required.
        badAlg              (0),
        -- unrecognized or unsupported Algorithm Identifier
        badMessageCheck     (1),
        -- integrity check failed (e.g., signature did not verify)
        badRequest          (2),
        -- transaction not permitted or supported
```

```
        badTime             (3),
        -- messageTime was not sufficiently close to the system time,
        -- as defined by local policy
        badCertId           (4),
        -- no certificate could be found matching the provided criteria
        badDataFormat       (5),
        -- the data submitted has the wrong format
        wrongAuthority      (6),
        -- the authority indicated in the request is different from the
        -- one creating the response token
        incorrectData       (7),
        -- the requester's data is incorrect (for notary services)
        missingTimeStamp    (8),
        -- when the timestamp is missing but should be there
        -- (by policy)
        badPOP              (9),
        -- the proof-of-possession failed
        certRevoked         (10),
        -- the certificate has already been revoked
        certConfirmed       (11),
        -- the certificate has already been confirmed
        wrongIntegrity      (12),
        -- invalid integrity, password based instead of signature or
        -- vice versa
        badRecipientNonce   (13),
        -- invalid recipient nonce, either missing or wrong value
        timeNotAvailable    (14),
        -- the TSA's time source is not available
        unacceptedPolicy    (15),
        -- the requested TSA policy is not supported by the TSA
        unacceptedExtension (16),
        -- the requested extension is not supported by the TSA
        addInfoNotAvailable (17),
        -- the additional information requested could not be
        -- understood or is not available
        badSenderNonce      (18),
        -- invalid sender nonce, either missing or wrong size
        badCertTemplate     (19),
        -- invalid cert. template or missing mandatory information
        signerNotTrusted    (20),
        -- signer of the message unknown or not trusted
        transactionIdInUse  (21),
        -- the transaction identifier is already in use
        unsupportedVersion  (22),
        -- the version of the message is not supported
        notAuthorized       (23),
        -- the sender was not authorized to make the preceding
        -- request or perform the preceding action
```

```
        systemUnavail       (24),
        -- the request cannot be handled due to system unavailability
        systemFailure       (25),
        -- the request cannot be handled due to system failure
        duplicateCertReq    (26)
        -- certificate cannot be issued because a duplicate
        -- certificate already exists
    }

    PKIStatusInfo ::= SEQUENCE {
        status        PKIStatus,
        statusString  PKIFreeText     OPTIONAL,
        failInfo      PKIFailureInfo  OPTIONAL }

    OOBCert ::= CMPCertificate

    OOBCertHash ::= SEQUENCE {
        hashAlg     [0] AlgorithmIdentifier{DIGEST-ALGORITHM, {...}}
                            OPTIONAL,
        certId      [1] CertId                  OPTIONAL,
        hashVal         BIT STRING
        -- hashVal is calculated over the DER encoding of the
        -- self-signed certificate with the identifier certID.
    }

    POPODecKeyChallContent ::= SEQUENCE OF Challenge
    -- One Challenge per encryption key certification request (in the
    -- same order as these requests appear in CertReqMessages).

    Challenge ::= SEQUENCE {
        owf                 AlgorithmIdentifier{DIGEST-ALGORITHM, {...}}
                                OPTIONAL,
        -- MUST be present in the first Challenge; MAY be omitted in
        -- any subsequent Challenge in POPODecKeyChallContent (if
        -- omitted, then the owf used in the immediately preceding
        -- Challenge is to be used).
        witness             OCTET STRING,
        -- the result of applying the one-way function (owf) to a
        -- randomly-generated INTEGER, A.  [Note that a different
        -- INTEGER MUST be used for each Challenge.]
        challenge           OCTET STRING
        -- the encryption (under the public key for which the cert.
        -- request is being made) of Rand, where Rand is specified as
        --   Rand ::= SEQUENCE {
        --      int      INTEGER,
        --       - the randomly-generated INTEGER A (above)
        --      sender   GeneralName
        --       - the sender's name (as included in PKIHeader)
```

```
        --   }
    }

    POPODecKeyRespContent ::= SEQUENCE OF INTEGER
    -- One INTEGER per encryption key certification request (in the
    -- same order as these requests appear in CertReqMessages).  The
    -- retrieved INTEGER A (above) is returned to the sender of the
    -- corresponding Challenge.

    CertRepMessage ::= SEQUENCE {
        caPubs       [1] SEQUENCE SIZE (1..MAX) OF CMPCertificate
                      OPTIONAL,
        response         SEQUENCE OF CertResponse }

    CertResponse ::= SEQUENCE {
        certReqId           INTEGER,
        -- to match this response with the corresponding request (a value
        -- of -1 is to be used if certReqId is not specified in the
        -- corresponding request)
        status              PKIStatusInfo,
        certifiedKeyPair    CertifiedKeyPair    OPTIONAL,
        rspInfo             OCTET STRING        OPTIONAL
        -- analogous to the id-regInfo-utf8Pairs string defined
        -- for regInfo in CertReqMsg [RFC4211]
    }

    CertifiedKeyPair ::= SEQUENCE {
        certOrEncCert       CertOrEncCert,
        privateKey      [0] EncryptedKey      OPTIONAL,
        -- see [RFC4211] for comment on encoding
        -- Changed from Encrypted Value to EncryptedKey as a CHOICE of
        -- EncryptedValue and EnvelopedData due to the changes made in
        -- CMP Updates [thisRFC]
        -- Using the choice EncryptedValue is bit-compatible to the
        -- syntax without this change
        publicationInfo [1] PKIPublicationInfo  OPTIONAL }

    CertOrEncCert ::= CHOICE {
        certificate     [0] CMPCertificate,
        encryptedCert   [1] EncryptedKey
        -- Changed from Encrypted Value to EncryptedKey as a CHOICE of
        -- EncryptedValue and EnvelopedData due to the changes made in
        -- CMP Updates [thisRFC]
        -- Using the choice EncryptedValue is bit-compatible to the
        -- syntax without this change
    }

    KeyRecRepContent ::= SEQUENCE {
```

```
        status                  PKIStatusInfo,
        newSigCert          [0] CMPCertificate OPTIONAL,
        caCerts             [1] SEQUENCE SIZE (1..MAX) OF
                                    CMPCertificate OPTIONAL,
        keyPairHist         [2] SEQUENCE SIZE (1..MAX) OF
                                    CertifiedKeyPair OPTIONAL }

   RevReqContent ::= SEQUENCE OF RevDetails

   RevDetails ::= SEQUENCE {
        certDetails         CertTemplate,
        -- allows requester to specify as much as they can about
        -- the cert. for which revocation is requested
        -- (e.g., for cases in which serialNumber is not available)
        crlEntryDetails     Extensions{{...}}    OPTIONAL
        -- requested crlEntryExtensions
   }

   RevRepContent ::= SEQUENCE {
        status       SEQUENCE SIZE (1..MAX) OF PKIStatusInfo,
        -- in same order as was sent in RevReqContent
        revCerts [0] SEQUENCE SIZE (1..MAX) OF CertId OPTIONAL,
        -- IDs for which revocation was requested
        -- (same order as status)
        crls     [1] SEQUENCE SIZE (1..MAX) OF CertificateList OPTIONAL
        -- the resulting CRLs (there may be more than one)
   }

   CAKeyUpdAnnContent ::= SEQUENCE {
        oldWithNew   CMPCertificate, -- old pub signed with new priv
        newWithOld   CMPCertificate, -- new pub signed with old priv
        newWithNew   CMPCertificate  -- new pub signed with new priv
   }

   CertAnnContent ::= CMPCertificate

   RevAnnContent ::= SEQUENCE {
        status              PKIStatus,
        certId              CertId,
        willBeRevokedAt     GeneralizedTime,
        badSinceDate        GeneralizedTime,
        crlDetails          Extensions{{...}}  OPTIONAL
        -- extra CRL details (e.g., crl number, reason, location, etc.)
   }

   CRLAnnContent ::= SEQUENCE OF CertificateList
   PKIConfirmContent ::= NULL
```

```
   NestedMessageContent ::= PKIMessages

   -- Added in CMP Updates [thisRFC]

   RootCaKeyUpdateContent ::= SEQUENCE {
      newWithNew       CMPCertificate
      -- new root CA certificate
      newWithOld   [0] CMPCertificate OPTIONAL,
      -- X.509 certificate containing the new public root CA key
      -- signed with the old private root CA key
      oldWithNew   [1] CMPCertificate OPTIONAL
      -- old root CA key certificate
      }

   -- Added in CMP Updates [thisRFC]

   CertReqTemplateContent ::= SEQUENCE {
      certTemplate           CertTemplate,
      -- prefilled certTemplate structure elements
      rsaKeyLen              INTEGER OPTIONAL
      -- Any reasonable RSA key length, if subjectPublicKeyInfo
      -- of the certTemplate has the OID rsaEncryption.
      }

   INFO-TYPE-AND-VALUE ::= TYPE-IDENTIFIER

   InfoTypeAndValue ::= SEQUENCE {
       infoType    INFO-TYPE-AND-VALUE.
                       &id({SupportedInfoSet}),
       infoValue   INFO-TYPE-AND-VALUE.
                       &Type({SupportedInfoSet}{@infoType}) }

   SupportedInfoSet INFO-TYPE-AND-VALUE ::= { ... }

   -- Example InfoTypeAndValue contents include, but are not limited
   -- to, the following (uncomment in this ASN.1 module and use as
   -- appropriate for a given environment):
   --
   --   id-it-caProtEncCert    OBJECT IDENTIFIER ::= {id-it 1}
   --      CAProtEncCertValue      ::= CMPCertificate
   --   id-it-signKeyPairTypes OBJECT IDENTIFIER ::= {id-it 2}
   --      SignKeyPairTypesValue   ::= SEQUENCE OF
   --                                      AlgorithmIdentifier{{...}}
   --   id-it-encKeyPairTypes  OBJECT IDENTIFIER ::= {id-it 3}
   --      EncKeyPairTypesValue    ::= SEQUENCE OF
   --                                      AlgorithmIdentifier{{...}}
   --   id-it-preferredSymmAlg OBJECT IDENTIFIER ::= {id-it 4}
   --      PreferredSymmAlgValue   ::= AlgorithmIdentifier{{...}}
```

```
   --   id-it-caKeyUpdateInfo  OBJECT IDENTIFIER ::= {id-it 5}
   --      CAKeyUpdateInfoValue    ::= CAKeyUpdAnnContent
   --   id-it-currentCRL       OBJECT IDENTIFIER ::= {id-it 6}
   --      CurrentCRLValue         ::= CertificateList
   --   id-it-unsupportedOIDs  OBJECT IDENTIFIER ::= {id-it 7}
   --      UnsupportedOIDsValue    ::= SEQUENCE OF OBJECT IDENTIFIER
   --   id-it-keyPairParamReq  OBJECT IDENTIFIER ::= {id-it 10}
   --      KeyPairParamReqValue    ::= OBJECT IDENTIFIER
   --   id-it-keyPairParamRep  OBJECT IDENTIFIER ::= {id-it 11}
   --      KeyPairParamRepValue    ::= AlgorithmIdentifer
   --   id-it-revPassphrase    OBJECT IDENTIFIER ::= {id-it 12}
   --      RevPassphraseValue      ::= EncryptedKey
   --      -- Changed from Encrypted Value to EncryptedKey as a CHOICE
   --      -- of EncryptedValue and EnvelopedData due to the changes
   --      -- made in CMP Updates [thisRFC]
   --      -- Using the choice EncryptedValue is bit-compatible to
   --      -- the syntax without this change
   --   id-it-implicitConfirm  OBJECT IDENTIFIER ::= {id-it 13}
   --      ImplicitConfirmValue    ::= NULL
   --   id-it-confirmWaitTime  OBJECT IDENTIFIER ::= {id-it 14}
   --      ConfirmWaitTimeValue    ::= GeneralizedTime
   --   id-it-origPKIMessage   OBJECT IDENTIFIER ::= {id-it 15}
   --      OrigPKIMessageValue     ::= PKIMessages
   --   id-it-suppLangTags     OBJECT IDENTIFIER ::= {id-it 16}
   --      SuppLangTagsValue       ::= SEQUENCE OF UTF8String
   --   id-it-caCerts OBJECT IDENTIFIER ::= { id-it 17}
   --      CaCertsValue ::= SEQUENCE OF CMPCertificate
   --      -- id-it-caCerts added in CMP Updates [thisRFC]
   --   id-it-rootCaKeyUpdate OBJECT IDENTIFIER ::= { id-it 18}
   --      RootCaKeyUpdateValue ::= RootCaKeyUpdateContent
   --      -- id-it-rootCaKeyUpdate added in CMP Updates [thisRFC]
   --   id-it-certReqTemplate OBJECT IDENTIFIER ::= { id-it 19}
   --      CertReqTemplateValue ::= CertReqTemplateContent
   --      -- id-it-certReqTemplate added in CMP Updates [thisRFC]
   --
   -- where
   --
   --   id-pkix OBJECT IDENTIFIER ::= {
   --      iso(1) identified-organization(3)
   --      dod(6) internet(1) security(5) mechanisms(5) pkix(7)}
   -- and
   --   id-it   OBJECT IDENTIFIER ::= {id-pkix 4}
   --
   --
   -- This construct MAY also be used to define new PKIX Certificate
   -- Management Protocol request and response messages, or general-
   -- purpose (e.g., announcement) messages for future needs or for
   -- specific environments.
```

```
   GenMsgContent ::= SEQUENCE OF InfoTypeAndValue

   -- May be sent by EE, RA, or CA (depending on message content).
   -- The OPTIONAL infoValue parameter of InfoTypeAndValue will
   -- typically be omitted for some of the examples given above.
   -- The receiver is free to ignore any contained OBJECT IDs that it
   -- does not recognize.  If sent from EE to CA, the empty set
   -- indicates that the CA may send
   -- any/all information that it wishes.

   GenRepContent ::= SEQUENCE OF InfoTypeAndValue
   -- Receiver MAY ignore any contained OIDs that it does not
   -- recognize.

   ErrorMsgContent ::= SEQUENCE {
       pKIStatusInfo          PKIStatusInfo,
       errorCode              INTEGER           OPTIONAL,
       -- implementation-specific error codes
       errorDetails           PKIFreeText       OPTIONAL
       -- implementation-specific error details
   }

   CertConfirmContent ::= SEQUENCE OF CertStatus

   CertStatus ::= SEQUENCE {
       certHash    OCTET STRING,
       -- the hash of the certificate, using the same hash algorithm
       -- as is used to create and verify the certificate signature
       certReqId   INTEGER,
       -- to match this confirmation with the corresponding req/rep
       statusInfo  PKIStatusInfo OPTIONAL }

   PollReqContent ::= SEQUENCE OF SEQUENCE {
       certReqId              INTEGER }

   PollRepContent ::= SEQUENCE OF SEQUENCE {
       certReqId              INTEGER,
       checkAfter             INTEGER,  -- time in seconds
       reason                 PKIFreeText OPTIONAL }

   --
   -- Extended Key Usage extension for PKI entities used in CMP
   -- operations, added due to the changes made in
   -- CMP Updates [thisRFC]
   -- The EKUs for the CA and RA are reused from CMC as defined in
   -- [RFC6402]
   --
```

```
   -- id-kp-cmcCA OBJECT IDENTIFIER ::= { id-kp 27 }
   -- id-kp-cmcRA OBJECT IDENTIFIER ::= { id-kp 28 }
   id-kp-cmKGA OBJECT IDENTIFIER ::= { id-kp 32 }

   END
```

Appendix B. History of changes

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

From version 03 -> 04:

- Added Section 2.7 to introduce three new id-it IDs for uses in general messages as discussed (see thread "draft-ietf-lamps-cmp-updates add section to introduce id-it-caCerts, id-it-rootCaKeyUpdate, and id-it-certReqTemplate")
- Added the new id-it IDs and the /.well-known/cmp to the IANA Considerations of [RFC4210] in Section 2.9
- Updated the IANA Considerations of [RFC4210] in Section 2.10
- Some changes in wording on Section 3 due to review comments from Martin Peylo

From version 02 -> 03:

- Added a ToDo on aligning with the CMP Algorithms draft that will be set up as decided in IETF 108
- Updated section on Encrypted Values in Section 2.4 to add the AsymmetricKey Package structure to transport a newly generated private key as decided in IETF 108
- Updated the IANA Considerations of [RFC4210] in Section 2.10
- Added the pre-registered OID in Section 2.10 and the ASN.1 module
- Added Section 3 to document the changes to RFC 6712 [RFC6712] regarding URI discovery and using the path-prefix of '/.well-known/' as discussed in IETF 108
- Updated the IANA Considerations section
- Added a complete updated ASN.1 module in 1988 syntax to update Appendix F of [RFC4210] and a complete updated ASN.1 module in 2002 syntax to update Section 9 of [RFC5912]

- Minor changes in wording

From version 01 -> 02:

- Updated section on EKU OIDs in Section 2.2 as decided in IETF 107
- Changed from symmetric key-encryption to password-based key management technique in Section 2.4 as discussed with Russ and Jim on the mailing list
- Defined the attribute containing the key identifier for the revocation passphrase in Section 2.10
- Moved the change history to the Appendix

From version 00 -> 01:

- Minor changes in wording

From draft-brockhaus-lamps-cmp-updates-03 -> draft-ietf-lamps-cmp-updates-00:

- Changes required to reflect WG adoption

From version 02 -> 03:

- Added some clarification in Section 2.1

From version 01 -> 02:

- Added clarification to section on multiple protection
- Added clarification on new EKUs after some exchange with Tomas Gustavsson
- Reused OIDs from RFC 6402 [RFC6402] as suggested by Sean Turner at IETF 106
- Added clarification on the field containing the key identifier for a revocation passphrase
- Minor changes in wording

From version 00 -> 01:

- Added a section describing the new extended key usages

- Completed the section on changes to the specification of encrypted values
- Added a section on clarification to Appendix D.4
- Minor generalization in RFC 4210 [RFC4210] Sections 5.1.3.4 and 5.3.22
- Minor changes in wording

Author's Address

Hendrik Brockhaus
Siemens AG

Email: hendrik.brockhaus@siemens.com