rfc6712bis and rfc4210bis

draft-ietf-lamps-rfc6712bis-05
Hendrik Brockhaus, David von Oheimb , Mike Ounsworth, John Gray

draft-ietf-lamps-rfc4210bis-12
Hendrik Brockhaus, David von Oheimb , Mike Ounsworth, John Gray

Hendrik Brockhaus

IETF 120 – LAMPS Working Group

Activities since IETF 118 on rfc6712bis

Changes since IETF 119:

- WGLC closed without change requests.

Next Steps:

- No open issues on github.
- IANA early review comments were addressed on github and will be submitted with the next version.
- Waiting for shepherd write-up and IESG evaluation.

Activities since IETF 119 on rfc4210bis

Changes since IETF 119:

- Implemented the changes discussed during IETF 119.
  - Removed normative language in Section 4.2.2.
  - Deprecated CAKeyUpdAnnContent in favor of RootCaKeyUpdateContent.
  - Solved erratum #7888.
- Clarifying use of the recipient identifier with the indirect and direct POP method.
- Fixed a nit in the ASN.1 module.
- Changed reference from ITU-T X.509 to RFC 5280.
- Added IANA considerations addressing IANA early review.
- PoC implementation of KEM-support using Bouncy Castle and OpenSSL are ongoing.
- **Extended WGLC closed on July 12th with review feedback on KEM-based message protection and POP from Thom and David H but without change requests.**

Next Steps:

- No open issues on github.
- The WG chairs to decide on WGLC result.
- Waiting for shepherd write-up and IESG evaluation.
- Add nonceRequest /nonceResponse syntax, if draft-ietf-lamps-csr-attestation and draft-ietf-lamps-attestation-freshness develop quickly.

New ASN.1 structures for KEM-based message protection

```
id-KemBasedMac OBJECT IDENTIFIER ::= {1 2 840 113533 7 66 16}

KemBMParameter ::= SEQUENCE {
  kdf               AlgorithmIdentifier{KEY-DERIVATION, {...}},
  kemContext    [0] OCTET STRING     OPTIONAL,
  len               INTEGER (1..MAX),
  mac               AlgorithmIdentifier{MAC-ALGORITHM, {...}}
}
```

Algorithm identifier to be used in PKIHeader.protectionAlg when KEM-based MAC is used.
Entrust registered the OID in the same branch as PBMParameter.
Optional kemContext if needed with the used KEM algorithm like ukm in cms-kemri.

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

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

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

```
KemOtherInfo ::= SEQUENCE {
  staticString      PKIFreeText,
  transactionID     OCTET STRING,
  kemContext    [0] OCTET STRING     OPTIONAL
}
```

Context information as input to the KDF for domain separation and for ensuring uniqueness of MAC-keys.
Uses transactionID from the message containing the KemCiphertextInfoValue.ct.
Optional kemContext if needed with the used KEM algorithm like ukm in cms-kemri.

Question on KEM-based message protection raised by Thom Wiggers

- In the KemBMParameters structure, **does the AlgorithmIdentifier structure for the kdf parameter carry the required salt and/or other labels necessary** to derive the same shared secret? (Seems weird that those parameters would be part of an AlgorithmIdentifier, but that might just be convention that I'm less familiar with). I did note that the running text does say that 'kdf' should contain any necessary parameters.

[HB] **If kdf parameter are required they are given in the parameters component of the AlgorithmIdentifier**, e.g., RFC 8018 for PBKDF2. For HKDF the parameters component is left empty, see RFC 8619, accommodating operation without salt.

- I noticed that **the key for the MAC is derived using optional, algorithm-specific KemContext information**. I'm assuming that this information is present in part to prevent re-encapsulation attacks for algorithms that are not ciphertext-binding [https://eprint.iacr.org/2023/1933] (I have not evaluated if this protocol can be vulnerable to such attacks—but eliminating them can't hurt). If this is an implicit design requirement/desirable property, this might be worth including in the security considerations? Mike probably has more informed ideas about this.

[HB] We followed the discussion on cms-kemri if the public key or the ct shall be used as additional input to the kdf. **As ML-KEM (Kyber) already uses the ct with an internal kdf it was decided not to add it again in the cms-kemri but leave it to the algorithm profiling for CMS to require it or not. The authors of rfc4210bis have followed this path.** If people think adding ct as standard input to the kdf to be on the safe side, it can be considered. If there is anything else to add to the input of the kdf, e.g., the AlgorithmIdentifier of the MAC, please speak up.

- My main question is: **what is the message input to the MAC?** I could not easily find this in the document, and the input to the MAC is relevant to the ability of shared key ssk being re-used for further PKI transactions (i.e. inputs likely must not collide).

[HB**] The message input to the MAC is the sequence of PKIHeader and the PKIBody** as specified in Section 5.1.3.

Question on KEM-based POP raised by Thom Wiggers

- I am having a hard time figuring out **how the shared secret between EE and RA/CA is computed; there should presumably be some KDF involved** like in message protection. Am I just missing where this is specified?

[HB] **We use standards CMS EnvelopedData incl. cms-kemri as is**.

- The **same comments regarding (implicitly) avoiding KEM re-encapsulation attacks** apply here. Again, I'm not sure if these attacks are theoretically applicable and/or actually a problem but seem worth briefly looking into (if only to ease any modeling).

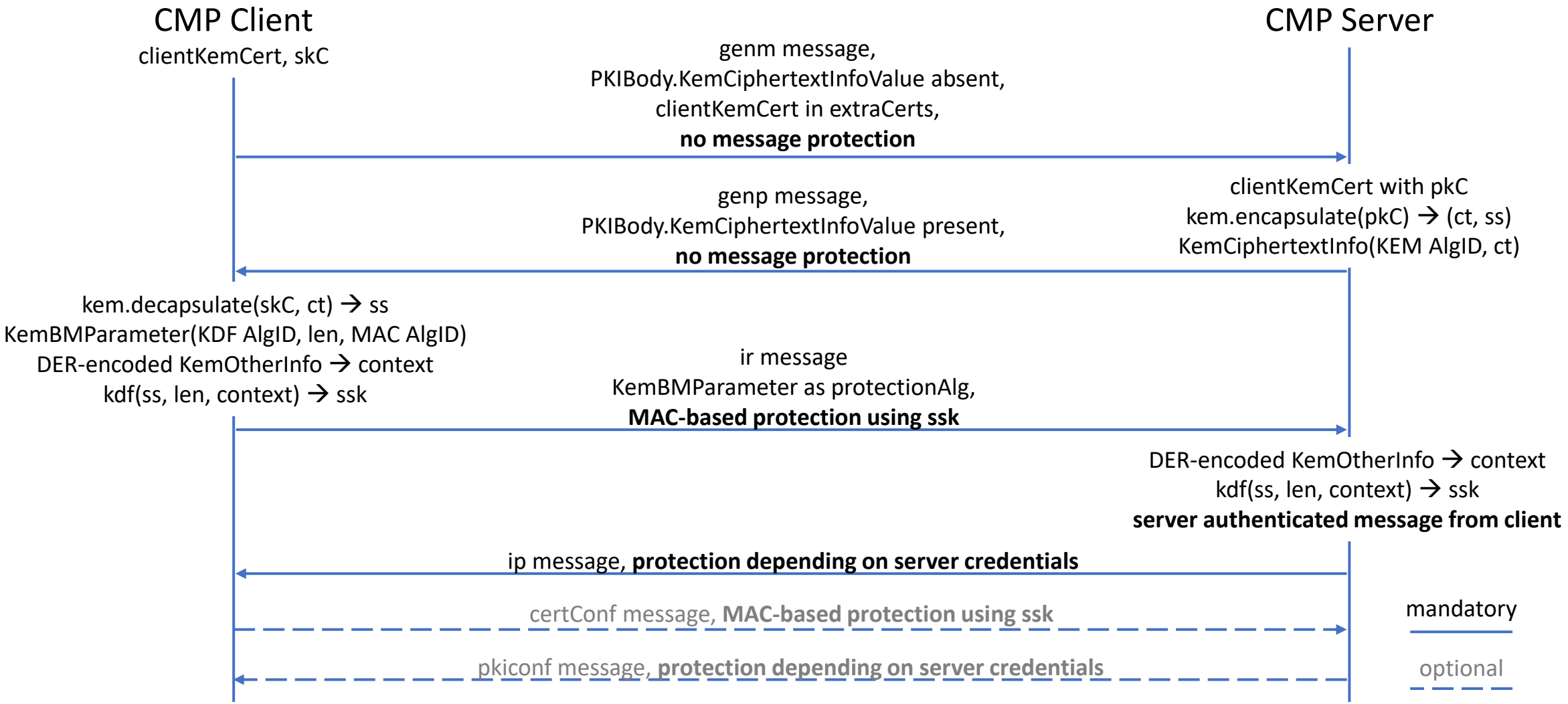
[HB] **We use standards CMS EnvelopedData incl. cms-kemri as is**.

- Regarding security consideration 8.2, **have you considered returning H(rand) instead of rand to further avoid these decryption oracles?** However, arguably the derivation of challenge encryption key should be set up in a way to sufficiently domain-separate this. Again, this comes back to how the challenge encryption keys are set up.

[HB] If no signature-based POP is possible the preferred method in CMP is the indirect method. Therefore, the direct method is a sideline. **When updating the text on the direct POP method we intended to change as little as possible to the current text and only introduced EnvelopedData instead of the challenge field containing the plain encrypted Rand as OCTET STRING.** If people think, we should return H(rand) instead of the plain integer, we can be considered it.

BAK

Client owns KEM key pair



Server owns KEM key pair