Use of HPKE with COSE

draft-ietf-cose-hpke-07

IETF118, 2023-11-07
H. Tschofenig, O. Steele, D. Ajitomi, L. Lundblade
Status Update

-06 (2023-10-06)
  ○ Switched from a-la-carte approach to ciphersuite approach based on the result of the Consensus Call.
    ■ Introduced encapsulated_key parameter instead of hsi (HPKE Sender Info) parameter.
    ■ Defined a ciphersuite naming convention, such as HPKE-Base-P256-SHA256-AES128GCM
    ■ Introduced 16 ciphersuites an as a rough draft. (discuss the details later)
  ○ Changed authors: +O. Steele, +D. Ajitomi, +L. Laurence, -B. Moran

-07 (2023-10-22)
  ○ Re-generated the examples using a draft-06 compliant implementation.
  ○ Added an example for the COSE_Mac structure.
Status Update

a-la-carte approach (- draft-05)

ciphersuite approach (draft-06 -)

merged into the alg value
Remaining Issues: Summary
https://github.com/cose-wg/HPKE/issues

○ **Ciphersuite selection.**
  ■ (#36) Reduce the number of ciphersuites.

○ **Context Information Structure**
  ■ (#24) Externally Supplied AAD only processed at layer 0.
  ■ (#25) Empty String for Info Value.
  ■ (#44) COSE_Recipient header protection

○ **Key representation**
  ■ (#35) COSE Elliptic Curve needs to be added for the x25519/Kyber hybrid.
  ■ (#46) Key representation

○ **Editorial issues** (improve readability, remove redundancies)
  ■ (#41) Could reduce restatement of COSE requirements.
  ■ (#44) Specify in terms of Seal() and Open APIs.
<table>
<thead>
<tr>
<th>Cipher Suite Label</th>
<th>KEM</th>
<th>HPKE</th>
<th>AEAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>COSE-HPKE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cipher Suite Label</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HPKE-Base-P256-SHA256-AES128GCM</td>
<td>0x10</td>
<td>0x1</td>
<td>0x1</td>
</tr>
<tr>
<td>HPKE-Base-P256-SHA256-ChaCha20Poly1305</td>
<td>0x10</td>
<td>0x1</td>
<td>0x3</td>
</tr>
<tr>
<td>HPKE-Base-P384-SHA384-AES256GCM</td>
<td>0x11</td>
<td>0x2</td>
<td>0x2</td>
</tr>
<tr>
<td>HPKE-Base-P384-SHA384-ChaCha20Poly1305</td>
<td>0x11</td>
<td>0x2</td>
<td>0x3</td>
</tr>
<tr>
<td>HPKE-Base-P521-SHA512-AES256GCM</td>
<td>0x12</td>
<td>0x3</td>
<td>0x2</td>
</tr>
<tr>
<td>HPKE-Base-P521-SHA512-ChaCha20Poly1305</td>
<td>0x12</td>
<td>0x3</td>
<td>0x3</td>
</tr>
<tr>
<td>HPKE-Base-X25519-SHA256-AES128GCM</td>
<td>0x20</td>
<td>0x1</td>
<td>0x1</td>
</tr>
<tr>
<td>HPKE-Base-X25519-SHA256-ChaCha20Poly1305</td>
<td>0x20</td>
<td>0x1</td>
<td>0x3</td>
</tr>
<tr>
<td>HPKE-Base-X448-SHA512-AES256GCM</td>
<td>0x21</td>
<td>0x3</td>
<td>0x2</td>
</tr>
<tr>
<td>HPKE-Base-X448-SHA512-ChaCha20Poly1305</td>
<td>0x21</td>
<td>0x3</td>
<td>0x3</td>
</tr>
<tr>
<td>HPKE-Base-X25519Kyber768-SHA256-AES256GCM</td>
<td>0x30</td>
<td>0x1</td>
<td>0x2</td>
</tr>
<tr>
<td>HPKE-Base-X25519Kyber768-SHA256-ChaCha20Poly1305</td>
<td>0x30</td>
<td>0x1</td>
<td>0x3</td>
</tr>
<tr>
<td>HPKE-Base-CP256-SHA256-ChaCha20Poly1305</td>
<td>0x13</td>
<td>0x1</td>
<td>0x3</td>
</tr>
<tr>
<td>HPKE-Base-CP256-SHA256-AES128GCM</td>
<td>0x13</td>
<td>0x3</td>
<td>0x1</td>
</tr>
<tr>
<td>HPKE-Base-CP521-SHA512-ChaCha20Poly1305</td>
<td>0x15</td>
<td>0x3</td>
<td>0x3</td>
</tr>
<tr>
<td>HPKE-Base-CP521-SHA512-AES256GCM</td>
<td>0x15</td>
<td>0x3</td>
<td>0x2</td>
</tr>
</tbody>
</table>
Shape the initial ciphersuite list

How should the list be constructed?

A) Adopt the same set of the ciphersuites defined in MLS (Messaging Layer Security).
B) Add some combinations of the DHKEMs with NIST curves and ChaCha20Poly1305 to A).
C) Add some combinations using the DHKEMs with compact NIST curves to B).
D) Add some combinations using a post-quantum hybrid KEM (X25519Kyber768) to C).

- Plan is to go for A + B, i.e.
  - remove the compact NIST curve based DHKEMs, and
  - remove the post-quantum hybrid KEMs.

- Comments?
  - Can always be extended later!
Proposed Ciphersuite List
(green + orange marked list)

- **A)** 7 ciphersuites from RFC9180 (same as the MLS choices).
  - HPKE-Base-P256-SHA256-AES128GCM (MLS-0x0002)
  - HPKE-Base-P384-SHA384-AES256GCM (MLS-0x0007)
  - HPKE-Base-P521-SHA512-AES256GCM (MLS-0x0005)
  - HPKE-Base-X25519-SHA256-AES128GCM (MLS-0x0001)
  - HPKE-Base-X25519-SHA256-ChaCha20Poly1305 (MLS-0x0003)
  - HPKE-Base-X448-SHA512-AES256GCM (MLS-0x0004)
  - HPKE-Base-X448-SHA512-ChaCha20Poly1305 (MLS-0x0006)

- **B)** +3 for adding some combinations with ChaCha20Poly1305.
  - HPKE-Base-{P256-SHA256, P384-SHA384, P521-SHA512}-ChaCha20Poly1305

- **C)** +6 for compact NIST curve-based DHKEMs.
  - HPKE-Base-C256-SHA256-AES128GCM
  - HPKE-Base-{C384-SHA384, C521-SHA512}-AES256GCM
  - HPKE-Base-{C256-SHA256, C384-SHA384, C521-SHA512}-ChaCha20Poly1305

- **D)** +2 for post-quantum hybrid KEMs
  - HPKE-Base-X25519Kyber768-SHA256-{AES256GCM, ChaCha20Poly1305}
Key Representation

```
{
  "kty": "EC",
  "crv": "P-256",
  "alg": "HPKE-Base-P256-SHA256-AES128GCM",
  "kid": "test-key-42",
  "x": "xXCWZk-jG9Tjd7M361sAEUi8JvKBxFlQgqhqkZa5cgs",
  "y": "y-9jpXy5gNhxl9BV4smqO36MXRlbkrC3PvvjDOrpOgU",
  "use": "enc",
  "key_ops": [
    "deriveBits"
  ]
}
```

Considering to add brief text about how to represent public and private keys for use with HPKE.

Should be non-controversial.
Our Plan

- Handle remaining issues towards draft-08.
- Add more examples / test vectors.
- Get draft-08 ready for WGLC.