COSE Key and JWK Representation for HPKE KEM

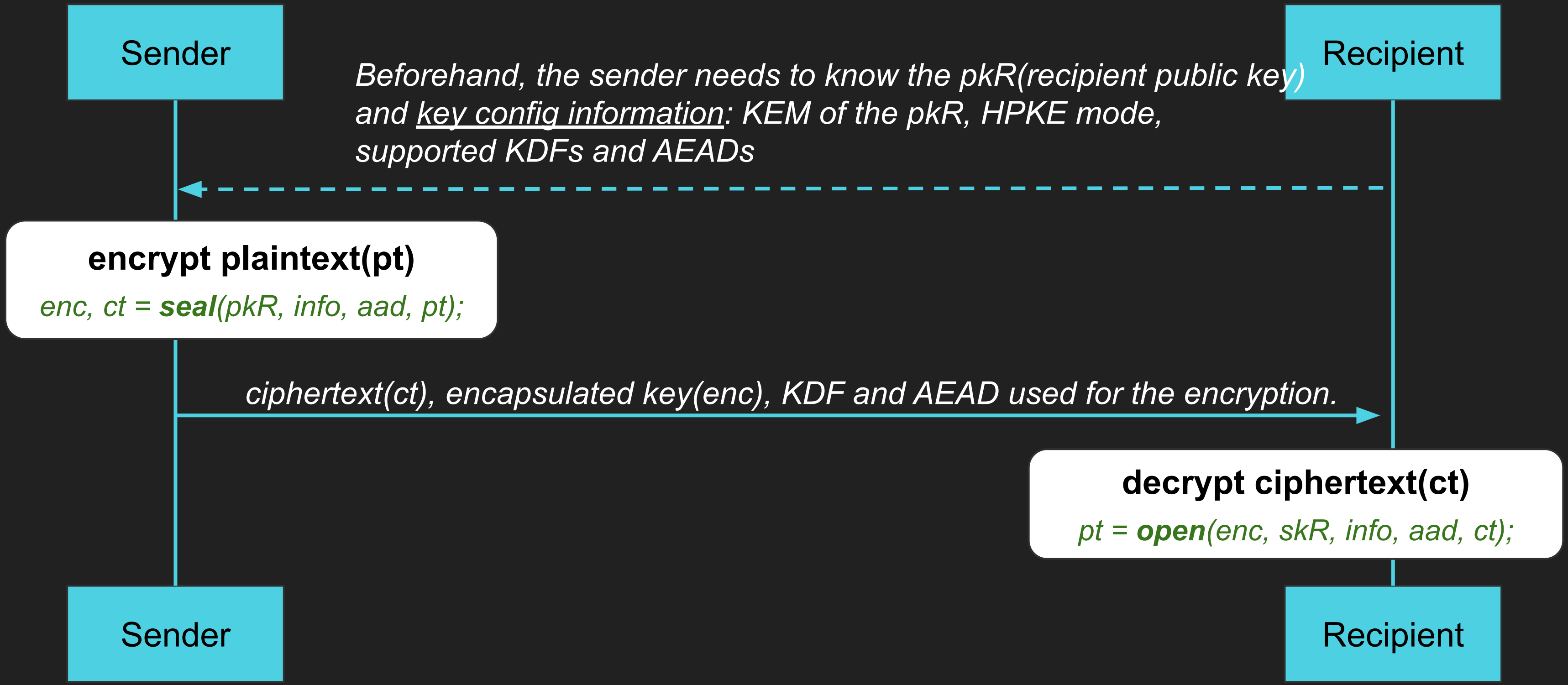
https://datatracker.ietf.org/doc/draft-ajitomi-cose-cose-key-jwk-hpke-kem/

AJITOMI Daisuke

Background

- RFC9180: Hybrid Public Key Encryption (HPKE)
  - https://www.rfc-editor.org/rfc/rfc9180.html
  - Defines a scheme for hybrid public key encryption which works with any combination of asymmetric KEM, KDF and AEAD.
  - Has already been adopted by TLS ECH, OHTTP, ODoH, etc.
- draft-ietf-cose-hpke-03: Use of HPKE with COSE (COSE-HPKE)
  - https://datatracker.ietf.org/doc/draft-ietf-cose-hpke/
  - Defines how to use HPKE with COSE for encrypting a payload or a CEK.
  - Supposed to be used for “Firmware Encryption with SUIT Manifests”.

HPKE Transaction



Sender

Recipient

Beforehand, the sender needs to know the pkR(recipient public key) and key config information: KEM of the pkR, HPKE mode, supported KDFs and AEADs

encrypt plaintext(pt)

enc, ct = ***seal****(pkR, info, aad, pt);*

ciphertext(ct), encapsulated key(enc), KDF and AEAD used for the encryption.

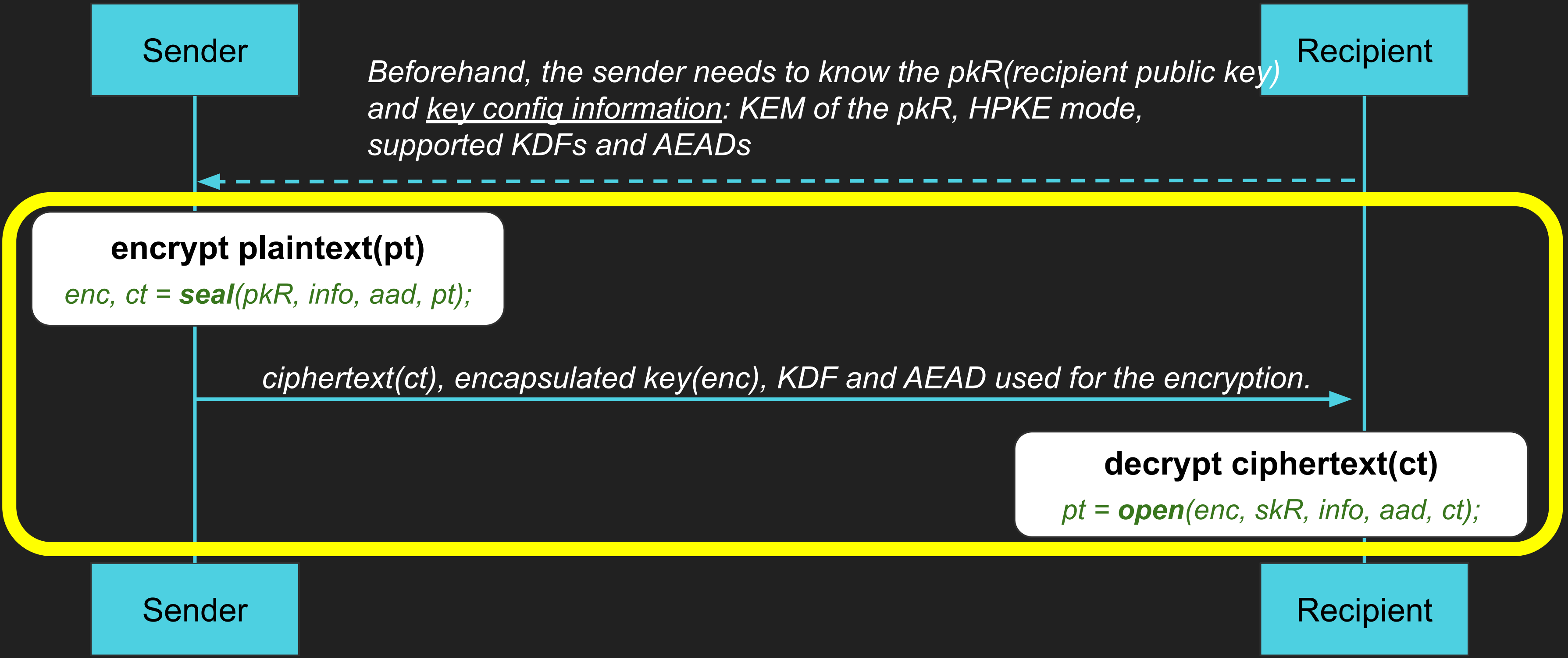
decrypt ciphertext(ct)

pt = ***open****(enc, skR, info, aad, ct);*

Sender

Recipient

The Scope of the COSE-HPKE Draft



Sender

Recipient

Beforehand, the sender needs to know the pkR(recipient public key) and key config information: KEM of the pkR, HPKE mode, supported KDFs and AEADs

encrypt plaintext(pt)

*enc, ct = **seal**(pkR, info, aad, pt);*

ciphertext(ct), encapsulated key(enc), KDF and AEAD used for the encryption.

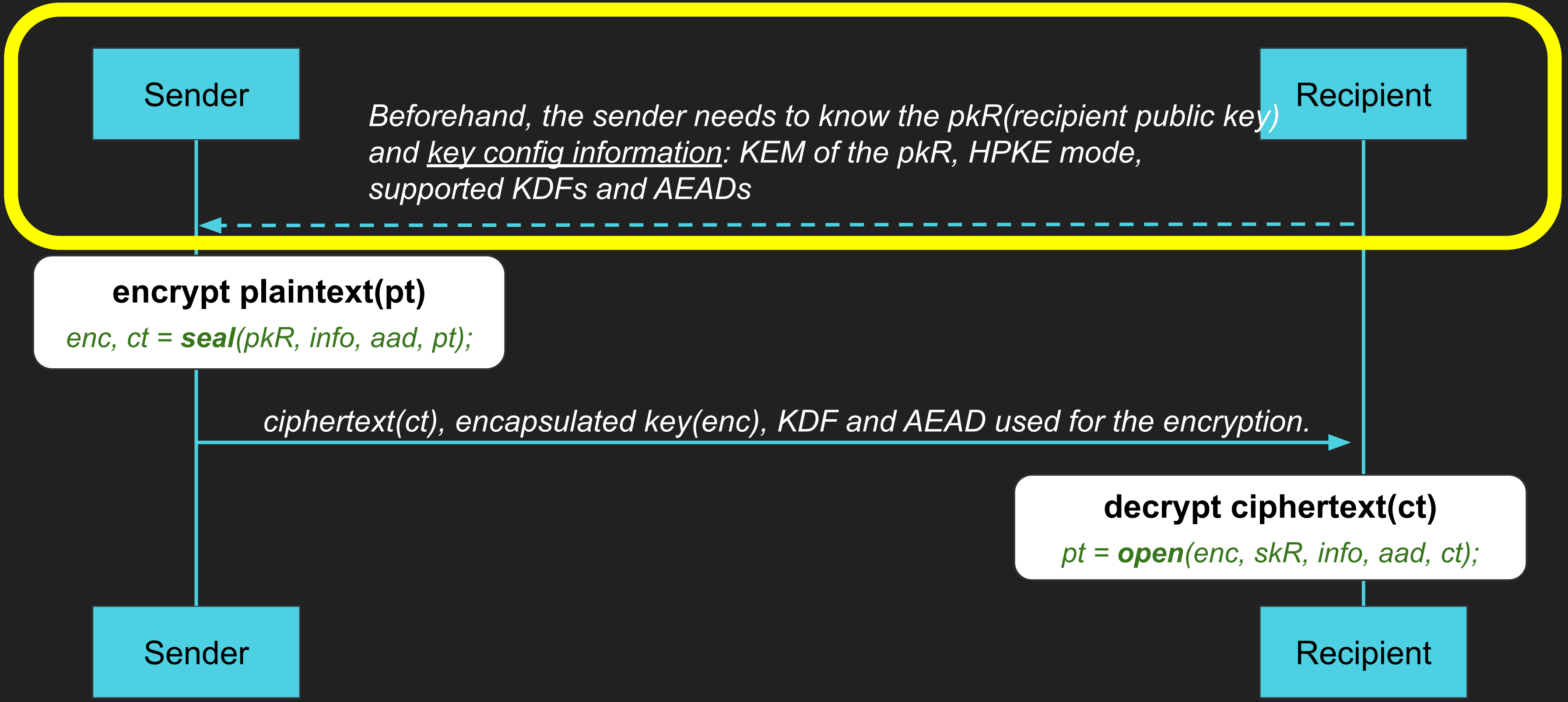
decrypt ciphertext(ct)

*pt = **open**(enc, skR, info, aad, ct);*

Sender

Recipient

The Scope of this Proposal



Beforehand, the sender needs to know the pkR(recipient public key) and key config information: KEM of the pkR, HPKE mode, supported KDFs and AEADs

encrypt plaintext(pt)

enc, ct = ***seal****(pkR, info, aad, pt);*

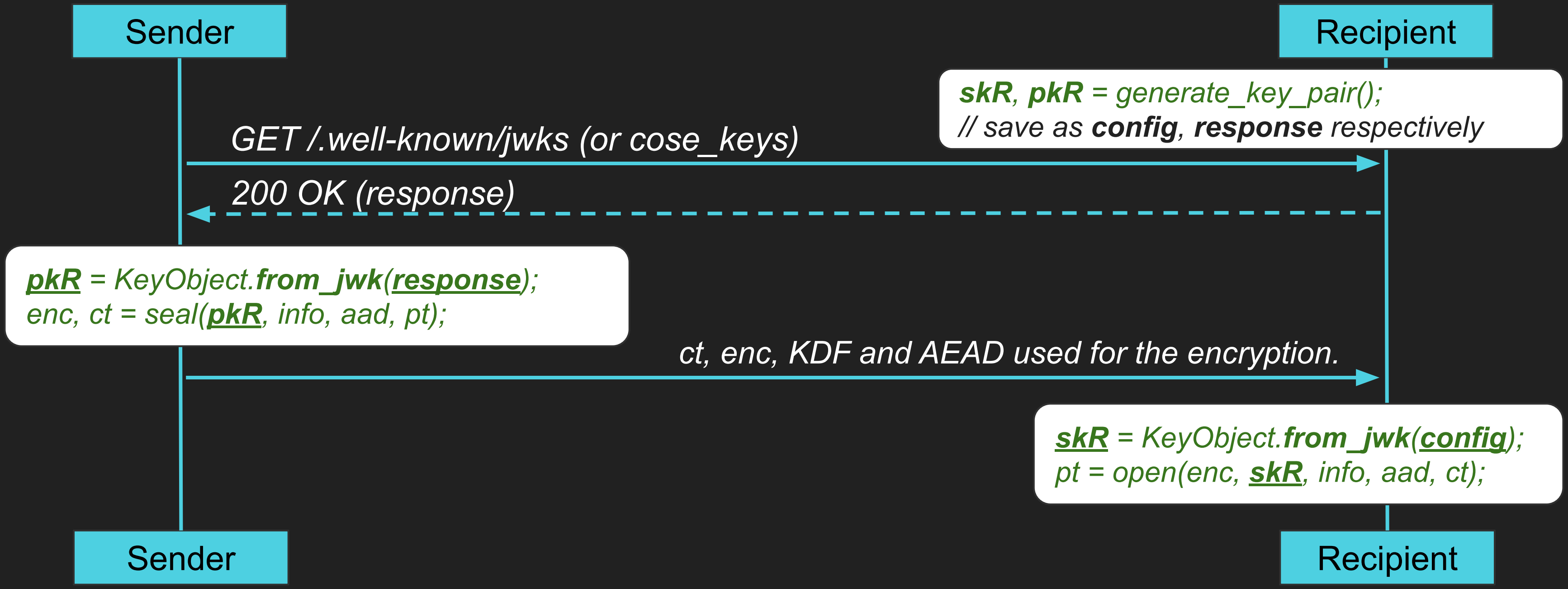
ciphertext(ct), encapsulated key(enc), KDF and AEAD used for the encryption.

decrypt ciphertext(ct)

pt = ***open****(enc, skR, info, aad, ct);*

Use Cases

- COSE Key and JWK Representation for HPKE KEM can be used for transmitting the pkR and key config information and for storing them as config data.



Sender

Recipient

***skR*, *pkR* = generate_key_pair();**
*// save as **config**, **response** respectively*

GET /.well-known/jwks (or cose_keys)

200 OK (response)

***pkR** = KeyObject.**from_jwk**(**response**);*
*enc, ct = seal(**pkR**, info, aad, pt);*

ct, enc, KDF and AEAD used for the encryption.

***skR** = KeyObject.**from_jwk**(**config**);*
*pt = open(enc, **skR**, info, aad, ct);*

Sender

Recipient

COSE Key and JWK Representation for HPKE KEM

Defines:

1. a generic key type (“kty”) for HPKE, which can also represent post-quantum KEM keys to be defined in the future, and its algorithm values (“alg”).
   - “kty”: “HPKE-KEM”
   - “alg”: “HPKE-v1-Base” | “HPKE-v1-PSK” | “HPKE-v1-Auth” | “HPKE-v1-AuthPSK”
2. a new common key parameter (“hkc”) for representing the HPKE key config information both for the “HPKE-KEM” and for the existing key types used for key derivation. The “hkc” contains an object consisting of the following attributes:

   -
     ```
     “hkc”: {
         “kem”: 0x0010, // The HPKE KEM identifier associated with the pkR.
         “kdfs”:  0x0001, // The HPKE KDF identifiers supported by the recipient.
         “aeads”: 0x0002, // The HPKE AEAD identifiers supported by the recipient.
     }
     ```

   The KEM/KDF/AEAD identifiers are two-byte value registered in the HPKE IANA registry. This eliminates the need to define new “kty”s and “alg”s for future-defined post-quantum KEMs.

Examples

```
// JWK for DHKEM(X25519, KDF-SHA256) Public Key with kty "HPKE-KEM"
{
   "kty": "HPKE-KEM",
   "kid": "01",
   "alg": "HPKE-v1-Base",
   "hkc": {
      "kem": 0x020,
      "kdfs": [0x001, 0x002, 0x003],
      "aeads": [0x001, 0x002]
   },
   "pub": "y3wJq3uXPHeoCO4FubvTc7VcBuqpvUrSvU6ZMbHDTCI"
}
```

```
// COSE_Key for DHKEM(X25519, KDF-SHA256) Public Key with kty HPKE-KEM
{
   1:-1(T.B.D.),  // HPKE-KEM
   2:'01',
   3:-1(T.B.D),   // HPKE-v1-Base
   6(T.B.D): [    // hkc (HPKE Key Configuration)
      0x0020,                       // KEM identifier
      [0x0001, 0x0002, 0x0003],  // supported KDF identifiers
      [0x0001, 0x0002]              // supported AEAD identifiers
   ],
   -1:h'd75a980182b10ab7d54bfed3c964073a0ee172f3daa62325af021…'
}
```

```
// JWK for DHKEM(P-256, KDF-SHA256) Public Key with existing kty "EC"
{
   "kty": "EC",
   "kid": "01",
   "crv": "P-256",
   "alg": "HPKE-v1-Base",
   "hkc": {
      "kem": 0x010,
      "kdfs": [0x001, 0x002, 0x003],
      "aeads": [0x001, 0x002]
   },
   "x": "-eZXC6nV-xgthy8zZMCN8pcYSeE2XfWWqckA2fsxHPc",
   "y": "BGU5soLgsu_y7GN2I3EPUXS9EZ7Sw0qif-V70JtInFI"
}
```

```
// COSE_Key for DHKEM(P-256, KDF-SHA256) Public Key with existing kty EC2
{
   1: 2,            // EC2
   2: '01',
   -1: 1,           // P-256
   3: -1(T.B.D),  // HPKE-v1-Base
   6(T.B.D): [    // hkc (HPKE Key Configuration)
      0x0010,                       // KEM identifier
      [0x0001, 0x0002, 0x0003],  // supported KDF identifiers
      [0x0001, 0x0002]              // supported AEAD identifiers
   ],
   -2:h'65eda5a12577c2bae829437fe338701a10aaa375e1bb5b5de10…',
   -3:h'1e52ed75701163f7f9e40ddf9f341b3dc9ba860af7e0ca7ca7e9e…'
}
```

Controversial Points

Received some feedback from Ilari, Orie and Laurence (Thanks!):

- Should the draft be specialized for the COSE_Key representation?
  - I believe the JWK representation should be defined in the draft as well.
    - JWK representation can be used for COSE.
      - ex) EUDCC is CWT but the public keys for its verification are published as JWKs.
    - JOSE-HPKE will be needed as an alternative to ECDH-ES-* sooner or later.
- Can the kty ”HPKE-KEM” be accepted?
  - It's reasonable to associate a key type with the purpose of the key, but this differs from existing key types ("EC", “RSA”), which are defined for specific cryptographic algorithms.
- Should we support existing key types?
  - If the kty ”HPKE-KEM” can be accepted, the support for the existing key types might lead the implementation problems and some kind of confusion.
- Should the draft focus on the HPKE “Base” mode?
  - I prefer to define all of the HPKE modes in the draft because the “hkc” structure should be independent of the HPKE modes.

Next Steps

- Any comments?
- Interest in adopting this proposal into the WG?