Workgroup: LAMPS Working Group
Updates: 4210 (if approved)
Published: 20 January 2021
Intended Status: Standards Track
Expires: 24 July 2021
Authors: H. Brockhaus, Ed.
Siemens
H. Aschauer
Siemens
M. Ounsworth
Entrust
S. Mister
Entrust

CMP Algorithms

Abstract

This document describes the conventions for using concrete cryptographic algorithms with the Certificate Management Protocol (CMP). CMP is used to enroll and further manage the lifecycle of X.509 certificates.

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 24 July 2021.

Table of Contents

1. Introduction

[RFC Editor: please delete]: !!! The change history was moved to Appendix B !!!

1.1. 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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Message Digest Algorithms

This section provides references to object identifiers and conventions to be employed by CMP implementations that support SHA2 or SHAKE message digest algorithms.

Digest algorithm identifiers are located in the hashAlg field of OOBCertHash, the owf field of Challenge, PBMParameter, and DHBMParameter, and the digestAlgorithms field of SignedData and the digestAlgorithm field of SignerInfo.

Digest values are located in the hashVal field of OOBCertHash, the witness field of Challenge, and the certHash field of CertStatus. In addition, digest values are input to signature algorithms.

2.1. SHA2

The SHA2 algorithm family is defined in FIPS Pub 180-4 [NIST.FIPS.180-4].

The message digest algorithms SHA-224, SHA-256, SHA-384, and SHA-512 are identified by the following OIDs:

```
id-sha224 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16)
   us(840) organization(1) gov(101) csor(3) nistalgorithm(4)
   hashalgs(2) 4 }
id-sha256 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16)
   us(840) organization(1) gov(101) csor(3) nistalgorithm(4)
   hashalgs(2) 1 }
id-sha384 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16)
   us(840) organization(1) gov(101) csor(3) nistalgorithm(4)
   hashalgs(2) 2 }
id-sha512 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16)
   us(840) organization(1) gov(101) csor(3) nistalgorithm(4)
   hashalgs(2) 3 }
```

Specific conventions to be considered are specified in RFC 5754 Section 2 [RFC5754].

2.2. SHAKE

The SHAKE algorithm family is defined in FIPS Pub 202 [NIST.FIPS.202].

The message digest algorithms SHAKE128 and SHAKE256 are identified by the following OIDs:

```
id-shake128 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16)
   us(840) organization(1) gov(101) csor(3) nistAlgorithm(4)
   hashalgs(2) 11 }
id-shake256 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16)
   us(840) organization(1) gov(101) csor(3) nistAlgorithm(4)
   hashalgs(2) 12 }
```

Specific conventions to be considered are specified in RFC 8702 Section 3.1 [RFC8702].

3. Signature Algorithms

This section provides references to object identifiers and conventions to be employed by CMP implementations that support RSA, ECDSA, or EdDSA signature algorithms.

The signature algorithm is referred to as MSG_SIG_ALG in RFC 4210 Appendix D and E [RFC4210] and in the Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile].

Signature algorithm identifiers are located in the protectionAlg field of PKIHeader, the algorithmIdentifier field of POPOSigningKey, signatureAlgorithm field of CertificationRequest, SignKeyPairTypes, and the SignerInfo signatureAlgorithm field of SignedData.

Signature values are located in the protection field of PKIMessage, signature field of POPOSigningKey, signature field of CertificationRequest, and SignerInfo signature field of SignedData.

3.1. RSA

The RSA (RSASSA-PSS and PKCS#1 version 1.5) signature algorithm is defined in RFC 8017 [RFC8017].

The algorithm identifiers for RSASAA-PSS signatures used with SHA2 message digest algorithms is identified by the following OID:

```
id-RSASSA-PSS OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 10 }
```

Specific conventions to be considered are specified in RFC 4056 [RFC4056].

The signature algorithm RSASSA-PSS used with SHAKE message digest algorithms are identified by the following OIDs:

```
id-RSASSA-PSS-SHAKE128  OBJECT IDENTIFIER  ::=  { iso(1)
   identified-organization(3) dod(6) internet(1) security(5)
   mechanisms(5) pkix(7) algorithms(6) 30 }
id-RSASSA-PSS-SHAKE256  OBJECT IDENTIFIER  ::=  { iso(1)
   identified-organization(3) dod(6) internet(1) security(5)
   mechanisms(5) pkix(7) algorithms(6) 31 }
```

Specific conventions to be considered are specified in RFC 8702 Section 3.2.1 [RFC8702].

The signature algorithm PKCS#1 version 1.5 used with SHA2 message digest algorithms is identified by the following OIDs:

```
sha224WithRSAEncryption OBJECT IDENTIFIER ::= { iso(1)
   member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 14 }
sha256WithRSAEncryption  OBJECT IDENTIFIER  ::=  { iso(1)
   member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 11 }
sha384WithRSAEncryption  OBJECT IDENTIFIER  ::=  { iso(1)
   member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 12 }
sha512WithRSAEncryption  OBJECT IDENTIFIER  ::=  { iso(1)
   member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 13 }
```

Specific conventions to be considered are specified in RFC 5754 Section 3.2 [RFC5754].

3.2. ECDSA

The ECDSA signature algorithm is defined in FIPS Pub 186-4 [NIST.FIPS.186-4].

The signature algorithm ECDSA used with SHA2 message digest algorithms is identified by the following OIDs:

```
ecdsa-with-SHA224 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) 1 }
ecdsa-with-SHA256 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) 2 }
ecdsa-with-SHA384 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) 3 }
ecdsa-with-SHA512 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) 4 }
```

As specified in RFC 5480 [RFC5480] the NIST-recommended SECP curves are identified by the following OIDs:

```
secp192r1 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) ansi-X9-62(10045) curves(3) prime(1) 1 }
secp224r1 OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) curve(0) 33 }
secp256r1 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) ansi-X9-62(10045) curves(3) prime(1) 7 }
secp384r1 OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) curve(0) 34 }
secp521r1 OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) curve(0) 35 }
```

Specific conventions to be considered are specified in RFC 5754 Section 3.3 [RFC5754].

The signature algorithm ECDSA used with SHAKE message digest algorithms are identified by the following OIDs:

```
id-ecdsa-with-shake128 OBJECT IDENTIFIER  ::=  { iso(1)
   identified-organization(3) dod(6) internet(1) security(5)
   mechanisms(5) pkix(7) algorithms(6) 32 }
id-ecdsa-with-shake256 OBJECT IDENTIFIER  ::=  { iso(1)
   identified-organization(3) dod(6) internet(1) security(5)
   mechanisms(5) pkix(7) algorithms(6) 33 }
```

Specific conventions to be considered are specified in RFC 8702 Section 3.2.2 [RFC8702].

3.3. EdDSA

The EdDSA signature algorithm is defined in RFC 8032 Section 3.3 [RFC8032] and FIPS Pub 186-5 (Draft) [NIST.FIPS.186-5].

The signature algorithm Ed25519 MUST be used with SHA-512 message digest algorithms is identified by the following OIDs:

```
id-Ed25519 OBJECT IDENTIFIER  ::=  { iso(1)
   identified-organization(3) thawte(101) 112 }
```

The signature algorithm Ed448 MUST be used with SHAKE256 message digest algorithms is identified by the following OIDs:

```
id-Ed448 OBJECT IDENTIFIER  ::=  { iso(1)
   identified-organization(3) thawte(101) 113 }
```

Specific conventions to be considered are specified in RFC 8419 [RFC8419].

4. Key Management Algorithms

CMP utilizes the following general key management techniques: key agreement, key transport, and passwords.

CRMF [RFC4211] and CMP Updates [I-D.ietf-lamps-cmp-updates] promotes the use of CMS [RFC5652] EnvelopedData by deprecating the use of EncryptedValue.

4.1. Key Agreement Algorithms

The key agreement algorithm is referred to as PROT_ENC_ALG in RFC 4210 Appendix D and E [RFC4210] and in the Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile].

Key agreement algorithms are only used in CMP when using CMS [RFC5652] EnvelopedData together with the key agreement key management technique. When a key agreement algorithm is used, a key-encryption algorithm (Section 4.3) is needed next to the content-encryption algorithm (Section 5).

Key agreement algorithm identifiers are located in the EnvelopedData RecipientInfos KeyAgreeRecipientInfo keyEncryptionAlgorithm fields.

Key encryption algorithm identifiers are located in the EnvelopedData RecipientInfos KeyAgreeRecipientInfo keyEncryptionAlgorithm field.

Wrapped content-encryption keys are located in the EnvelopedData RecipientInfos KeyAgreeRecipientInfo RecipientEncryptedKeys encryptedKey field.

4.1.1. Diffie-Hellman

Diffie-Hellman key agreement is defined in RFC 2631 [RFC2631] and SHALL be used in the ephemeral-static as specified in RFC 3370 [RFC3370]. Static-static variants SHALL NOT be used.

The Diffie-Hellman key agreement algorithm is identified by the following OID:

```
id-alg-ESDH OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) alg(3) 5 }
```

Specific conventions to be considered are specified in RFC 3370 Section 4.1 [RFC3370].

4.1.2. ECDH

Elliptic Curve Diffie-Hellman (ECDH) key agreement is defined in RFC 5753 [RFC5753] and SHALL be used in the ephemeral-static variant as specified in RFC 5753 [RFC5753] or the 1-Pass ECMQV variant as specified in RFC 5753 [RFC5753]. Static-static variants SHALL NOT be used.

The ECDH key agreement algorithm used together with NIST-recommended SECP curves are identified by the following OIDs:

```
dhSinglePass-stdDH-sha224kdf-scheme OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) schemes(1) 11(11) 0 }
dhSinglePass-stdDH-sha256kdf-scheme OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) schemes(1) 11(11) 1 }
dhSinglePass-stdDH-sha384kdf-scheme OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) schemes(1) 11(11) 2 }
dhSinglePass-stdDH-sha512kdf-scheme OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) schemes(1) 11(11) 3 }
dhSinglePass-cofactorDH-sha224kdf-scheme OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) schemes(1)
    14(14) 0 }
dhSinglePass-cofactorDH-sha256kdf-scheme OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) schemes(1)
    14(14) 1 }
dhSinglePass-cofactorDH-sha384kdf-scheme OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) schemes(1)
    14(14) 2 }
dhSinglePass-cofactorDH-sha512kdf-scheme OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) schemes(1)
    14(14) 3 }
mqvSinglePass-sha224kdf-scheme OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) schemes(1) 15(15) 0 }
mqvSinglePass-sha256kdf-scheme OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) schemes(1) 15(15) 1 }
mqvSinglePass-sha384kdf-scheme OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) schemes(1) 15(15) 2 }
mqvSinglePass-sha512kdf-scheme OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) schemes(1) 15(15) 3 }
```

As specified in RFC 5480 [RFC5480] the NIST-recommended SECP curves are identified by the following OIDs:

```
secp192r1 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) ansi-X9-62(10045) curves(3) prime(1) 1 }
secp224r1 OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) curve(0) 33 }
secp256r1 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) ansi-X9-62(10045) curves(3) prime(1) 7 }
secp384r1 OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) curve(0) 34 }
secp521r1 OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) certicom(132) curve(0) 35 }
```

Specific conventions to be considered are specified in RFC 5753 [RFC5753].

The ECDH key agreement algorithm used together with curve25519 or curve448 are identified by the following OIDs:

```
id-X25519 OBJECT IDENTIFIER ::= { iso(1)
   identified-organization(3) thawte(101) 110 }
id-X448 OBJECT IDENTIFIER ::= { iso(1)
   identified-organization(3) thawte(101) 111 }
```

Specific conventions to be considered are specified in RFC 8418 [RFC8418].

4.2. Key Transport Algorithms

The key transport algorithm is also referred to as PROT_ENC_ALG in RFC 4210 Appendix D and E [RFC4210] and in the Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile].

Key transport algorithms are only used in CMP when using CMS [RFC5652] EnvelopedData together with the key transport key management technique.

Key transport algorithm identifiers are located in the EnvelopedData RecipientInfos KeyTransRecipientInfo keyEncryptionAlgorithm field.

Key transport encrypted content-encryption keys are located in the EnvelopedData RecipientInfos KeyTransRecipientInfo encryptedKey field.

4.2.1. RSA

The RSA key transport algorithm is the RSA encryption scheme defined in RFC 8017 [RFC8017].

The algorithm identifier for RSA (PKCS #1 v1.5) is

```
rsaEncryption OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 1 }
```

The algorithm identifier for RSAES-OAEP is:

```
id-RSAES-OAEP  OBJECT IDENTIFIER  ::=  { iso(1) member-body(2)
   us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 7 }
```

Further conventions to be considered for PKCS #1 v1.5 are specified in RFC 3370 Section 4.2.1 [RFC3370] and for RSAES-OAEP in RFC 3560 [RFC3560].

4.3. Symmetric Key-Encryption Algorithms

The symmetric key-encryption algorithm is also referred to as PROT_SYM_ALG in RFC 4210 Appendix D and E [RFC4210] and in the Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile].

As symmetric key-encryption key management technique is not used by CMP, the symmetric key-encryption algorithm is only needed when using the key agreement or password-based key management technique with CMS [RFC5652] EnvelopedData.

Key-encryption algorithm identifiers are located in the EnvelopedData RecipientInfos KeyAgreeRecipientInfo keyEncryptionAlgorithm and EnvelopedData RecipientInfos PasswordRecipientInfo keyEncryptionAlgorithm fields.

Wrapped content-encryption keys are located in the EnvelopedData RecipientInfos KeyAgreeRecipientInfo RecipientEncryptedKeys encryptedKey and EnvelopedData RecipientInfos PasswordRecipientInfo encryptedKey fields.

4.3.1. AES Key Wrap

The AES encryption algorithm is defined in FIPS Pub 197 [NIST.FIPS.197] and the key wrapping is defined in RFC 3394 [RFC3394].

AES key encryption has the algorithm identifier:

```
id-aes128-wrap OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
   country(16) us(840) organization(1) gov(101) csor(3)
   nistAlgorithm(4) aes(1) 5 }
id-aes192-wrap OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
   country(16) us(840) organization(1) gov(101) csor(3)
   nistAlgorithm(4) aes(1) 25 }
id-aes256-wrap OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
   country(16) us(840) organization(1) gov(101) csor(3)
   nistAlgorithm(4) aes(1) 45 }
```

The underlying encryption functions for the key wrap and content-encryption algorithms (as specified in Section 5) and the key sizes for the two algorithms MUST be the same (e.g., AES-128 key wrap algorithm with AES-128 content-encryption algorithm), see also RFC 8551 [RFC8551].

Further conventions to be considered for AES key wrap are specified in RFC 3394 Section 2.2 [RFC3394] and RFC 3565 Section 2.3.2 [RFC3565].

4.4. Key Derivation Algorithms

Key derivation algorithms are only used in CMP when using CMS [RFC5652] EnvelopedData together with password-based key management technique.

Key derivation algorithm identifiers are located in the EnvelopedData RecipientInfos PasswordRecipientInfo keyDerivationAlgorithm field.

When using the password-based key management technique with EnvelopedData as specified in CMP Updates together with MAC-based PKIProtection, a different salt MUST be used with the password-based MAC and KDF to ensure usage of different symmetric keys.

4.4.1. Password-based Key Derivation Function 2

The password-based key derivation function 2 (PBKDF2) is defined in RFC 8018 [RFC8018].

Password-based key derivation function 2 has the algorithm identifier:

```
id-PBKDF2 OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
   rsadsi(113549) pkcs(1) pkcs-5(5) 12 }
```

Further conventions to be considered for PBKDF2 are specified in RFC 3370 Section 4.4.1 [RFC3370] and RFC 8018 Section 5.2 [RFC8018].

5. Content Encryption Algorithms

The content encryption algorithm is also referred to as PROT_SYM_ALG in RFC 4210 Appendix D and E [RFC4210] and in the Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile].

Content encryption algorithms are only used in CMP when using CMS [RFC5652] EnvelopedData to transport a signed private key package in case of central key generation or key archiving, a certificate to facilitate implicit prove-of-possession, or a revocation passphrase in encrypted form.

Content encryption algorithm identifiers are located in the EnvelopedData EncryptedContentInfo contentEncryptionAlgorithm field.

Encrypted content is located in the EnvelopedData EncryptedContentInfo encryptedContent field.

5.1. AES-CBC

The AES encryption algorithm is defined in FIPS Pub 197 [NIST.FIPS.197].

AES-CBC content encryption has the algorithm identifier:

```
id-aes128-CBC OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
   country(16) us(840) organization(1) gov(101) csor(3)
   nistAlgorithm(4) aes(1) 2 }
id-aes192-CBC OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
   country(16) us(840) organization(1) gov(101) csor(3)
   nistAlgorithm(4) aes(1)22 }
id-aes256-CBC OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
   country(16) us(840) organization(1) gov(101) csor(3)
   nistAlgorithm(4) aes(1)42 }
```

Specific conventions to be considered for AES-CBC content encryption are specified in RFC 3565 [RFC3565].

6. Message Authentication Code Algorithms

The message authentication code is either used for shared-secret-based CMP message protection or together with the password-based key derivation function (PBKDF2).

The message authentication code algorithm is also referred to as MSG_MAC_ALG in RFC 4210 Appendix D and E [RFC4210] and in the Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile].

6.1. Password-based MAC

Password-based MAC algorithms combine the derivation of a symmetric-key from a password and a symmetric-key-based MAC function as specified in Section 6.2 using this derived key.

Message authentication code algorithm identifiers are located in the protectionAlg field of PKIHeader.

Message authentication code values are located in the PKIProtection field.

6.1.1. PasswordBasedMac

The PasswordBasedMac algorithm is defined in RFC 4210 Section 5.1.3.1 [RFC4210] and Algorithm Requirements Update to the Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF) [I-D.ietf.lamps-crmf-update-algs].

The PasswordBasedMac algorithm is identified by the following OID:

```
id-PasswordBasedMac OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) nt(113533) nsn(7) algorithms(66) 13 }
```

Further conventions to be considered for password-based MAC are specified in RFC 4210 Section 5.1.3.1 [RFC4210] and Algorithm Requirements Update to the Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF) [I-D.ietf.lamps-crmf-update-algs].

6.1.2. PBMAC1

The password-based message authentication code 1 (PBMAC1) is defined in RFC 8018 [RFC8018]. PBMAC1 combines a password-based key derivation function like PBKDF2 (Section 4.4.1) with an underlying message authentication scheme.

PBMAC1 has the following OID:

```
id-PBKDF2 OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
   rsadsi(113549) pkcs(1) pkcs-5(5) 12 }
```

Specific conventions to be considered for PBMAC1 are specified in RFC 8018 Section 7.1 and A.5 [RFC8018].

6.2. Symmetric-key-based MAC

Symmetric-key-based MAC algorithms are used for deriving the symmetric encryption key when using PBKDF2 as described in Section 4.4.1.

Message authentication code algorithm identifiers are located in the protectionAlg field of PKIHeader, the mac field of PBMParameter, the messageAuthScheme field of PBMAC1, and the prf field of PBKDF2-params.

Message authentication code values are located in the PKIProtection field.

6.2.1. SHA2-based HMAC

The HMAC algorithm is defined in RFC 2104 [RFC2104] and FIPS Pub 198-1 [NIST.FIPS.198-1].

The HMAC algorithm used with SHA2 message digest algorithms is identified by the following OIDs:

```
id-hmacWithSHA224 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) rsadsi(113549) digestAlgorithm(2) 8 }
id-hmacWithSHA256 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) rsadsi(113549) digestAlgorithm(2) 9 }
id-hmacWithSHA384 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) rsadsi(113549) digestAlgorithm(2) 10 }
id-hmacWithSHA512 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
   us(840) rsadsi(113549) digestAlgorithm(2) 11 }
```

Specific conventions to be considered for SHA2-based HMAC are specified in RFC 4231 Section 3.1 [RFC4231].

6.2.2. AES-GMAC

The AES-GMAC algorithm is defined in FIPS Pub 197 [NIST.FIPS.197] and FIPS SP 800-38d [NIST.SP.800-38d].

The AES-GMAC algorithm is identified by the following OIDs:

```
id-aes128-GMAC OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
   country(16) us(840) organization(1) gov(101) csor(3)
   nistAlgorithm(4) aes(1) 9 }
id-aes192-GMAC OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
   country(16) us(840) organization(1) gov(101) csor(3)
   nistAlgorithm(4) aes(1) 29 }
id-aes256-GMAC OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
   country(16) us(840) organization(1) gov(101) csor(3)
   nistAlgorithm(4) aes(1) 49 }
```

Specific conventions to be considered for AES-GMAC are specified in [draft-housley-lamps-cms-aes-mac-alg].draft-ietf-lamps-cms-aes-gmac-alg [I-D.ietf.lamps-cms-aes-gmac-alg].

6.2.3. SHAKE-based KMAC

The KMAC algorithm is defined in RFC 2104 [RFC2104] and FIPS SP 800-195 [NIST.SP.800-195].

The HMAC algorithm used with SHA2 message digest algorithms is identified by the following OIDs:

```
id-KmacWithSHAKE128 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
   country(16) us(840) organization(1) gov(101) csor(3)
   nistAlgorithm(4) 2 19 }
id-KmacWithSHAKE256 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
   country(16) us(840) organization(1) gov(101) csor(3)
   nistAlgorithm(4) 2 20 }
```

Specific conventions to be considered for KMAC with SHAKE are specified in RFC 8702 Section 3.4 [RFC8702].

7. IANA Considerations

This document does not request changes to the IANA registry.

8. Security Considerations

RFC 4210 Appendix D.2 [RFC4210] contains a set of algorithms, mandatory to be supported by conforming implementations. Theses algorithms were appropriate at the time CMP was released, but as cryptographic algorithms weaken over time, some of them should not be used anymore. In general, new attacks are emerging due to research cryptoanalysis or increase in computing power. New algorithms were introduced that are more resistant to today's attacks.

This document lists many cryptographic algorithms usable with CMP to offer implementers a more up to date choice. Finally, the algorithms to be supported also heavily depend on the utilized certificates in the target environment.

In the appendix of this document there is also an update to the Appendix D.2 of RFC 4210 [RFC4210] and a set of algorithms to be supported when implementing the Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile].

To keep the list of algorithms to be used with CMP up to date and to enlist secure algorithms resisting known attack scenarios, future algorithms should be added and weakened algorithms should be deprecated.

9. Acknowledgements

Thanks to Russ Housley for supporting this draft with submitting [I-D.ietf.lamps-cms-aes-gmac-alg] and [I-D.ietf.lamps-crmf-update-algs].

May thanks also to all reviewers like John Gray, Mark Ferreira, Yuefei Lu, Tomas Gustavsson, Lijun Liao, David von Oheimb and Steffen Fries for their input and feedback to this document. Apologies to all not mentioned reviewers and supporters.

10. Normative References

[I-D.ietf-lamps-cmp-updates]
Brockhaus, H., "CMP Updates", Work in Progress, Internet-Draft, draft-ietf-lamps-cmp-updates-06, 2 November 2020, <https://tools.ietf.org/html/draft-ietf-lamps-cmp-updates-06>.

[I-D.ietf.lamps-cms-aes-gmac-alg]
Housley, R., "Using the AES-GMAC Algorithm with the Cryptographic Message Syntax (CMS)", Work in Progress, Internet-Draft, draft-ietf-lamps-cms-aes-gmac-alg-00, 2 December 2020, <https://datatracker.ietf.org/doc/draft-ietf-lamps-cms-aes-gmac-alg-00>.

[I-D.ietf.lamps-crmf-update-algs]
Housley, R., "Algorithm Requirements Update to the Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", Work in Progress, Internet-Draft, draft-ietf-lamps-crmf-update-algs-00, 10 December 2020, <http://datatracker.ietf.org/doc/draft-ietf-lamps-crmf-update-algs-00>.

[NIST.FIPS.180-4] Dang, Quynh H., "Secure Hash Standard", NIST NIST FIPS 180-4, DOI 10.6028/NIST.FIPS.180-4, July 2015, <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf>.

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

[NIST.FIPS.186-5] National Institute of Standards and Technology (NIST), "FIPS Pub 186-5 (Draft): Digital Signature Standard (DSS)", October 2019, <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5-draft.pdf>.

[NIST.FIPS.197] National Institute of Standards and Technology (NIST), "Advanced encryption standard (AES)", NIST NIST FIPS 197, DOI 10.6028/NIST.FIPS.197, November 2001, <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf>.

[NIST.FIPS.198-1]
National Institute of Standards and Technology (NIST), "The Keyed-Hash Message Authentication Code (HMAC)", NIST NIST FIPS 198-1, DOI 10.6028/NIST.FIPS.198-1, July 2008, <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.198-1.pdf>.

[NIST.FIPS.202] Dworkin, Morris J., "SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions", NIST NIST FIPS 202, DOI 10.6028/NIST.FIPS.202, July 2015, <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf>.

[NIST.SP.800-195] O'Reilly, Patrick., Rigopoulos, Kristina., Feldman, Larry., and Greg. Witte, "2016 NIST/ITL cybersecurity program: annual report", NIST NIST SP 800-195, DOI 10.6028/NIST.SP.800-195, September 2017, <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-195.pdf>.

[NIST.SP.800-38d] Dworkin, M J., "Recommendation for block cipher modes of operation :GaloisCounter Mode (GCM) and GMAC", NIST NIST SP 800-38d, DOI 10.6028/NIST.SP.800-38d, 2007, <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf>.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/info/rfc2104>.

[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>.

[RFC2631] Rescorla, E., "Diffie-Hellman Key Agreement Method", RFC 2631, DOI 10.17487/RFC2631, June 1999, <https://www.rfc-editor.org/info/rfc2631>.

[RFC3370] Housley, R., "Cryptographic Message Syntax (CMS) Algorithms", RFC 3370, DOI 10.17487/RFC3370, August 2002, <https://www.rfc-editor.org/info/rfc3370>.

[RFC3394] Schaad, J. and R. Housley, "Advanced Encryption Standard (AES) Key Wrap Algorithm", RFC 3394, DOI 10.17487/RFC3394, September 2002, <https://www.rfc-editor.org/info/rfc3394>.

[RFC3560] Housley, R., "Use of the RSAES-OAEP Key Transport Algorithm in Cryptographic Message Syntax (CMS)", RFC

3560, DOI 10.17487/RFC3560, July 2003, <https://www.rfc-editor.org/info/rfc3560>.

[RFC3565] Schaad, J., "Use of the Advanced Encryption Standard (AES) Encryption Algorithm in Cryptographic Message Syntax (CMS)", RFC 3565, DOI 10.17487/RFC3565, July 2003, <https://www.rfc-editor.org/info/rfc3565>.

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

[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>.

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

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

[RFC5753] Turner, S. and D. Brown, "Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax (CMS)", RFC 5753, DOI 10.17487/RFC5753, January 2010, <https://www.rfc-editor.org/info/rfc5753>.

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

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

[RFC8018] Moriarty, K., Ed., Kaliski, B., and A. Rusch, "PKCS #5: Password-Based Cryptography Specification Version 2.1",

RFC 8018, DOI 10.17487/RFC8018, January 2017, <https://www.rfc-editor.org/info/rfc8018>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/info/rfc8032>.

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

[RFC8418] Housley, R., "Use of the Elliptic Curve Diffie-Hellman Key Agreement Algorithm with X25519 and X448 in the Cryptographic Message Syntax (CMS)", RFC 8418, DOI 10.17487/RFC8418, August 2018, <https://www.rfc-editor.org/info/rfc8418>.

[RFC8419] Housley, R., "Use of Edwards-Curve Digital Signature Algorithm (EdDSA) Signatures in the Cryptographic Message Syntax (CMS)", RFC 8419, DOI 10.17487/RFC8419, August 2018, <https://www.rfc-editor.org/info/rfc8419>.

[RFC8702] Kampanakis, P. and Q. Dang, "Use of the SHAKE One-Way Hash Functions in the Cryptographic Message Syntax (CMS)", RFC 8702, DOI 10.17487/RFC8702, January 2020, <https://www.rfc-editor.org/info/rfc8702>.

11. Informative References

[ECRYPT.CSA.D5.4] University of Bristol, "Algorithms, Key Size and Protocols Report (2018)", March 2015, <https://www.ecrypt.eu.org/csa/documents/D5.4-FinalAlgKeySizeProt.pdf>.

[I-D.ietf-lamps-lightweight-cmp-profile]
Brockhaus, H., Fries, S., and D. Oheimb, "Lightweight CMP Profile", Work in Progress, Internet-Draft, draft-ietf-lamps-lightweight-cmp-profile-04, 2 November 2020, <https://tools.ietf.org/html/draft-ietf-lamps-lightweight-cmp-profile-04>.

[NIST.SP.800-57pt1r5]
Barker, Elaine., "Recommendation for key management:part 1 - general", NIST NIST SP 800-57pt1r5, DOI 10.6028/NIST.SP.800-57pt1r5, May 2020, <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf>.

[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>.

Appendix A. Algorithm Use Profiles

This appendix provides profiles of algorithms and respective conventions for different application use cases.

A.1. Algorithm selection guideline

To promote interoperability, based on the recommendations of NIST SP 800-57 Recommendation for Key Management [NIST.SP.800-57pt1r5] and ECRYPT Algorithms, Key Size and Protocols Report (2018) [ECRYPT.CSA.D5.4], the following choices are RECOMMENDED:

< To be done. >

A.2. Algorithm Profile for PKI Management Message Profiles

The following table contains definitions of algorithms used within PKI Management Message Profiles as defined in CMP Appendix D.2 [RFC4210].

The columns in the table are:

Name: an identifier used for message profiles

Use: description of where and for what the algorithm is used

Mandatory: algorithms which MUST be supported by conforming implementations

Name	Use	Mandatory
MSG_SIG_ALG	protection of PKI messages using signature	RSA
MSG_MAC_ALG	protection of PKI messages using MACing	PasswordBasedMac
SYM_PENC_ALG	symmetric encryption of an end entity's private key where symmetric key is distributed out-of-band	AES-wrap
PROT_ENC_ALG	asymmetric algorithm used for encryption of (symmetric keys for encryption of) private keys transported in PKIMessages	D-H
PROT_SYM_ALG	symmetric encryption algorithm used for encryption of private key bits (a key of this type is encrypted using PROT_ENC_ALG)	AES

Table 1

Mandatory Algorithm Identifiers and Specifications:

RSA: sha256WithRSAEncryption with 2048 bit, see Section 3.1

PasswordBasedMac: id-PasswordBasedMac, see Section 6.1 (with id-sha256 as the owf parameter, see Section 2.1 and id-hmacWithSHA256 as the mac parameter, see Section 6.2.1)

D-H: id-alg-ESDH, see Section 4.1.1

AES-wrap: id-aes256-wrap, see Section 4.3.1

AES: id-aes256-CBC, see Section 5.1

< To be checked. >

A.3. Algorithm Profile for Lightweight CMP Profile

The following table contains definitions of algorithms which MUST be supported by conforming implementations This profile is referenced in the Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile].

The columns in the table are:

Name: an identifier used for message profiles

Use: description of where and for what the algorithm is used

Mandatory: algorithms which MUST be supported by conforming implementations (only if a PKI management operation using the respective algorithms is supported)

Name	Use	Mandatory
MSG_SIG_ALG	protection of PKI messages using signature	RSA, ECDSA
MSG_MAC_ALG	protection of PKI messages using MACing	PasswordBasedMac
KM_KA_ALG	asymmetric key agreement algorithm used for agreement of a symmetric keys for encryption of EnvelopedData, e.g., a private key transported in PKIMessages	D-H, ECDH
KM_KT_ALG	asymmetric key encryption algorithm used for transport of a symmetric keys for encryption of EnvelopedData, e.g., a private key transported in PKIMessages	RSA

Name	Use	Mandatory
KM_PB_ALG	symmetric derivation algorithm used to derive a symmetric key for encryption of EnvelopedData, e.g., a private key transported in PKIMessages, from a password	PBKDF2
KM_KW_ALG	symmetric key encryption algorithm to encrypt a content encryption key	AES-wrap
PROT_SYM_ALG	symmetric content encryption algorithm used for encryption of, e.g., private key bits (a key of this type is encrypted using PROT_ENC_ALG)	AES

Table 2

Mandatory Algorithm Identifiers and Specifications:

RSA: sha256WithRSAEncryption with 2048 bit, see Section 3.1

ECDSA: ecdsa-with-SHA256 with curve SECP-256, see Section 3.2

PasswordBasedMac: id-PasswordBasedMac, see Section 6.1 (with id-sha256 as the owf parameter, see Section 2.1 and id-hmacWithSHA256 as the mac parameter, see Section 6.2.1)

D-H: id-alg-ESDH, see Section 4.1.1

ECDH: dhSinglePass-stdDH-sha256kdf-scheme, see Section 4.1.2

RSA: rsaEncryption with 2048 bit, see Section 4.2.1

PBKDF2: id-PBKDF2, see Section 4.4.1

AES-wrap: id-aes256-wrap, see Section 4.3.1

AES: id-aes256-CBC, see Section 5.1

< To be checked. >

Appendix B. History of changes

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

From version 01 -> 02:

*Added Hans Aschauer, Mike Ounsworth, and Serge Mister as co-author

*Changed to XML V3

*Added SHAKE digest algorithm to Section 2 as discussed at IETF 109

*Deleted DSA from Section 3 as discussed at IETF 109

*Added RSASSA-PSS with SHAKE to Section 3

*Added SECP curves the section on ECDSA with SHA2, ECDSA with SHAKE, and EdDSA to Section 3 as discussed at IETF 109

*Deleted static-static D-H and ECDH from Section 4.1 based on the discussion on the mailing list (see thread "[CMP Algorithms] Section 4.1.1 and 4.1.2 drop static-static (EC)DH key agreement algorithms for use in CMP")

*Added ECDH OIDs and SECP curves, as well as ECDH with curve25519 and curve448 to Section 4.1 as discussed at IETF 109

*Deleted RSA-OAEP from Section 4.2 first as discussed at IETF 109, but re-added it after discussion on the mailing list (see thread "Mail regarding draft-ietf-lamps-cmp-algorithms")

*Added a paragraph to Section 4.3.1 to explain that the algorithms and key length for content encryption and key wrapping must be aligned as discussed on the mailing list (see thread "[CMP Algorithms] Use Key-Wrap with or without padding in Section 4.3 and Section 5")

*Deleted AES-CCM and AES-GMC from and added AES-CBC to Section 5 as discussed at IETF 109

*Added Section 6.1.2 to offer PBMAC1 as discusses on the mailing list (see thread "Mail regarding draft-ietf-lamps-crmf-update-algs-02") and restructured text in Section 6 to be easier to differentiate between password- and shared-key-based MAC

*Deleted Diffie-Hellmann based MAC from Section 6 as is only relevant when using enrolling Diffie-Hellmann certificates

*Added AES-GMAC and SHAKE-based KMAC to Section 6 as discussed at IETF 109

*Extended Section 9 to mention Russ supporting with two additional I-Ds and name further supporters of the draft

*Added a first draft of a generic algorithm selection guideline to Appendix A

*Added a first proposal for mandatory algorithms for the Lightweight CMP Profile to Appendix A

*Minor changes in wording

From version 00 -> 01:

*Changed sections Symmetric Key-Encryption Algorithms and Content Encryption Algorithms based on the discussion on the mailing list (see thread "[CMP Algorithms] Use Key-Wrap with or without padding in Section 4.3 and Section 5")

*Added Appendix A with updated algorithms profile for RDC4210 Appendix D.2 and first proposal for the Lightweight CMP Profile

*Minor changes in wording

Authors' Addresses

Hendrik Brockhaus (editor)
Siemens AG

Email: hendrik.brockhaus@siemens.com

Hans Aschauer
Siemens AG

Email: hans.aschauer@siemens.com

Mike Ounsworth
Entrust

Email: mike.ounsworth@entrust.com

Serge Mister
Entrust

Email: serge.mister@entrust.com