A MODEST PROPOSAL FOR HPKE

DAN HARKINS, HPE

IETF 111, JULY 2021

HPKE is great,

but…

HPKE has some bloat*

Serialization produces a public key that is > 2x that which is necessary

Problematic for:

- constrained devices or constrained environments
- “one shot” API usage with short messages

HPKE supports “compact output”– the secret is the x-coordinate only of the ECDH shared point

- If the sign doesn’t matter to the secret then it doesn’t matter to the public keys
- Public keys can be serialized as the x-coordinate only-- ”compact representation”

Compact output compels use of compact representation!

* For KEMs using P-256, P-384, and P-521

HPKE has some bloat*

`p256 with SEC uncompressed serialization:`

```
041e9081e36299e5d4c7c30f3eeae1b4ccef32b9953f6cd3cf97e3fb76cd7e6791a0e8aee893612305ae4f32b2f9f0219ecede8fabd96ee07c063f802b43731471
```

`p256 with RFC 6090 compact output serialization:`

```
931538d413b9ca3692a9e7ec34ec0d29db62715eee8044fa06e5cd111d17f30b
```

* For KEMs using P-256, P-384, and P-521

HPKE assumes guaranteed, in-order delivery

- Nonce management is entirely in the HPKE context
- No way to know which nonce was used with a given ciphertext
- No way to synchronize after loss/reorder
- No way to even notice except everything suddenly stops working
- Packet loss or packet reordering is tragic with HPKE

The Internet is not guaranteed, in-order delivery

HPKE APIs don't, and shouldn't, care about nonces

- Users not managing nonces is A Good Thing™
- Addressing loss/reordering shouldn't change the APIs
- Just need to ensure that loss/reordering isn't tragic

Need deterministic authenticated encryption cipher modes

- There are no nonces to worry about
- Each packet can be decrypted *in situ*

"*why not just export a key from the context and do any cipher you want outside of HPKE?*"

Because I want to use HPKE; if I wanted a static-ephemeral ECDH key exchange I'd use one.

Security Proof for DAE is in the paper*

TL;DR: It's deterministic, so it cannot conceal whether a plaintext+AAD combo was encrypted twice in a sequence of ciphertexts– i.e. it's not IND-CCA2

Implications for using a DAE cipher mode:

- Some use cases won't care– idempotent messages, or just want AE
- Some use cases can ensure something "new" in each message:
  - Put time-since-epoch in the AAD or, to obtain privacy, as a tweak in the plaintext
  - If any bit in the AAD or plaintext is "new" then the synthetic IV will look random and therefore the output will, likewise, look random– adversary is no longer able to determine whether the same plaintext(+AAD) was encrypted twice

A reasonable approach to obtain resistance to packet loss/reordering!

* Rogaway, P. and T. Shrimpton,
"Deterministic Authenticated Encryption, A Provable-Security Treatment of the Key-Wrap Problem",
EUROCRYPT '06, Saint Petersberg, Russia, 2006

Proposal: Add the following…

Compact representation
- New KEMs for NIST curves
- Compact representation (RFC 6090)

Deterministic Authenticated Encryption
- Support for AES-SIV (RFC 5297)
- No nonce generated/used in HPKE context

Internet-Draft:

draft-harkins-cfrg-dnhpke-00

Source code:

https://github.com/danharkins/hpke-wrap

- Compliant with -10 of HPKE
- Supports compact representation with new KEM values*
- Supports deterministic authenticated encryption ciphers*
- Complete test vectors (based on the latest version of HPKE test vectors)

* Took the liberty of stealing some values reserved to IANA for test vector generation