rfc6712bis and rfc4210bis

draft-ietf-lamps-rfc6712bis-03
Hendrik Brockhaus, David von Oheimb, Mike Ounsworth, John Gray

draft-ietf-lamps-rfc4210bis-06
Hendrik Brockhaus, David von Oheimb, Mike Ounsworth, John Gray

Hendrik Brockhaus

IETF 116 – LAMPS Working Group
Activities since IETF 115 on rfc6712bis

Changes since IETF 115:
• Minor editorial update preventing expiration
• Moved the draft to github.com/lamps-wg/cmp-updates/
  converting from XML to MD

March 2023
Hendrik Brockhaus - Siemens
Activities since IETF 115 on rfc4210bis

Changes since IETF 115:

- Added section on POP for KEM keys
- Added a proposal for message protection using KEM keys and HPKE in version -04/-05 and updated this proposal in -06 to using plain KEM and KDF without HPKE
- Updated guidance on which CMS-based key management to use with encrypted values
- Added a text regarding use of Certificate Transparency
- Moved the draft to github.com/lamps-wg/cmp-updates/converting from XML to MD

March 2023

Hendrik Brockhaus - Siemens
Message protection using KEMs – HPKE vs. plain KEM+KDF

CMP message protection

• KEM certificate to deliver an authenticated public KEM key.
• This public KEM key is to be used to establish a shared secret.
• A KDF is to be used to derive a shared secret key.
• This shared secret key is to be used for MAC-based message protection.

The authors believe that using plain KEM+KDF as proposed for CMS (see draft-ietf-lamps-cms-kemri) is more straightforward than using HPKE because confidentiality (seal/open) is needed.

➔ Shall we move forward using plain KEM+KDF?
Message protection using KEMs – Concatenation of two keys vs. separate keys on both sides

CMP message protection
• KEM certificate to deliver an authenticated public KEM key.
• This public KEM key is to be used to establish a shared secret.
• A KDF is to be used to derive a shared secret key.
• This shared secret key is to be used for MAC-based message protection.

Both sides could derive one shared secret key resulting from the concatenation of the two KEM shared secrets. This results in one shared secret key used by both sides for MAC calculation and verification.

Alternatively, both sides could derive two shared secret keys, one from the shared secret resulting its decapsulate operation and one resulting from the encapsulate operation. This results in two shared secret keys used on both sides, one for MAC calculation and the other for MAC verification (and vice versa). This approach could also be used like one side uses a KEM key pair and the other uses a signature key pair.

➔ Shall we move forward deriving two different shared secret keys?
Client and server use shares symmetric key

**CMP Client**
- clientKemCert, \( skC \)

**CMP Server**
- serverKemCert, \( skS \)

1. **genm message**, PKIBody.KemCiphertext absent, clientKemCert in extraCerts
2. **genp message**, ct1 in PKIBody.KemCiphertext, serverKemCert in extraCerts
3. **Kdf(ikm*, \( \text{len} \), \( \text{ukm}* \) → \( \text{ssk} \)**

\[
\text{Kdf(ikm*, len, ukm*) → ssk}
\]

1. **Kem.decaps(\( skC \), ct1) → ss1**
2. **Kem.encaps(pkC) → (ct1, ss1)**

1. **Kem.decaps(\( skS \), ct2) → ss2**
2. **Kdf(ikm, \( \text{len} \), \( \text{ukm}* \) → \( \text{ssk} \)**

\[
\text{Kdf(ikm, len, ukm*) → ssk}
\]

1. **ir message**, ct2 in generalInfo.KemCiphertext, **MAC-based protection using ssk**
2. **ip message**, **MAC-based protection using ssk**
3. **certConf message**, **MAC-based protection using ssk**
4. **pkiconf message**, **MAC-based protection using ssk**

\[
\text{*) ikm = concat(ss1, ss2)}
\]
\[
\text{ukm = concat("CMP-KEM", transactionID, genp_senderNonce, genp_recipNonce)}
\]
Client and server use different secret keys

**CMP Client**
- clientKemCert, skC

  genm message,
  PKIBody.KemCiphertext absent,
  clientKemCert in extraCerts

  Kem.decaps(pkC) $\rightarrow$ (ct1, ss1)

  Kdf(ss1, len, ukm*) $\rightarrow$ ssk1

  Kem.encaps(pkS) $\rightarrow$ (ct2, ss2)

  Kdf(ss2, len, ukm*) $\rightarrow$ ssk2

**CMP Server**
- serverKemCert, skS

  genp message,
  ct1 in PKIBody.KemCiphertext,
  serverKemCert in extraCerts

  Kem.encaps(pkC) $\rightarrow$ (ct1, ss1)

  Kdf(ss1, len, ukm*) $\rightarrow$ ssk1

  Kem.decaps(skS, ct2) $\rightarrow$ ss2

  Kdf(ss2, len, ukm*) $\rightarrow$ ssk2

  ir message,
  ct2 in generalInfo.KemCiphertext,
  MAC-based protection using ssk1

  ip message, MAC-based protection using ssk2

  certConf message, MAC-based protection using ssk1

  pkiconf message, MAC-based protection using ssk2

- $\text{ukm} = \text{concat}("\text{CMP-KEM}", \text{transactionID}, \text{genp_senderNonce}, \text{genp_recipNonce})$