Using Pre-Shared Key (PSK) in the Cryptographic Message Syntax (CMS)

draft-ietf-lamps-cms-mix-with-psk-03

Russ Housley
LAMPS WG at IETF 104
March 2019
Use PSK for Quantum Protection

• Open question whether a large-scale quantum computer is feasible, and if so, when it might happen
• If it happens, RSA and Diffie-Hellman and Elliptic Curve Diffie-Hellman become vulnerable
• The concern ...
  – Today: Adversary saves CMS-protected content
  – Someday: Decrypt content when a large-scale quantum computer becomes available
• The solutions ...
  – Near-term: Strong PSK as an input to the derivation of the content-encryption key
  – Long-term: Quantum-resistant public-key cryptographic algorithms (the winners of NIST competition)
Mixing with a PSK

• The draft defines two quantum-resistant ways to establish encryption keys. In both cases, a PSK MUST be distributed to the sender and all of the recipients by some out-of-band means that does not make it vulnerable to the future invention of a large-scale quantum computer, and an identifier MUST be assigned to the PSK.

• Two new OtherRecipientInfo structures:
  – KeyTransPSKRecipientInfo
  – KeyAgreePSKRecipientInfo
Overview

1. The content-encryption key is generated at random.
2. The key-derivation key is generated at random.
3. The key-encryption key is established for each recipient:
   - **key transport**: the key-derivation key is encrypted in the recipient's public key, then the key derivation function (KDF) is used to mix the pre-shared key (PSK) and the key-derivation key to produce the key-encryption key; or
   - **key agreement**: the recipient's public key and the sender’s private key are used to generate a pairwise symmetric key, then the key derivation function (KDF) is used to mix the pre-shared key (PSK) and the pairwise symmetric key to produce the key-encryption key.
4. The key-encryption key is used to encrypt the content-encryption key.
Summary of Recent Changes

- Changed explanation of key agreement to use KEK1 and KEK2

- PSK part of the 'info' structure for input to KDF

```plaintext
CMSORIforPSKOtherInfo ::= SEQUENCE {
  psk OCTET STRING,
  keyMgmtAlgType ENUMERATED {
    keyTrans (5),
    keyAgree (10) },
  keyEncryptionAlgorithm KeyEncryptionAlgorithmIdentifier,
  pskLength INTEGER (1..MAX),
  kdkLength INTEGER (1..MAX) }
```

- Added examples in Appendix A and Appendix B
Please Review

• I think the draft is ready for WG Last Call

• Please review the draft
• Can someone check the examples?
• Please send comments to the mail list

• Tim will make all LAMPS WG consensus calls related to this document