HAETAE, SMAUG-T 표준화를 위한 분석 및 개선 방안

Hyeongmin Choe

CryptoLab, Inc.

July 15th, 2025

2025 KpqC Workshop, 아일랜드 리솜

HEAAN CRYPTO LAB

CRYPTO LAB
주식회사 크립토랩

Summary

- Possible Modifications to SMAUG-T and HAETAE
  - Prefix hashing for SMAUG-T's FO transform & Symmetric Primitives
  - No special changes for HAETAE

- Security Analysis
  - Multi-target/user security
  - Beyond SUF-CMA

- Standardizing SMAUG-T and HAETAE

Prefix hashing

- Shared key (K) generation during Encap
  - Put H(pk) for multi-user security
  - Public-to-Public Hash

$\underline{\text{Encap}(\text{pk} = (\text{seed}_{\mathbf{A}}, \mathbf{b}))}$

1: $\mu \leftarrow \{0,1\}^{256}$
2: $(K, \text{seed}) \leftarrow G(\mu, H(\text{pk}))$
3: ct $\leftarrow$ SMAUG-T.PKE.Enc(pk, μ; seed)
4: **return** ct, K

- Introduced in [CCS:DHKLS21] and suggested by TU/e Reports
  - Put (part of) pk, with sufficient entropy
    - Pk=(seed of A, b) → first 256 bits of b
  - One less hash (on long pk) in Encap & Decap
    - 15-25% reduction in AVX2 cycles

$\underline{\text{Encap}(\text{pk} = (\text{seed}_{\mathbf{A}}, \mathbf{b}))}$

1: $\mu \leftarrow \{0,1\}^{256}$
2: $(K, \text{seed}) \leftarrow G(\mu, \text{ID}(\text{pk}))$
3: ct $\leftarrow$ SMAUG-T.PKE.Enc(pk, μ; seed)
4: **return** ct, K
▷ ID : $\{0,1\}^* \rightarrow \{0,1\}^n$: extract fixed bits

* [CCS:DHKLS21] Faster Lattice-Based KEMs via a Generic Fujisaki-Okamoto Transform Using Prefix Hashing

Symmetric Primitives

- PRFs
  - SHAKE128 for public value extension
  - SHAKE256 for secret value extension
- G: SHAKE256
- H: SHA3-256 → SHAKE128
  - No need for (2nd) preimage resistance, as H(ct) is never used for the challenge ct.
  - Slightly better latency

$\underline{\mathsf{Encap}(\mathsf{pk} = (\mathsf{seed}_{\mathbf{A}}, \mathbf{b}))}$

1: $\mu \leftarrow \{0,1\}^{256}$
2: $(K, \mathsf{seed}) \leftarrow G(\mu, \mathsf{ID}(\mathsf{pk}))$
3: $\mathsf{ct} \leftarrow \mathsf{SMAUG\text{-}T.PKE.Enc}(\mathsf{pk}, \mu; \mathsf{seed})$
4: **return** ct, K

$\underline{\mathsf{Decap}(\mathsf{sk} = (\mathsf{sk}_{\mathsf{PKE}}, d, \mathsf{pk}), \mathsf{ct})}$:

1: $\mu' = \mathsf{SMAUG\text{-}T.PKE.Dec}(\mathsf{sk}', \mathsf{ct})$
2: $\mathsf{prefix} = \mathsf{ID}(\mathsf{pk})$
3: $(K', \mathsf{seed}') \leftarrow G(\mu', \mathsf{prefix})$
4: $\mathsf{ct}' = \mathsf{SMAUG\text{-}T.PKE.Enc}\,(\mathsf{pk}, \mu'; \mathsf{seed}')$
5: $(\hat{K}, \cdot\,) \leftarrow G(d, H(\mathsf{ct}))$
6: **if** $\mathsf{ct} \neq \mathsf{ct}'$ **then**
7: $\quad K' \leftarrow \hat{K}$
8: **return** K'

with appropriate domain separations.

Beyond IND-CCA2 and SUF-CMA

- Basically, it’s for single-user, single-ctxt.

- Multi-target security?
  - Already in consideration:
    - Multi-user security: use pk-related components
    - Multi-target security: use ct-related components

- Beyond Unforgeability from [S&P:CDFFJ21]
  - Exclusive Ownership, Message-boundness, Non Re-signability

* [S&P:CDFFJ21] BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures

Beyond IND-CCA2 and SUF-CMA: Example Analysis

- FrodoKEM-like approach for Single-user, Multi-ciphertext security
  - N ($\leq 2^{64}$) challenge ciphertexts: $(K_i^0, ct_i), K_i^1 \leftarrow \{0,1\}^{256}$
  - M hash trials for $\mu \in \{0,1\}^{256}$
    - $(K, seed) \leftarrow G(\mu, ID(pk))$ & $ct \leftarrow PKE.Enc(pk, \mu, seed)$
  - $ct = ct_i \rightarrow \mu = \mu_i \rightarrow K = K_i^0$ and reveal b from K_i^b

- Bit-Security:

$$\log_2\left(\frac{\text{Time}}{\text{Succ. Prob.}}\right) \geq \log_2\left(\frac{M}{MN/2^{256}}\right) = 256 - \log_2 N\,.$$

Beyond IND-CCA2 and SUF-CMA: Example Analysis

- FrodoKEM-like approach for Single-user, Multi-ciphertext security
  - N ($\leq 2^{64}$) challenge ciphertexts: $(K_i^0, ct_i), K_i^1 \leftarrow \{0,1\}^{256}$
  - M hash trials for $(\mu, salt) \in \{0,1\}^{256+\ell_{salt}}$
    - $(K, seed) \leftarrow G(\mu, salt, ID(pk))$ & $ct \leftarrow PKE.Enc(pk, \mu, seed)$
  - $ct = ct_i \rightarrow \mu = \mu_i \rightarrow K = K_i^0$ and reveal b from K_i^b

- Bit-Security:

$$\log_2\left(\frac{\text{Time}}{\text{Succ. Prob.}}\right) \geq \log_2\left(\frac{M}{MN/2^{256+\ell_{salt}}}\right) = 256 - \log_2 N + \ell_{salt}.$$

  - Assuming $N \leq 2^{64}$, we can add a 64-bit salt when hashing, for level-5 parameters
  - But, this is a very limited situation & similarly applies to, e.g., IND-security of AES.

- Under Consideration
  - Compatibility w/ CMVP & Easily verifiable Test Vectors
  - Possible modifications for better efficiency
    - Techniques that may degrade security (in any sense) may not be applied!
  - HAETAE's precomputation variant (hyperball sampling)
    - Similar to [FIPS 186-5] pre-computed k for ECDSA
    - But not easy to formalize

- Currently NOT Under Consideration
  - SMAUG-T: Separated hash function for implicit-rejection key (J in ML-KEM)
  - HAETAE : Pre-Hash variant (HashML-DSA)

Thank You!

The slides have been slightly modified with the feedbacks from the workshop.

HEAAN CRYPTO LAB

CRYPTO LAB
주식회사 크립토랩