Standardization efforts for PQC in OpenPGP in the Project PQC@Thunderbird

Stephan Ehlen[BSI], Andreas Hülsing[TU/e], Evangelos Karatsiolis[MTG], **Stavros Kousidis**[BSI], Johannes Roth[MTG], **Falko Strenzke**[MTG], Christian Tobias[MTG] **Aron Wussler**[Proton]

BSI: German Federal Office for Information Security
MTG: MTG AG, Germany
Proton: Proton AG, Switzerland
TU/e: University of Eindhoven

Design Criteria

Design Criteria

- Use **composite** multi-algorithm (classic + PQC, a.k.a. hybrid) for Kyber and Dilithium, standalone for SPHINCS$^+$
- Backwards compatibility:
  - Having two different certificates (v4/v5)
  - Multiple signatures on the protocol level
- As classical algorithms we propose ECC:
  - "fix" all previously existing inconsitencies regarding data formats
    - i.e. native format for CFRG curves

Algorithm Choices

Kyber512 + X25519	**MUST**
Kyber1024 + X448	SHOULD
Kyber768 + ECDH-NIST-P-384	MAY
Kyber1024 + ECDH-NIST-P-521	MAY
Kyber768 + ECDH-brainpoolP384r1	MAY
Kyber1024 + ECDH-brainpoolP512r1	MAY
Dilithium2 + Ed25519	**MUST**
Dilithium5 + Ed448	SHOULD
Dilithium3 + ECDSA-NIST-P-384	MAY
Dilithium5 + ECDSA-NIST-P-521	MAY
Dilithium3 + ECDSA-brainpoolP384r1	MAY
Dilithium5 + ECDSA-brainpoolP512r1	MAY
SPHINCS$^+$-simple-SHA2	SHOULD
SPHINCS$^+$-simple-SHAKE	MAY

SPHINCS$^+$ Parameters

SPHINCS$^+$-simple-**SHA2**-128s	SHOULD
SPHINCS$^+$-simple-**SHA2**-128f	SHOULD
SPHINCS$^+$-simple-**SHA2**-192s	SHOULD
SPHINCS$^+$-simple-**SHA2**-192f	SHOULD
SPHINCS$^+$-simple-**SHA2**-256s	SHOULD
SPHINCS$^+$-simple-**SHA2**-256f	SHOULD
SPHINCS$^+$-simple-**SHAKE**-128s	MAY
SPHINCS$^+$-simple-**SHAKE**-128f	MAY
SPHINCS$^+$-simple-**SHAKE**-192s	MAY
SPHINCS$^+$-simple-**SHAKE**-192f	MAY
SPHINCS$^+$-simple-**SHAKE**-256s	MAY
SPHINCS$^+$-simple-**SHAKE**-256f	MAY

Kyber-KEM

Basic design paradigms:

- use ECDH / X25519 / X448 as KEMs
  - omit the key derivation step and output a shared key
- derive the KEK from the ECDH and Kyber shared keys
  - use SHA3-based simple concatenate-and-hash construction with some fixed info

PQC and Hash-and-Sign

- SPHINCS$^+$ and Dilithium are not following the simple hash-and-sign paradigm
  - SPHINCS$^+$ uses randomized hashing, not to rely on the collision resistance of the hash function
  - Dilithium prepends the public key
- OpenPGP v5 signatures also features randomized hashing but the details differ

Hashing in PQC Schemes vs. v5 Signatures

	SPHINCS$^+$	**Dilithium**	**v5 signatures**
Hash algo	SHA2 / SHA3	SHA3	SHA2 / SHA3
Salt size	128, 192, or 256 bit	N/A	128 bit

- Depending on the SPHINCS$^+$ security level, SPHINCS$^+$ hash-and-sign v5 signatures will be weaker than original SPHINCS$^+$.
- In order to preserve the security level of SPHINCS$^+$, a larger salt value in v5 signatures is necessary for some parameters.
- Dilithium uses only SHA3 hashing. We considered binding the hash function to the algorithm ID.

Signature Concatenation

The goal is backwards compatibility to legacy clients

- E-mail: concatenate two signatures
  - For instance Thunderbird and Proton clients currently process only the first signature
  - Classical signature followed by PQC
- OpenPGP messages
  - Multiple signatures already specified
  - State of implementation support apparently not optimal
  - Need additional testing in interoperability suite

Next Steps

- Wait for publication of Kyber IP results from NIST
- Publication of the draft
  - Currently draft is still under construction
  - Expected publication Nov. or Dec. '22
- Implementations
  - Proton already has an experimental go implementation
  - MTG will work on implementations:
    - in Libgcrypt/OpenPGP, Botan/RNP/Thunderbird
    - covering all algorithms proposed here
    - work from Jan. '23 to Nov. '23
- Improve the testing suite to include the missing tests

Questions for the WG

What do you think of:

- The algorithm selection?
- Binding the signature salt size to the hash ID?
- Binding the hash function to the algorithm ID?

Any feedback on the draft is very welcome!

An open discussion will follow in Mezzanine 12 starting at 14:50
(link for remote participants in the side-meetings wiki)