

# KDF for content encryption

IETF 118

Russ Housley

# Summary of the attack disclosed by Falko and Johannes

Attacker intercepts a CMS Authenticated-Enveloped-Data content [RFC5083] that uses either AES-CCM or AES-GCM [RFC5084].

Then, the attacker turns the intercepted content into a "garbage" CMS Enveloped-Data content [RFC5652] that is composed of AES-CBC guess blocks.

Then, send the "garbage" to the victim, and the victim shares the result of the decryption with the attacker. If any of the transformed plaintext blocks match  $H_t$ , then the attacker learns the plaintext for that block.

# Mitigation of the attack

***The attack is thwarted if the identifier for the encryption algorithm cannot be changed.***

Proposal has three parts:

- Assign OID for an unprotected attribute to indicate this mitigation is being used
- Potential recipients include the OID (no parameters) in S/MIME Capabilities to advertize support for this mitigation
- Encryption with  $CEK' = HKDF(CEK, AlgorithmIdentifier)$

# Example (1 of 2)

CEK = c702e7d0a9e064b09ba55245fb733cf3

The AES-128 CGM AlgorithmIdentifier:

algorithm=2.16.840.1.101.3.4.1.6

parameters=GCMParameters:

aes-nonce=0x5c79058ba2f43447639d29e2

In hex: 301b0609608648016503040106300e040c5c79058ba2f43447639d29e2

CEK' = HKDF(CEK, AlgorithmIdentifier)

CEK' = 4ae85bd6d45e990a401e5f8fc093d6d2

# Example (2 of 2)

CEK = c702e7d0a9e064b09ba55245fb733cf3

The AES-128 CBC AlgorithmIdentifier:

algorithm=2.16.840.1.101.3.4.1.2

parameters=AES\_IV=0x651f722ffd512c52fe072e507d72b377

In hex:

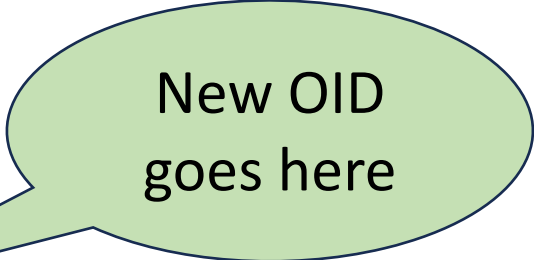
301d06096086480165030401020410651f722ffd512c52fe072e507d72b377

CEK' = HKDF(CEK, AlgorithmIdentifier)

CEK' = 474fd8239b7fa5e011862a59465ab369


# EnvelopedData

```
EnvelopedData ::= SEQUENCE {  
  version CMSVersion,  
  originatorInfo [0] IMPLICIT OriginatorInfo OPTIONAL,  
  recipientInfos RecipientInfos,  
  encryptedContentInfo EncryptedContentInfo,  
  unprotectedAttrs [1] IMPLICIT UnprotectedAttributes OPTIONAL }
```



New OID  
goes here

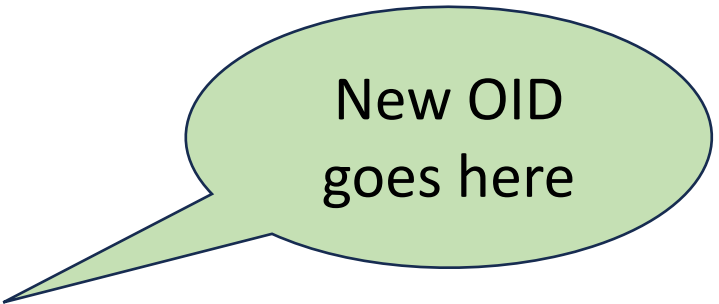
```
EncryptedContentInfo ::= SEQUENCE {  
  contentType ContentType,  
  contentEncryptionAlgorithm ContentEncryptionAlgorithmIdentifier,  
  encryptedContent [0] IMPLICIT EncryptedContent OPTIONAL }
```



Encrypt with HKDF(CEK,  
ContentEncryption  
AlgorithmIdentifier)

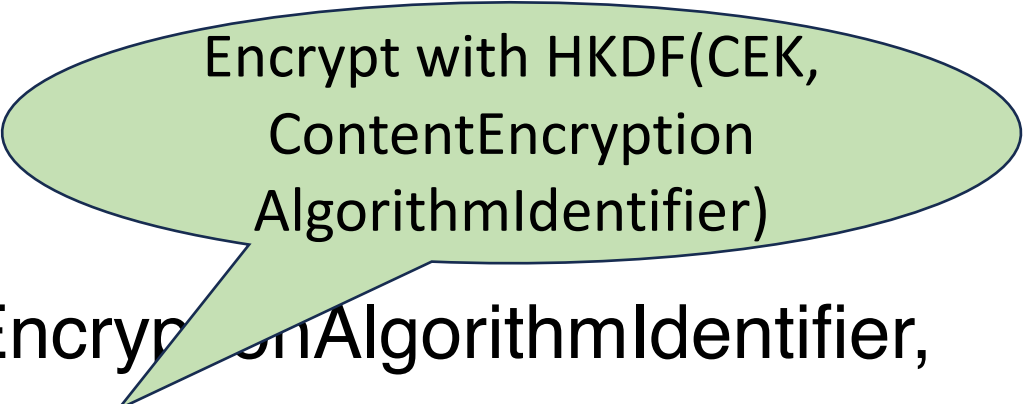
# EncryptedData

```
EncryptedData ::= SEQUENCE {  
  version CMSVersion,  
  encryptedContentInfo EncryptedContentInfo,  
  unprotectedAttrs [1] IMPLICIT UnprotectedAttributes OPTIONAL }
```



New OID  
goes here

```
EncryptedContentInfo ::= SEQUENCE {  
  contentType ContentType,  
  contentEncryptionAlgorithm ContentEncryptionAlgorithmIdentifier,  
  encryptedContent [0] IMPLICIT EncryptedContent OPTIONAL }
```



Encrypt with HKDF(CEK,  
ContentEncryption  
AlgorithmIdentifier)

# AuthEnvelopedData

```
AuthEnvelopedData ::= SEQUENCE {  
  version CMSVersion,  
  originatorInfo [0] IMPLICIT OriginatorInfo OPTIONAL,  
  recipientInfos RecipientInfos,  
  authEncryptedContentInfo EncryptedContentInfo,  
  authAttrs [1] IMPLICIT AuthAttributes OPTIONAL,  
  mac MessageAuthenticationCode,  
  unauthAttrs [2] IMPLICIT UnauthAttributes OPTIONAL }
```

New OID  
goes here

AEAD Encrypt with  
HKDF(CEK,  
ContentEncryption  
AlgorithmIdentifier)

```
EncryptedContentInfo ::= SEQUENCE {  
  contentType ContentType,  
  contentEncryptionAlgorithm ContentEncryptionAlgorithmIdentifier,  
  encryptedContent [0] IMPLICIT EncryptedContent OPTIONAL }
```



# Works with all flavors of RecipientInfo

- KeyTransRecipientInfo [RFC5652]
- KeyAgreeRecipientInfo [RFC5652]
- KEKRecipientInfo [RFC5652]
- PasswordRecipientInfo [RFC5652]
- KeyTransPSKRecipientInfo [RFC8696]
- KeyAgreePSKRecipientInfo [RFC8696]
- KEMRecipientInfo [I-D.ietf-lamps-cms-kemri]

# Design Rationale

- Use HKDF with SHA-256, and avoid negotiation of a KDF
- If the attacker removes the OID from the unprotected attributes, then the recipient will use a different key to try to decrypt the content
  - The attack fails
  - The recipient is denied access to the "garbage" message content
- If the attacker changes the algorithm identifier, then the recipient will use a different key to try to decrypt the content
  - The attack fails
  - The recipient is denied access to the "garbage" message content

# Way Forward

- Publish an Internet-Draft with this mitigation
- Proceed with publication of draft-ietf-lamps-cms-kemri
- Early assignment of the new OID
- Gain development and deployment experience
- Publish as standards-track RFC
- Publish rfc8551bis to require this mitigation (S/MIME 4.1)