Converting HPKE to be PQ

Norbert Pócs

February 4 2023

Red Hat

Overview

Hybrid **P**ublic **K**ey **E**ncryption

- asymmetric + symmetric encryption
- scheme using KEM (Key Encapsulation Mechanism)
- RFC 9180 (Barnes et al. 2022)

Fundamental parts

KEM

P-256

P-384

P-521

X25519

X448

Key Schedule

HKDF-SHA256

HKDF-SHA384

HKDF-SHA512

AEAD

AES-128-GCM

AES-256-GCM

ChaCha20Poly1305

Fundamental parts

KEM operations

- Key generation
- Encapsulation
- Decapsulation

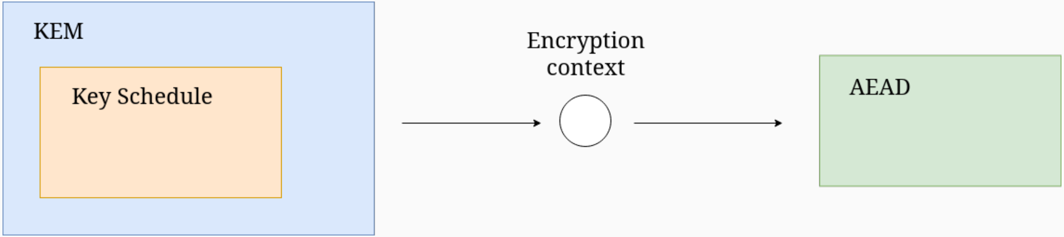
Key schedule operations

- KDF (Key Derivation Function)
  - Extract
  - Expand

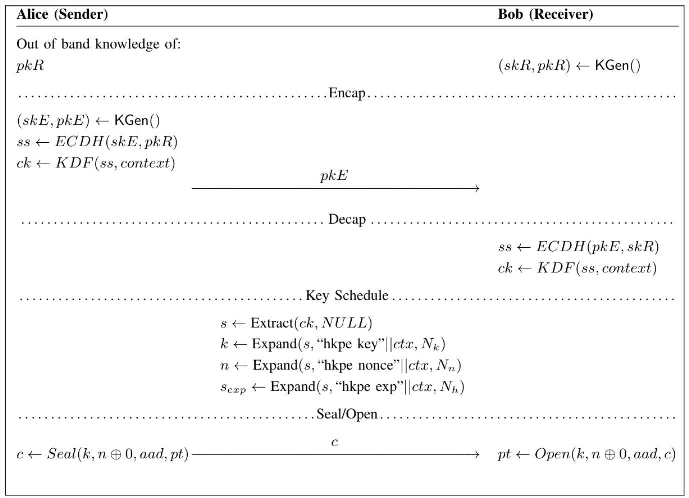
AEAD operations

- Seal
- Open

The mechanism itself



The mechanism itself



Alice (Sender) **Bob (Receiver)**

Out of band knowledge of:

pkR $(skR, pkR) \leftarrow \mathsf{KGen}()$

Encap

$(skE, pkE) \leftarrow \mathsf{KGen}()$

$ss \leftarrow ECDH(skE, pkR)$

$ck \leftarrow KDF(ss, context)$

$\xrightarrow{pkE}$

Decap

$ss \leftarrow ECDH(pkE, skR)$

$ck \leftarrow KDF(ss, context)$

Key Schedule

$s \leftarrow \text{Extract}(ck, NULL)$

$k \leftarrow \text{Expand}(s, \text{“hkpe key”}||ctx, N_k)$

$n \leftarrow \text{Expand}(s, \text{“hkpe nonce”}||ctx, N_n)$

$s_{exp} \leftarrow \text{Expand}(s, \text{“hkpe exp”}||ctx, N_h)$

Seal/Open

$c \leftarrow Seal(k, n \oplus 0, aad, pt) \xrightarrow{c} pt \leftarrow Open(k, n \oplus 0, aad, c)$

Figure 1: HPKE overview (Anastasova, Kampanakis, and Massimo 2022)

Use cases, applications

Possible usages

- Messaging layer Security (MLS)
- TLS ClientHello
- Oblivious DNS over HTTPS (ODoH)

Modes

HPKE modes

- Base mode
- Authentication modes:
    - Auth mode
    - PSK mode
    - AuthPSK mode

Security of HPKE

- Base mode
    - IND-CCA2 secure[1]
- Authenticated modes
    - Outsider-CCA secure
    - Insider-CCA secure[2]

[1] Full report here (Campagna and Petcher 2020)

[2] Full report here (Alwen et al. 2021)

PQ HPKE

KEM

P-256, ..

X25519, ..

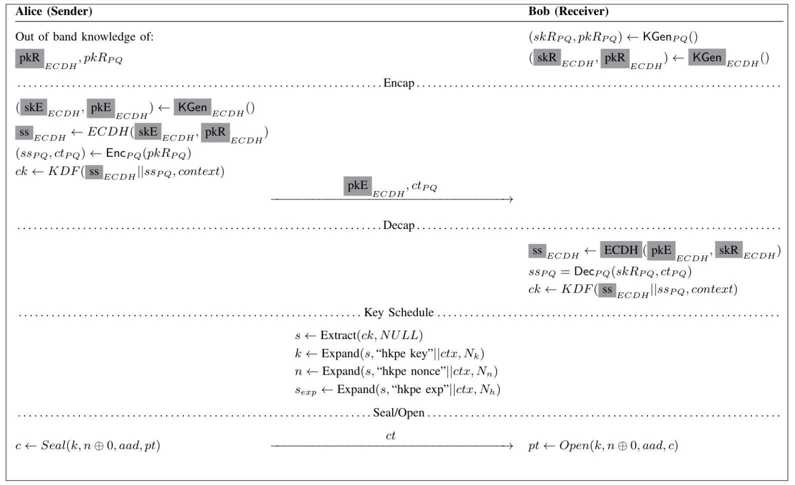
+

Kyber

~~SIKE~~

- SIKE is out of game
- Kyber is one of the NIST finalists for KEX
- KEM instead of DH style KEX
- lattice based scheme - learning with errors and rounding problem (MLWER)
- IND-CCA2 secure

Hybrid PQ HPKE



Alice (Sender)

Out of band knowledge of:

$\mathrm{pkR}_{ECDH}, pkR_{PQ}$

Bob (Receiver)

$(skR_{PQ}, pkR_{PQ}) \leftarrow \mathsf{KGen}_{PQ}()$

$(\mathrm{skR}_{ECDH}, \mathrm{pkR}_{ECDH}) \leftarrow \mathsf{KGen}_{ECDH}()$

Encap

$(\mathrm{skE}_{ECDH}, \mathrm{pkE}_{ECDH}) \leftarrow \mathsf{KGen}_{ECDH}()$

$\mathrm{ss}_{ECDH} \leftarrow ECDH(\mathrm{skE}_{ECDH}, \mathrm{pkR}_{ECDH})$

$(ss_{PQ}, ct_{PQ}) \leftarrow \mathsf{Enc}_{PQ}(pkR_{PQ})$

$ck \leftarrow KDF(\mathrm{ss}_{ECDH} || ss_{PQ}, context)$

$\xrightarrow{\mathrm{pkE}_{ECDH},\, ct_{PQ}}$

Decap

$\mathrm{ss}_{ECDH} \leftarrow \mathrm{ECDH}(\mathrm{pkE}_{ECDH}, \mathrm{skR}_{ECDH})$

$ss_{PQ} = \mathsf{Dec}_{PQ}(skR_{PQ}, ct_{PQ})$

$ck \leftarrow KDF(\mathrm{ss}_{ECDH} || ss_{PQ}, context)$

Key Schedule

$s \leftarrow \text{Extract}(ck, NULL)$

$k \leftarrow \text{Expand}(s, \text{“hkpe key”}||ctx, N_k)$

$n \leftarrow \text{Expand}(s, \text{“hkpe nonce”}||ctx, N_n)$

$s_{exp} \leftarrow \text{Expand}(s, \text{“hkpe exp”}||ctx, N_h)$

Seal/Open

$c \leftarrow Seal(k, n \oplus 0, aad, pt)$ $\xrightarrow{ct}$ $pt \leftarrow Open(k, n \oplus 0, aad, c)$

Figure 2: PQ-only and PQ-hybrid HPKE Overview (Anastasova, Kampanakis, and Massimo 2022)

Security of PQ versions

- PQ only Base mode is still IND-CCA2 secure as long as the underlying KEM is IND-CCA2 safe
- PQ hybrid needs more proof
- Authentication mode for both would need more work to prove Outsider-CCA and Insider-CCA

Benchmarks

HW of the test machine

Intel(R) Core(TM) i7-10610U CPU (4 cores and 8MB of cache) with 8GB of RAM running 1.80GHz with maximum turbo frequency of up to 4.90 GHz. For our implementations we used the AWS-LC cryptographic library.[3]

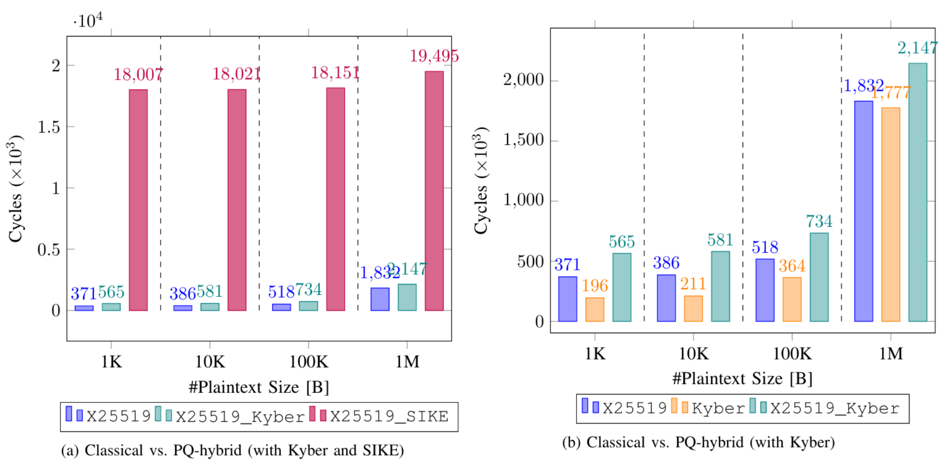
Test variables

We ran our experiments for each algorithm 1,000 times. To increase our accuracy, we eliminated the first and fourth quartile of our measurements. Additionally, all our results include the mean of the measured algorithm in CPU clock cycles.[4]

[3] (Anastasova, Kampanakis, and Massimo 2022)

[4] (Anastasova, Kampanakis, and Massimo 2022)

Benchmarks graphs

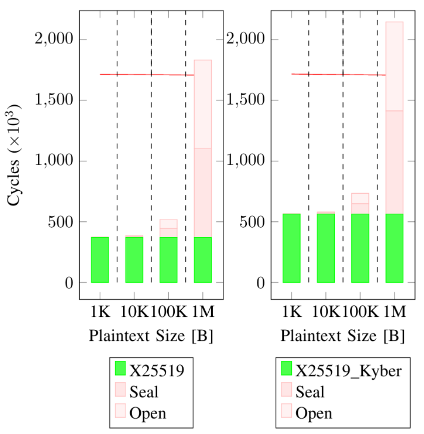


(a) Classical vs. PQ-hybrid (with Kyber and SIKE)

(b) Classical vs. PQ-hybrid (with Kyber)

Figure 3: Classical vs PQ HPKE Performance (Anastasova, Kampanakis, and Massimo 2022)

Benchmarks graphs



(a) Classical HPKE (b) Hybrid HPKE (Kyber)

Figure 4: Performance Breakdown for Classical and PQ-hybrid HPKE(Anastasova, Kampanakis, and Massimo 2022)

References

Alwen, Joël, Bruno Blanchet, Eduard Hauck, Eike Kiltz, Benjamin Lipp, and Doreen Riepel. 2021. "Analysing the HPKE Standard." In *Eurocrypt 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 12696:87–116. Lecture Notes in Computer Science. Zagreb, Croatia: Springer International Publishing. https://doi.org/10.1007/978-3-030-77870-5/_4.

Anastasova, Mila, Panos Kampanakis, and Jake Massimo. 2022. "PQ-HPKE: Post-Quantum Hybrid Public Key Encryption." Cryptology ePrint Archive, Paper 2022/414. https://eprint.iacr.org/2022/414.

Barnes, Richard, Karthikeyan Bhargavan, Benjamin Lipp, and Christopher A. Wood. 2022. "Hybrid Public Key Encryption." Request for Comments. RFC 9180; RFC Editor. https://doi.org/10.17487/RFC9180.

Campagna, Matthew, and Adam Petcher. 2020. "Security of Hybrid Key Encapsulation." Cryptology ePrint Archive, Paper 2020/1364. https://eprint.iacr.org/2020/1364.

Thank you for attention!