rfc6712bis and rfc4210bis

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

draft-ietf-lamps-rfc4210bis-07
Hendrik Brockhaus, David von Oheimb, Mike Ounsworth, John Gray

Hendrik Brockhaus

IETF 118 – LAMPS Working Group
Activities since IETF 117 on rfc6712bis

Changes since IETF 117:
• No updates since IETF117
• Draft on github.com/lamps-wg/cmp-updates/

Next Steps:
• Alignment of the draft after publication of CMP Updates (RFC 9480)
Activities since IETF 117 on rfc4210bis

Changes since IETF 117:
• No updates since IETF117
• Some changes in the pipeline at github.com/lamps-wg/cmp-updates/

Status:
• PoC implementation are ongoing during the hackathon
• Waiting for review feedback from the WG as requested by Russ during IETF 116 (“Russ: requests that the KEM section get a lot of review since this part is very new.”)
  Specifically opinions on the content of the KemOtherInfo is welcome.

Next Steps:
• Alignment of the draft after publication of CMP Updates (RFC 9480)
• Resolving issues in text on proof-of-possession structures using challenge-response, Section 5.2.8.3.
New ASN.1 structures

id-KemBasedMac OBJECT IDENTIFIER ::= {1 2 840 113533 7 66 TBD4}

KemBMParameter ::= SEQUENCE {
    kdf          AlgorithmIdentifier{KEY-DERIVATION, {...}},
    len          INTEGER (1..MAX),
    mac          AlgorithmIdentifier{MAC-ALGORITHM, {...}}
}

id-it-KemCiphertextInfo OBJECT IDENTIFIER ::= {id-it TBD1}
KemCiphertextInfoValue ::= KemCiphertextInfo

KemCiphertextInfo ::= SEQUENCE {
    kem          AlgorithmIdentifier{KEM-ALGORITHM, {...}},
    ct           OCTET STRING
}

KemOtherInfo ::= SEQUENCE {
    staticString          PKIFreeText,
    transactionID [0] OCTET STRING OPTIONAL,
    senderNonce [1] OCTET STRING OPTIONAL,
    recipNonce [2] OCTET STRING OPTIONAL,
    len           INTEGER (1..MAX),
    mac          AlgorithmIdentifier{MAC-ALGORITHM, {...}},
    ct            OCTET STRING
}

November 2023

Algorithm identifier to be used in PKIHeader.protectionAlg when KEM-based MAC is used.
Entrust is willing to register the OID in the same branch like PBMParameter.

InfoTypeAndValue to deliver the KEM ciphertext in body of general message or in generalInfo field of message header.

Context information as input to the KDF for domain separation and for ensuring uniqueness of MAC-keys.
Uses transactionID, senderNonce, and recipNonce from the message containing the KemCiphertextInfoValue.ct, if present.
Client owns KEM key pair

**CMP Client**
- clientKemCert, skC

- kem.decapsulate(skC, ct) $\rightarrow$ ss
- KemBMParameter(KDF AlgID, len, MAC AlgID)
- DER-encoded KemOtherInfo $\rightarrow$ context
- kdf(ss, len, context) $\rightarrow$ ssk

- genm message, PKIBody.KemCiphertextInfoValue absent, clientKemCert in extraCerts, no message protection

- ip message, protection depending on server credentials
- certConf message, MAC-based protection using ssk
- pkiconf message, protection depending on server credentials

**CMP Server**
- clientKemCert with pkC
- kem.encapsulate(pkC) $\rightarrow$ (ct, ss)
- KemCiphertextInfo(KEM AlgID, ct)

- genp message, PKIBody.KemCiphertextInfoValue present, no message protection

- ir message
  - KemBMParameter as protectionAlg, MAC-based protection using ssk

- DER-encoded KemOtherInfo $\rightarrow$ context
  - kdf(ss, len, context) $\rightarrow$ ssk
  - server authenticated message from client

Mandatory

Optional
Server owns KEM key pair

**CMP Client**
- **serverKemCert with pkS**
- `kem.encapsulate(pkS) -> (ct, ss)`
- `KemCiphertextInfo(KEM AlgID, ct)`

**CMP Server**
- `serverKemCert, skS`
- `kem.decapsulate(skS, ct) -> ss`

---

**ir message,**
- ct in `generalInfo.KemCiphertextInfoValue` present,
- protection depending on client credentials

**ip message,**
- `KemBMParameter` as protectionAlg,
- MAC-based protection using ssk

**DER-encoded KemOtherInfo -> context**
- `kdf(ss, len, context) -> ssk`

---

**certConf message,**
- protection depending on client credentials

**pkiconf message,**
- MAC-based protection using ssk