CMCbis I-Ds

IETF 121 - LAMPS WG

Joe Mandel & **Sean Turner**

Datatracker: draft-ietf-lamps-rfc5272bis & draft-ietf-lamps-rfc5273bis & draft-ietf-lamps-rfc5273bis
GitHub: CMCbis

Update

Merged PRs we noted @ IETF 120.

Added direct POP (more slides follow). CMC’s PoP background:

- CMC only supported direct POP.
- Subsequent rekey/renewals just repeat original request pattern; nothing special.

PoP Messages

Message 1: EE to CA
aka “Gimme a Cert!”

```
ContentInfo.contentType = id-signedData
 ContentInfo.content
  SignedData.encapContentInfo
   eContentType = id-cct-PKIData
   eContent
    controlSequence
     {102, id-cmc-transactionId, 10132985123483401}
     {103, id-cmc-senderNonce, 10001}
     {104, id-cmc-dataReturn, <stuff>}
    reqSequence
     certRequest
      certReqId = 201
      certTemplate
       subject = < My DN >
       publicKey = My Public Key
       extensions {id-ce-keyUsage, keyEncipherment}
      popo
       keyEncipherment
         subsequentMessage = challengeResp
  SignedData.SignerInfos
   SignerInfo
    sid = < subjectKeyIdentifier >
    signatureAlgorithm = id-alg-noSignature
```

PoP Messages

Message 2: CA to EE
aka “Nah do some PoP”

```
ContentInfo.contentType = id-signedData
 ContentInfo.content
  SignedData.encapContentInfo
   eContentType = id-cct-PKIResponse
   eContent
    controlSequence
     {101, id-cmc-statusInfoV2, {failed, 201, popRequired}}
     {102, id-cmc-transactionId, 10132985123483401}
     {103, id-cmc-senderNonce, 10005}
     {104, id-cmc-recipientNonce, 10001}
     {105, id-cmc-encryptedPOP, {
      request {
       certRequest
       certReqId = 201
       certTemplate
        subject = < My DN >
        publicKey = My Public Key
        extensions {id-ce-keyUsage, keyEncipherment}
       popo
        keyEncipherment
          subsequentMessage = challengeResp }
      cms
       contentType = id-envelopedData
       content < uses ori.KEMRecipientInfo >
        recipientInfos.ori.riid.issuerSerialNumber = < NULL, 201>
        encryptedContentInfo
         eContentType = id-data
         eContent = <Encrypted value of 'y'>
    thePOPAlgID = KmacWithSHAKE128
    witnessAlgID = SHAKE128
    witness <hashed value of 'y'> } }
     {106, id-cmc-dataReturn, <stuff>}
     Certificates
       Other certificates (optional)
     SignedData.SignerInfos
       Signed by CA
```

PoP Messages

Message 3: EE to CA
aka “POP goes the weasel”

```
ContentInfo.contentType = id-signedData
 ContentInfo.content
   SignedData.encapContentInfo
     eContentType = id-cct-PKIData
     eContent
       controlSequence
         {102, id-cmc-transactionId, 10132985123483401}
         {103, id-cmc-senderNonce, 100101}
         {104, id-cmc-dataReturn, <stuff>}
         {105, id-cmc-recipientNonce, 10005}
         {107, id-cmc-decryptedPOP, {
           bodyPartID 201,
           thePOPAlgID KmacWithSHAKE128,
           thePOP <KMAC computed value goes here>}}
       reqSequence
         certRequest
           certReqId = 201
           certTemplate
             subject = < My DN >
             publicKey = My Public Key
             extensions
               {id-ce-keyUsage, keyEncipherment}
           popo
             keyEncipherment
               subsequentMessage = challengeResp
   SignedData.SignerInfos
     SignerInfo
       sid = < subjectKeyIdentifier >
       signatureAlgorithm = id-alg-noSignature
```

PoP Messages

Message 4: CA to EE
aka “Profit!”

```
ContentInfo.contentType = id-signedData
 ContentInfo.content
  SignedData.encapContentInfo
   eContentType = id-cct-PKIResponse
   eContent
    controlSequence
     {101, id-cmc-transactionId, 10132985123483401}
     {102, id-cmc-statusInfoV2, {success, 201}}
     {103, id-cmc-senderNonce, 10019}
     {104, id-cmc-recipientNonce, 100101}
     {105, id-cmc-dataReturn, <stuff>}
  certificates
   Newly issued certificate
   Other certificates
  SignedData.SignerInfos
   Signed by CA
```

To Do: 5274bis - Cryptographic Algorithm Requirements

RFC 5274 text includes :(

- MUST verify DSA-SHA1 and RSA-SHA1 signatures in SignedData
- strongly suggested that RSA-PSS with SHA-1
- strongly suggested that SHA-256 using RSA and RSA-PSS
- MUST generate either DSA-SHA1 or RSA-SHA1 signatures
- MUST support RSA as a key transport algorithm for EnvelopedData
- SHOULD support RSA-OAEP as a key transport algorithm for EnvelopedData
- If an entity supports key agreement for EnvelopedData, it MUST support Diffie-Hellman
- …

To Do: Provide input to CSR Attestation / Attestation Freshness

---

CSR Attestation: Define CMC Control

Attestation Freshness: Need a way to ask for Nonce

> All requests are for certificates / revocation;
> no generic request :shrug: