Lightweight Lattice-based Single-Party Public-Key Authenticated Key Exchange

Alex Aïdan[1][0009-0008-8383-7055], Sébastien Canard[2][0009-0006-9588-7418], Emmanuel Fouotsa[3][0000-0002-7729-0419], and Nyiang Melchisedech Mbeng[3][0009-0001-3037-0649]

[1] Université Paris-Saclay, CEA, List, Palaiseau, France
first.last@cea.fr
[2] LTCI, Télécom Paris, Institut Polytechnique de Paris, France
first.last@telecom-paris.fr
[3] Centre for Cybersecurity and Mathematical Cryptology, The University of Bamenda, P.O.Box 39, Bambili, Cameroon
first.last@uniba.cm

Abstract. Authenticated Key Exchange (AKE) is a cornerstone of secure communication, especially in resource-constrained IoT environments where lightweight and post-quantum security are paramount. While lattice-based cryptography offers promising solutions, existing post-quantum AKE protocols often prioritize strong security notions, such as the use of an IND-CCA encryption scheme, incurring overheads incompatible with IoT devices. This raises a critical question: Can one-way security (OW), a weaker but potentially more efficient notion, suffice for secure AKE in the post-quantum era? We address this challenge by revisiting the ALIKE framework (ISO/IEC 29192-4), which achieves OW-CCA-based AKE using deterministic RSA. However, RSA's quantum vulnerability and the lack of lattice-based OW-CCA schemes hinder its applicability today. Our work bridges this gap through three key contributions. First, we prove that the Hash-Before-Encrypt (HBE) paradigm generically transforms deterministic OW-CPA schemes into OW-CCA-secure ones. We additionally present the Fujisaki–Okamoto transform and its security proof construction, providing a reference for understanding the efficiency advantages of the proposed HBE-based approach. Second, we modify Bai et al.'s efficient lattice-based OW-CPA scheme to a deterministic variant and rigorously prove its security. Third, we generalize the SPAKE framework to support our OW-CCA construction, enabling post-quantum AKE with minimal assumptions, implement and benchmark the resulting protocol, demonstrating state-of-the-art efficiency for lightweight, quantum-resistant AKE. By relaxing security requirements from IND-CCA to OW-CCA while preserving adaptive security we achieve a practical balance between robustness and performance, paving the way for deployable solutions in constrained environments like IoT and connected vehicles.

1 Introduction

Authenticated key exchange (AKE) stands as a cornerstone of modern cryptography, enabling secure communication between two parties over insecure channels by ensuring both confidentiality and mutual authentication. Its importance is further amplified in the era of the Internet of Things (IoT) and connected devices, where vehicles, sensors, and embedded systems routinely exchange sensitive data. In these resource-constrained environments such as smart cities, industrial IoT, and autonomous vehicles the deployment of robust yet efficient cryptographic mechanisms is critical. Many IoT devices and connected systems operate with limited computational power, memory, and energy, making the lightweight aspect of cryptographic protocols not just desirable, but essential. Traditional AKE schemes, while secure, often impose prohibitive overheads on such devices, necessitating the design of protocols that balance security with practical feasibility. In this paper, we focus on a special case of AKE, called Single-party Public-key Authenticated Key Exchange (SPAKE), which is a two-party cryptographic protocol played between a Prover and a Verifier. In this case, the Verifier authenticates the Prover using public key cryptography (hence relative to a certification authority) and the Prover and the Verifier establish a session key later used for secure messaging [17]. We remark that the Prover does not authenticate the Verifier, implying that any attacker can fake a Verifier and establish a session key with the Prover. Hence, the security model is less strong than traditional AKE.

In IoT and connected vehicle environments, such a single-party authentication model offers a compelling balance between security and efficiency. Unlike traditional mutual AKE, this approach eliminates the need for the Prover to verify the Verifier, significantly reducing computational overhead, latency, and energy consumption. This is critical for battery-powered devices or real-time applications like contactless access control, toll systems, or vehicle-to-infrastructure (V2I) communication, where the Prover (e.g., an RFID tag or onboard unit) operates under strict resource constraints. By leveraging lightweight public-key cryptography, SPAKE-like protocols ensure provable security while avoiding the complexity of symmetric key management or bidirectional authentication. The model's simplicity also enables low-cost, SAM-less readers, making it scalable for large-scale deployments such as smart cities or fleet management, where fast transaction times and minimal bandwidth usage are essential.

However, the advent of quantum computing poses an existential threat to classical cryptographic primitives, rendering widely used algorithms like RSA and ECC vulnerable to Shor's algorithm. This looming quantum threat has spurred the development of post-quantum cryptographic solutions, which rely on mathematical problems believed to be resistant to quantum attacks. Among these, lattice-based, code-based, hash-based, and multivariate approaches have emerged as leading candidates. However, lattice-based cryptography has gained particular prominence due to its versatility, efficiency, and strong security guarantees. It supports a wide range of cryptographic primitives, including AKE and their variants, and offers a compelling combination of theoretical robustness and

practical performance. Hence, we focus on lattice-based constructions in the sequel. As the cryptographic community transitions toward a post-quantum future, the challenge lies in designing AKE-like protocols that are not only quantum-resistant but also lightweight enough to be deployed on constrained devices. This paper addresses this dual challenge, proposing a Single-party Public-key AKE scheme that meets both post-quantum security requirements and the stringent resource limitations of IoT and connected systems.

1.1 Related Work on lattice-based AKE

A natural approach to constructing an AKE protocol is to leverage public-key encryption (PKE) schemes, where one party encrypts a shared secret under the other's public key, and authentication is achieved through digital signatures or certificates. In this paradigm, the security of the key exchange relies on the underlying PKE scheme's security properties. Classical security notions for PKE include one-wayness (OW) and indistinguishability (IND) under two different kinds of attacks: chosen-plaintext attacks (CPA), and the stronger chosen-ciphertext attacks (CCA). It is well-established that IND-CPA security is enough for passively secure AKE, while CCA security is necessary to resist adaptive adversaries, who may exploit decryption oracles or side-channel information to break weaker schemes (see for example [37,24] in the lattice seting). A foundational result in this context is the Fujisaki-Okamoto transform [25], which generically converts an IND-CPA scheme into an IND-CCA one, albeit at a cost in efficiency.

In [23], Fujioka et al. have proposed a generic way to go from a OW-CCA secure Key Encapsulation Message (KEM) to reach a so-called CK^+ secure AKE. Such a property permits to manage weak perfect forward secrecy, resilience to key compromise impersonation, and resilience to maximal exposure attacks. But their result if far from being "lightweight", executing several times the key generation, the encapsulation and the decaspulation algorithms by both parties.

In the post-quantum setting, Kyber (now standardized as ML-KEM) is a leading lattice-based key encapsulation mechanism (KEM) achieving IND-CCA security. However, its efficiency while acceptable for many applications remains a bottleneck for resource-constrained IoT devices. Recent works have explored optimizing this trade-off. For instance, Düzlü et al. [22] propose a modified Kyber variant that achieves IND-CPA security and demonstrates that, when combined with a clever adaptation of the Fujisaki-Okamoto (FO) transform, the resulting scheme can both lead to a secure AKE scheme but also improve side-channel resistance without significant overhead. However, the efficiency gains are modest, as the transform itself introduces computational and communication costs.

An alternative direction is pursued by Kundu et al. [30], who introduce a new lattice-based PKE scheme natively achieving IND-CCA security. While their construction offers competitive performance, it still adheres to the IND-CCA paradigm, leaving open the question of whether weaker security notions could suffice for AKE in practice.

More recently, the PQuAKE protocol [8], proposed as an IETF Internet-Draft, introduces a lightweight authenticated key exchange mechanism based on ML-KEM and ML-DSA, specifically targeting constrained environments. They are only considering IND-CCA secure KEM, and they additionally need a digital signature. Although this is still a work in progress, with which it is difficult to draw comparisons at this stage, this draft reflects current trends toward reducing communication and computational overhead in post-quantum AKE design.

Following [22], one idea would be to further reduce the requirements on the encryption system used, in order to be able to use instantiations that could be considered as "lightweight". This raises a critical research question:

Can the security requirements for the underlying PKE be relaxed to One-Wayness (OW) while still ensuring the overall security of the AKE protocol?

If so, lattice-based OW candidates potentially more efficient than their IND-CCA counterparts could enable lighter AKE protocols. Exploring this possibility could yield significant efficiency improvements, particularly for IoT deployments where every cycle and byte counts. Focusing on single-party public key AKE, this paper investigates whether OW security is sufficient for this case in the post-quantum setting and identifies promising lattice-based candidates that balance minimal security assumptions with practical performance.

1.2 Our Approach

A pioneering work by Coron et al. [17] demonstrates that one-way security under chosen-ciphertext attacks (OW-CCA) can indeed suffice for secure SPAKE, even against adaptive adversaries. Their construction, ALIKE (later standardized as ISO/IEC 29192-4), achieves this by leveraging the deterministic nature of RSA-OAEP. However, RSA's vulnerability to quantum attacks renders this approach obsolete in the post-quantum era. This begs the question:

Can their framework be generalized to a SPAKE-like construction compatible with any OW-CCA scheme, and can lattice-based primitives fill this role?

Positively answering those questions could provide an interesting trade-off between security and efficiency for IoT devices. It would allow the use of OW-CCA schemes instead of their more secure but more costly IND-CCA counterparts. To date, no lattice-based public-key encryption scheme is known to natively achieve both one-way (OW) security and chosen-ciphertext (CCA) security.

Achieving OW-CCA from IND-CPA. A first strategy to reach OW-CCA, is to start from an IND-CPA-secure KEM or PKE, as it has been done in [23] for their CK^+ AKE. Using the result of [27], which shows that IND-CPA security with a sufficiently large message space implies OW-CPA security in the quantum random oracle model, the Fujisaki–Okamoto (FO) transform [25] can then be

applied to obtain OW-CCA security. This could then be applied to any IND-CPA-secure KEM, such as [34] and [39], as proposed in [23], but also to Kyber [9], Lizard-based constructions [13], SABER [20], Peikert's proposals [37], etc.

We consider this way to proceed as not optimal for two reasons: (i) starting from IND security when only one-wayness is required introduces stronger assumptions than necessary and (ii) the Fujisaki–Okamoto transform incurs additional computational and communication overhead, making it less suitable for lightweight applications. Hence, in this paper, we consider another alternative. Since the FO transform represents the standard approach for achieving chosen-ciphertext security from weaker notions, we present its complete construction and security proof in the full version for completeness.

Achieving OW-CCA from OW-CPA. The second natural way to proceed is to start from a OW-CPA-secure scheme. Several recent lattice-based candidates achieve OW-CPA security, including works by Bai et al. [2] and Cong et al. [15]. Again, applying the Fujisaki–Okamoto transform would produce an OW-CCA-secure construction, but at the cost of additional hashing, ciphertext validation, and re-encryption operations, which are undesirable in resource-constrained environments.

Instead, we extend the ideas introduced in ALIKE [17] to probabilistic encryption, yielding a generic Hash-Before-Encrypt (HBE) paradigm for transforming deterministic OW-CPA schemes into OW-CCA-secure ones. This approach consists in hashing before encryption. More precisely, from a message m and a hash function H, we compute $h = H(m)$ and then use the OW-CPA encryption scheme to encrypt $m\|h$. This enables a ALIKE-inspired SPAKE protocol that retains both post-quantum security and efficiency, addressing the gap left by prior work.

Our contributions. Given all that, our contributions are threefold.

- Generic transformation for OW-CCA security: We formally prove that the Hash-Before-Encrypt (HBE) paradigm enables a generic transformation of any deterministic OW-CPA-secure encryption scheme into an OW-CCA-secure variant, preserving efficiency while strengthening security guarantees. While generic transformations have been extensively studied for indistinguishability-based security notions [25,35,19], to the best of our knowledge this is the first generic treatment in the one-way security setting.
- Deterministic lattice-based OW-CPA scheme: We adapt the probabilistic OW-CPA scheme of Bai et al. [2] to a deterministic setting and provide a rigorous security proof for the resulting construction, bridging a critical gap in lattice-based primitives.
- Efficiency benchmarking: Building on these results, we propose PQ-ALIKE, a post-quantum version of ALIKE [17]. Building on these results, we propose PQ-ALIKE, a post-quantum instantiation of ALIKE [17]. We implement and evaluate PQ-ALIKE, demonstrating state-of-the-art efficiency among lightweight post-quantum SPAKE protocols and validating its practicality for resource-constrained environments.

The paper is organised as follows. Section 2 introduces the public-key encryption background required throughout the paper. Section 3 presents generic transformations from deterministic OW-CPA to OW-CCA security, including the Hash-Before-Encrypt paradigm and the complete Fujisaki–Okamoto transform. Section 4 presents our lattice-based deterministic OW-CCA encryption scheme and its security analysis. Section 5 recalls Single-party Public-key Authenticated Key Exchange and presents the generic (PQ-)ALIKE construction. Finally, Section 6 describes our implementation of PQ-ALIKE and reports the corresponding performance evaluation.

2 One-Way Secure Encryption

In this section, we give the security definition for encryption schemes, and one-way security. We then provide some new results about the security, and several ways to go from CPA (Chosen Plaintext Attack) security to CCA (Chosen Ciphertext Attack) security.

2.1 Encryption Schemes

Public-key encryption schemes form the foundation for secure communication in authenticated key exchange protocols. An encryption scheme can be either probabilistic or deterministic, depending on how randomness is managed during encryption. We introduce both notions below.

Definition 1 (Public-Key Encryption Scheme). *A public-key encryption scheme Π is defined by the three following procedures, where $\mathcal{M}$ is the set of accepted messages.*

- *$\mathcal{K}$ is the key generation. It takes as input the security parameter 1^λ and outputs a private key sk and the corresponding public key pk. We consider that the security parameter is now included in the public key.*
- *$\mathcal{E}$ is the encryption algorithm. It takes as input the public key pk and a message $m \in \mathcal{M}$, and outputs a ciphertext c. The encryption algorithm may be probabilistic, taking additional randomness $r \in \mathcal{R}$ (denoted $\mathcal{E}(pk, m; r)$ in the sequel), or deterministic (denoted $\mathcal{E}(pk, m)$ in the sequel).*
- *$\mathcal{D}$ is the decryption algorithm. It takes as input a ciphertext c and a secret key sk and outputs a message m.*

Remark. A well-known technique to obtain deterministic encryption from probabilistic schemes is derandomization. Instead of sampling fresh randomness $r \in \mathcal{R}$ for each encryption, the randomness is derived deterministically from the message. Throughout this paper, we employ derandomization to convert our base probabilistic encryption scheme into a deterministic one that achieves OW-CCA security, as detailed in Section 3.

2.2 One-Way Security

We present the one-way security notions for public-key encryption schemes. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a public-key encryption scheme defined by $\mathcal{K}, \mathcal{E}$, and $\mathcal{D}$. We now define the two usual notions of one-wayness that differ in the adversarial capabilities.

One-Way Security under Chosen Plaintext Attack (OW-CPA). We first consider an adversarial model in which the adversary does not have access to an encryption oracle.

Definition 2 (OW-CPA Security). *A public-key encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is (t, ε)-OW-CPA secure if for any adversary $\mathcal{A}$ running in time at most t, the probability of recovering the message m from a challenge ciphertext $c = \mathcal{E}nc(\mathsf{pk}, m)$ is less than or equal to ε. Formally:*

$$\Pr\left[(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^\lambda); m \leftarrow \mathcal{M}; c \leftarrow \mathcal{E}(\mathsf{pk}, m); m' \leftarrow \mathcal{A}(\mathsf{pk}, c) : m' = m\right] \leq \varepsilon$$

where the adversary $\mathcal{A}$ only has access to the public key pk and the challenge ciphertext c.

In the public-key setting, having access to the public key *pk* means the adversary has the ability to encrypt any message of its choice. Therefore, OW-CPA security captures the scenario where the adversary can compute encryptions of arbitrary plaintexts but must still find it infeasible to decrypt a challenge ciphertext.

One-Way Security under Chosen Ciphertext Attack (OW-CCA). Secondly, we consider the strongest one-way security notion where the adversary has access to a decryption oracle, except for the challenge ciphertext itself.

Definition 3 (OW-CCA Security). *A public-key encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is (t, ε)-OW-CCA secure if for any adversary $\mathcal{A}$ running in time at most t and having access to a decryption oracle (except for the challenge ciphertext), the probability of recovering the message m from a challenge ciphertext c is less than or equal to ε. Formally:*

$$\Pr\left[(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^\lambda); m \leftarrow \mathcal{M}; c \leftarrow \mathcal{E}(\mathsf{pk}, m); m' \leftarrow \mathcal{A}^{\mathcal{D}(\cdot, \mathsf{sk})}(\mathsf{pk}, c) : m' = m\right] \leq \varepsilon$$

where $\mathcal{A}^{\mathcal{D}ec(\cdot, \mathsf{sk})}$ denotes that $\mathcal{A}$ has access to a decryption oracle that it can query on any ciphertext $c' \neq c$ to get the message $m' = \mathcal{D}(c', \mathsf{sk})$.

The OW-CCA notion models active adversaries who can manipulate ciphertexts and observe the results of decryption, making it a much stronger security requirement than OW-CPA. An encryption scheme that achieves OW-CCA security provides strong guarantees even against sophisticated adaptive attacks.

3 From OW-CPA to OW-CCA Security

OW-CCA security is significantly stronger than OW-CPA security, as the adversary is additionally granted adaptive access to a decryption oracle. Two generic approaches naturally arise for upgrading an OW-CPA-secure encryption scheme to OW-CCA security in the random oracle model: the Fujisaki–Okamoto (FO) transformation and the Hash-Before-Encrypt (HBE) transformation. In this section, we present both approaches. The FO transform serves as the canonical generic construction for achieving chosen-ciphertext security, whereas the HBE transform provides a more lightweight alternative for deterministic encryption schemes.

The remainder of this section first presents the Fujisaki–Okamoto transform as the canonical generic construction before introducing the Hash-Before-Encrypt paradigm, which forms the basis of our proposed scheme.

3.1 Generic OW-CPA to OW-CCA Transformation via the Fujisaki–Okamoto Transform

The Fujisaki-Okamoto (FO) transformation [25,26] is the most widely used generic method to achieve strong CCA security from weaker CPA security. Originally designed to convert schemes with indistinguishability properties (IND-CPA) into schemes with chosen-ciphertext security (IND-CCA) in the random oracle model, the modular analysis by Hofheinz et al. [26] demonstrated that the FO transformation can be decomposed into several sub-transformations, each addressing specific security aspects.

While the original FO literature focuses on indistinguishability-based security, the transformation can also be instantiated in the one-way security setting. For completeness, we present the corresponding construction and proof, which serve as a useful point of comparison with our Hash-Before-Encrypt transformation.

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a public-key encryption scheme. The FO-transformed scheme $\Pi_{FO} = (\mathcal{K}_{FO}, \mathcal{E}_{FO}, \mathcal{D}_{FO})$ is constructed as described in Figure 1.

Properties of the FO Transform. The FO transformation incorporates three key techniques:

1. **Derandomization**: By deriving encryption randomness deterministically from the message using $r = G(m, pk)$, the transformation ensures that each message has a unique valid ciphertext under a given public key, preventing adversaries from creating multiple valid ciphertexts for the same message.
2. **Re-encryption verification**: During decryption, the scheme re-derives the randomness $r' = G(m', pk)$ and re-encrypts to verify that $\mathcal{E}(pk, m'; r') = c$. This ensures the ciphertext was generated honestly and prevents submission of malformed ciphertexts to the decryption oracle.
3. **Hash-based integrity**: The additional hash $h = H(c, m)$ provides message-ciphertext binding. Implicit rejection (returning pseudorandom values on failure) prevents information leakage through malformed ciphertext queries.

- **Key Generation** $\mathcal{K}_{FO}(1^\lambda)$:
  1. Run $(pk, sk) \leftarrow \mathcal{K}(1^\lambda)$
  2. Return (pk, sk)
- **Encryption** $\mathcal{E}_{FO}(pk, m)$:
  1. Derive encryption randomness: $r \leftarrow G(m, pk)$ where G is a hash function modeled as a random oracle
  2. Compute ciphertext: $c \leftarrow \mathcal{E}(pk, m; r)$ using the derived randomness
  3. Compute integrity tag: $h \leftarrow H(c, m)$ where H is another hash function modeled as a random oracle
  4. Return (c, h)
- **Decryption** $\mathcal{D}_{FO}(sk, (c, h))$:
  1. Decrypt: $m' \leftarrow \mathcal{D}(sk, c)$
  2. If $m' = \perp$, return a pseudorandom value derived from c (implicit rejection)
  3. Re-derive randomness: $r' \leftarrow G(m', pk)$
  4. Re-encrypt: $c' \leftarrow \mathcal{E}(pk, m'; r')$
  5. If $c' \neq c$, return $\perp$ (or pseudorandom value)
  6. Re-compute hash: $h' \leftarrow H(c, m')$
  7. If $h' \neq h$, return $\perp$ (or pseudorandom value)
  8. If all checks pass, return m'

Fig. 1. Fujisaki-Okamoto transform

These properties together provide strong protection against chosen-ciphertext attacks, upgrading the base scheme's security from OW-CPA to OW-CCA.

3.2 Security Proof

We now formally prove that the Fujisaki-Okamoto transformation can be used to convert a OW-CPA secure scheme into an OW-CCA secure scheme.

Theorem 1 (FO Transform: OW-CPA to OW-CCA). *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a (t', ε')-OW-CPA secure public-key encryption scheme. Then the FO-transformed scheme $\Pi_{FO} = (\mathcal{K}_{FO}, \mathcal{E}_{FO}, \mathcal{D}_{FO})$ described above is (t, ε)-OW-CCA secure in the random oracle model, where*

$$\varepsilon \leq q_{\mathrm{RO}} \cdot \varepsilon' + (q_D + 1) \cdot 2^{-\lambda}, \tag{1}$$

and

$$t \approx t' - O((q_{\mathrm{RO}} + q_D) \cdot T_{\mathrm{enc}}), \tag{2}$$

where q_{RO} is the total number of queries made by the adversary to the random oracles G and H, q_D is the number of decryption oracle queries, T_{enc} is the time for one encryption operation, and λ is the security parameter.

Proof. We use a single random oracle R to instantiate the two random oracles H and G used in the FO transform. Domain separation allows the two functionalities to remain distinct. In particular, $H(\mathsf{pk})$ and $G(h_{\mathsf{pk}}, m)$ are evaluated through the same random oracle R with distinct domain identifiers. Thus, the two types of queries are counted jointly, and q_{RO} denotes the total number of queries to R.

with the corresponding output lengths required by the FO construction. Thus, all queries to G and H are counted jointly as queries to the single random oracle R, denoted by q_{RO}. We show that if there exists an adversary $\mathcal{A}$ that breaks the OW-CCA security of Π_{FO} with probability ε in time t, we can construct an adversary $\mathcal{B}$ that breaks the OW-CPA security of Π with probability related to ε.

Setup. Suppose $\mathcal{B}$ is given a public key pk and a challenge ciphertext $c^* = \mathcal{E}(pk, m^*; r^*)$ where m^* is uniformly random and r^* is the randomness used in the OW-CPA game for the base scheme Π. The goal of $\mathcal{B}$ is to recover m^*.

$\mathcal{B}$ will use $\mathcal{A}$ to help recover m^* by simulating the OW-CCA game for Π_{FO}. $\mathcal{B}$ proceeds as follows:

Challenge Construction. $\mathcal{B}$ needs to create a valid FO challenge (c^*, h^*) for $\mathcal{A}$:

- $\mathcal{B}$ sets the ciphertext component as c^* (from the OW-CPA challenge)
- $\mathcal{B}$ chooses a random hash value $h^* \leftarrow \{0,1\}^\lambda$
- $\mathcal{B}$ gives $(pk, (c^*, h^*))$ to $\mathcal{A}$ as the OW-CCA challenge

Oracle Simulation. $\mathcal{B}$ must simulate the random oracles G and H, as well as the decryption oracle $\mathcal{D}_{FO}$, for $\mathcal{A}$:

Random Oracle R Simulation:

- $\mathcal{B}$ maintains a single table T_R initially empty.
- For a query corresponding to $H(\mathsf{pk})$, $\mathcal{B}$ checks whether the corresponding query has already been recorded in T_R. If so, it returns the previously stored value. Otherwise, it samples a fresh value h_{pk} uniformly at random, stores the query and its answer in T_R, and returns h_{pk}.
- For a query corresponding to $G(h_{\mathsf{pk}}, m)$, $\mathcal{B}$ checks whether the corresponding query has already been recorded in T_R. If so, it returns the previously stored value. Otherwise, it samples a fresh value r uniformly at random, stores the query and its answer in T_R, and returns r.

Decryption Oracle $\mathcal{D}_{FO}$ Simulation:

- Search T_R for the corresponding query to $G(h_{\mathsf{pk}}, m)$.
- If no such entry exists, return $\perp$.
- If such an entry exists, retrieve the corresponding value r from T_R and re-encrypt: $c' \leftarrow \mathcal{E}(\mathsf{pk}, m; r)$.
- If $c' \neq c$, return $\perp$.
- If $c' = c$, return m.

Note that $\mathcal{B}$ can perform all these operations without knowing m^* or having access to the decryption oracle for Π, because:

- $\mathcal{B}$ never needs to decrypt c^* (it returns $\perp$ for such queries)
- For other ciphertexts, $\mathcal{B}$ can verify validity by re-encryption (which only requires pk)

Extracting m^*. When $\mathcal{A}$ outputs its guess m' for the challenge message:

– $\mathcal{B}$ outputs m' as its guess for m^* in the OW-CPA game

Analysis. We now analyze the probability that $\mathcal{B}$ successfully recovers m^*.

Event E_1: $\mathcal{A}$ succeeds without querying $H(c^*, m^*)$.

If $\mathcal{A}$ outputs the correct $m' = m^*$ without ever querying $H(c^*, m^*)$, then $\mathcal{A}$ must have guessed h^* correctly (since a valid FO ciphertext must satisfy $h = H(c, m)$). Since h^* was chosen uniformly at random by $\mathcal{B}$ and independently of $\mathcal{A}$'s view, the probability of this event is:

$$\Pr[E_1] \leq 2^{-\lambda} \tag{3}$$

Event E_2: $\mathcal{A}$ queries $H(c^*, m^*)$ at some point.

If $\mathcal{A}$ makes the query $H(c^*, m^*)$, then $\mathcal{B}$ will record (c^*, m^*, h^*) in table T_H. At the end of the game:

– If $\mathcal{A}$ outputs $m' = m^*$, then $\mathcal{B}$ succeeds
– $\mathcal{B}$ can also extract m^* from T_H by checking which entries have $c = c^*$

The probability that $\mathcal{A}$ queries $H(c^*, m^*)$ can be bounded by considering two sub-cases:

Case 2a: $\mathcal{A}$ queries $G(m^*, pk)$ before querying $H(c^*, m^*)$.

Since all queries to G and H are now counted jointly as queries to the single random oracle R, and the adversary makes at most q_{RO} such queries in total, the probability of obtaining the challenge message through these queries is bounded by $q_{\mathrm{RO}} \cdot \varepsilon'$.

Case 2b: $\mathcal{A}$ queries $H(c^*, m)$ for some m without first querying $G(m, pk)$.

For $\mathcal{A}$ to query $H(c^*, m)$ meaningfully, it must believe that m might be the message corresponding to c^*. If $\mathcal{A}$ has not queried $G(m, pk)$, then $\mathcal{A}$ has no information about whether $\mathcal{E}(pk, m; G(m, pk)) = c^*$. By the OW-CPA security of Π, guessing m correctly has probability at most ε' per guess. Since these G- and H-related queries are counted jointly as queries to R, the corresponding probability is bounded by $q_{\mathrm{RO}} \cdot \varepsilon'$..

Decryption Oracle Queries.

The decryption oracle queries do not help $\mathcal{A}$ learn m^* because:

– $\mathcal{B}$ rejects any query with $c = c^*$ and $h \neq h^*$
– $\mathcal{B}$ cannot decrypt (c^*, h^*) without knowing m^*, but this is prevented by returning $\perp$
– For other ciphertexts, $\mathcal{B}$ simulates decryption correctly using re-encryption

The simulation is perfect except with probability at most $q_D \cdot 2^{-\lambda}$ (the probability that $\mathcal{A}$ submits a ciphertext (c, h) where h happens to be the correct hash for some message m without having queried $H(c, m)$).

The probability that $\mathcal{B}$ recovers m^* is at least the probability that $\mathcal{A}$ succeeds minus the probabilities of the above failure events:

$$\Pr[\mathcal{B} \text{ succeeds}] \geq \Pr[\mathcal{A} \text{ succeeds}] - \Pr[E_1] - (\text{simulation failures}) \tag{4}$$

$$\geq \varepsilon - 2^{-\lambda} - q_D \cdot 2^{-\lambda} \tag{5}$$

Conversely, if $\mathcal{B}$ succeeds with probability ε' in the OW-CPA game, then:

$$\varepsilon \leq q_{\mathrm{RO}} \cdot \varepsilon' + (q_D + 1) \cdot 2^{-\lambda}. \tag{6}$$

This completes the proof.

Although the FO transform provides a generic approach for achieving OW-CCA security, it requires ciphertext re-encryption during decryption, increasing the computational cost. This motivates the search for lighter generic transformations. The Hash-Before-Encrypt paradigm presented next achieves the same security notion while avoiding the re-encryption step, making it particularly attractive for lightweight post-quantum applications.

3.3 Hash-Before-Encryption Transform

The Hash-Before-Encrypt (HBE) transformation provides a lightweight generic approach for obtaining OW-CCA security from deterministic OW-CPA-secure encryption schemes. The HBE idea has previously been studied in the context of authenticated encryption [38] and for deterministic RSA-based constructions such as ALIKE [17]. We generalize this paradigm to arbitrary deterministic OW-CPA-secure public-key encryption schemes and provide a complete security proof.

Compared with the FO transform, HBE avoids the ciphertext re-encryption step during decryption and requires only a hash verification. Consequently, decryption requires fewer expensive public-key operations, making the approach particularly attractive for lattice-based encryption schemes.

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a public-key encryption scheme with derandomization (i.e., $\mathcal{E}(pk, m)$ computes $r = G(m, pk)$ and encrypts using derived randomness). The HBE-transformed scheme $\Pi_{HBE} = (\mathcal{K}_{HBE}, \mathcal{E}_{HBE}, \mathcal{D}_{HBE})$ is constructed as described in Figure 2.

Properties of the HBE Transform. The HBE transformation offers two main advantages over the FO transform.

1. It naturally supports deterministic encryption through derandomization using $r = G(m', pk)$.
2. During decryption it requires only a hash verification, whereas the FO transform additionally performs ciphertext re-encryption.

3.4 Security Proof

We now formally prove that the HBE transformation generically converts any deterministic OW-CPA-secure encryption scheme into an OW-CCA-secure scheme in the random oracle model.

- **Key Generation** $\mathcal{K}_{HBE}(1^\lambda)$:
  1. Run $(pk, sk) \leftarrow \mathcal{K}(1^\lambda)$
  2. Return (pk, sk)
- **Encryption** $\mathcal{E}_{HBE}(pk, m)$:
  1. Compute hash of message: $h \leftarrow H(m)$ where $H : \{0,1\}^* \to \{0,1\}^t$ is a hash function modeled as a random oracle
  2. Concatenate message with hash: $m' \leftarrow m \| h$
  3. Derive randomness: $r \leftarrow G(m', pk)$ where G is another hash function
  4. Encrypt: $c \leftarrow \mathcal{E}(pk, m'; r)$ using the derived randomness
  5. Return c
- **Decryption** $\mathcal{D}_{HBE}(sk, c)$:
  1. Decrypt $m^* \leftarrow \mathcal{D}(sk, c)$
  2. If $m^* = \perp$, return $\perp$ (or pseudorandom value derived from c
  3. Parse m^* as $m \| h$ (split into message and hash components)
  4. Re-compute hash $h' \leftarrow H(m)$
  5. If $h' \neq h$, return $\perp$ (or pseudorandom value)
  6. If verification succeeds, return m

Fig. 2. Hash-Before-Encryption transform

Theorem 2 (Generic OW-CPA to OW-CCA Transformation via HBE). *Let $\Pi = (KeyGen, Encrypt, Decrypt)$ be a (t', ε')-OW-CPA secure public-key encryption scheme. Then the hash-before-encrypting variant $\Pi_{HBE} = (KeyGen, Encrypt_{HBE}, Decrypt_{HBE})$ is (t, ε)-OW-CCA secure in the random oracle model, with*

$$Encrypt_{HBE}(pk, m) = Encrypt(pk, m \| H(m)) \tag{7}$$

$$Decrypt_{HBE}(sk, c) \text{ first decrypts to } m' \| h, \text{ then verifies } H(m') = h \tag{8}$$

$$\varepsilon = q_h \cdot \varepsilon' + q_c \cdot 2^{-t} \tag{9}$$

where q_h is the number of hash queries, q_c is the number of ciphertext queries, ε' is the base security bound, and t is a security parameter determining the length of the hash output.

Proof. Assume that there exists an adversary $\mathcal{A}$ that breaks the OW-CCA security of Π_{HBE} with probability at least ε in time t, making at most q_h hash queries and q_c decryption queries. We construct an adversary $\mathcal{B}$ that breaks the OW-CPA security of the base encryption scheme Π.

Setup. Adversary $\mathcal{B}$ receives a public key pk and a challenge ciphertext

$$c^* = \text{Encrypt}(pk, m^* \| h^*)$$

for an unknown message m^*, where $h^* = H(m^*)$. Adversary $\mathcal{B}$ gives (pk, c^*) to $\mathcal{A}$ as the challenge ciphertext in the OW-CCA game and simulates the random oracle and the decryption oracle.

Random Oracle Simulation. $\mathcal{B}$ maintains a table $\mathcal{T}$ of pairs (m, h). When $\mathcal{A}$ queries $H(m)$:

1. If (m, h) already appears in $\mathcal{T}$, return h.
2. Otherwise choose $h \xleftarrow{\$} \{0,1\}^t$, store (m, h) in $\mathcal{T}$ and return h.

Decryption Oracle Simulation. When $\mathcal{A}$ submits a ciphertext c:

1. If $c = c^*$, return $\perp$.
2. Otherwise, for each (m, h) in $\mathcal{T}$, compute $c' = \text{Encrypt}(pk, m\|h)$. If $c' = c$, return m.
3. If no match is found, return $\perp$.

The simulation may differ from the real decryption oracle only if $\mathcal{A}$ submits a valid ciphertext $c = \text{Encrypt}(pk, m\|H(m))$ without querying $H(m)$ (item 3. above). This requires guessing the random oracle value $H(m)$, which occurs with probability at most $q_c 2^{-t}$, which will be taken into account in the below analysis.

Analysis. We distinguish two cases.

Type 1 adversary. Suppose that $\mathcal{A}$ never queries $H(m^*)$. Then the value $h^* = H(m^*)$ remains completely random from $\mathcal{A}$'s point of view. For $\mathcal{A}$ to succeed it must produce a valid ciphertext containing the correct value h^*. The probability of guessing this value is at most 2^{-t} per decryption query, hence

$$\Pr[\text{Type 1 success}] \leq q_c \cdot 2^{-t}.$$

Type 2 adversary. Otherwise, $\mathcal{A}$ queries $H(m^*)$ at some point. Since the two cases are mutually exclusive,

$$\Pr[\text{Type 2 success}] \geq \varepsilon - q_c 2^{-t}.$$

If this occurs, the value m^* necessarily appears among the q_h hash queries. Adversary $\mathcal{B}$ chooses uniformly at random one of these q_h queries and outputs its argument as its guess for m^*. Hence

$$\Pr[\mathcal{B} \text{ succeeds} \mid \text{Type 2}] = \frac{1}{q_h}.$$

Therefore

$$\Pr[\mathcal{B} \text{ succeeds}] \geq \frac{\varepsilon - q_c 2^{-t}}{q_h}.$$

Conversely, if Π is (t', ε')-OW-CPA secure, we must have

$$\varepsilon \leq q_h \varepsilon' + q_c 2^{-t}.$$

This proves that Π_{HBE} is OW-CCA secure in the random oracle model. □

As confirmed by our implementation results, the HBE transformation incurs only a minimal decryption overhead, making it particularly well suited for lightweight authenticated key exchange protocols in resource-constrained environments.

4 Our Post-Quantum OW-CCA Encryption Scheme

In this section, we present our deterministic adaptation of the compact lattice-based encryption scheme of Bai et al. [2]. We first introduce the computational assumptions underlying the construction, then describe the deterministic OW-CPA scheme, and finally show how the generic Hash-Before-Encrypt transformation from Section 3 yields an OW-CCA-secure variant.

4.1 Assumptions

Setting. We work over the polynomial ring:

$$R_q = \mathbb{Z}_q[X]/(X^n + 1) \tag{10}$$

where n is a power of 2 (typically $n = 384, 512$, or 768) and q is a prime modulus. Elements of R_q are polynomials of degree at most $n-1$ with coefficients in $\mathbb{Z}_q$. For a positive integer d (the module rank), we work with the module:

$$R_q^d = \underbrace{R_q \times R_q \times \cdots \times R_q}_{d \text{ times}} \tag{11}$$

Module-NTRU. Module-NTRU improves upon classic NTRU by using module lattices (rather than ideal lattices) to mitigate potential attacks exploiting the algebraic structure, while preserving NTRU's advantages of small key sizes and fast operations [11]. It is based on the following probability distributions.

Definition 4 ($\mathsf{MNTRU}_{R_q,k,\chi_{\mathbf{F}},\chi_{\mathbf{g}}}$ Distribution). *Let R_q be a quotient polynomial ring, k be a positive integer, and $\chi_{\mathbf{F}}, \chi_{\mathbf{g}}$ be distributions defined over $R_q^{k\times k}$ and R_q^k, respectively. The $\mathsf{MNTRU}_{R_q,k,\chi_{\mathbf{F}},\chi_{\mathbf{g}}}$ distribution is defined as:*

$$\left\{\mathbf{h} = \mathbf{F}^{-1}\mathbf{g} \mid \mathbf{F} \leftarrow \chi_{\mathbf{F}},\ \mathbf{F} \textit{ invertible},\ \mathbf{g} \leftarrow \chi_{\mathbf{g}}\right\}.$$

This leads to the following definition for the Decisonal MNTRU problem.

Definition 5 (Decisional $\mathbf{MNTRU}_{R_q,k,\chi_{\mathbf{F}},\chi_{\mathbf{g}}}$ Problem). *Let $R_q, k, \chi_{\mathbf{F}}, \chi_{\mathbf{g}}$ be defined as above. The Decisional $\mathsf{MNTRU}_{R_q,k,\chi_{\mathbf{F}},\chi_{\mathbf{g}}}$ problem asks to distinguish between:*

- *Arbitrarily many independent samples from the $\mathsf{MNTRU}_{R_q,k,\chi_{\mathbf{F}},\chi_{\mathbf{g}}}$ distribution,*
- *The same number of independent uniform samples on R_q^k.*

The best known attacks on Module-NTRU are lattice reduction algorithms (BKZ, sieving) that attempt to find short vectors in $\Lambda_{\mathbf{h}}$. While there is no known polynomial-time reduction from worst-case SVP to average-case NTRU, the problem has resisted cryptanalysis for long, providing strong empirical confidence in its hardness. Hence, the $\mathsf{MNTRU}_{R_q,k,\chi_{\mathbf{F}},\chi_{\mathbf{g}}}$ assumption states that no probabilistic polynomial-time (PPT) algorithm can solve the decisional MNTRU problem with non-negligible advantage over a random guess.

Module-LWE. The Module-LWE (MLWE) problem introduced in [31] can be considered as a balanced solution that interpolates the parameters used in-between Ring-LWE and LWE. It is based on the following distribution.

Definition 6 ($\mathsf{MLWE}_{R_q,k,\mathbf{s},\chi_e}$ Distribution). *Let R_q be a quotient polynomial ring, k be a positive integer, $\mathbf{s} \in R_q^k$, and χ_e be a distribution on R_q. The $\mathsf{MLWE}_{R_q,k,\mathbf{s},\chi_e}$ distribution is defined as:*

$$\left\{(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e) \mid \mathbf{a} \leftarrow R_q^k,\ e \leftarrow \chi_e\right\}.$$

And the corresponding search version of the MLWE problem is then defined as follows.

Definition 7 (Search $\mathsf{MLWE}_{R_q,k,\chi_s,\chi_e}$ Problem). *Let R_q, k be defined as above, and χ_s, χ_e be two distributions on R_q. The Search $\mathsf{MLWE}_{R_q,k,\chi_s,\chi_e}$ problem asks to recover the secret $\mathbf{s}$ given arbitrarily many samples from the $\mathsf{MLWE}_{R_q,k,\mathbf{s},\chi_e}$ distribution, where $\mathbf{s} \leftarrow \chi_s^k$.*

The MLWE assumption states that no PPT algorithm can solve the search $\mathsf{MLWE}_{R_q,k,\chi_s,\chi_e}$ problem with non-negligible advantage.

4.2 Our Deterministic OW-CPA Scheme

We present our compact encryption scheme based on the Module-NTRU problem, based on an initial scheme proposed by Bai et al. in [2]. We first describe how we modify the scheme of Bai et al. [2] to obtain a deterministic OW-CPA-secure construction under the Module-NTRU assumption. We then show that applying the Hash-Before-Encrypt transformation of Section 3 directly yields an OW-CCA-secure scheme.

Scheme description. We first modify the proposal by Bai et al. [2] so that it becomes deterministic, using differently a trick given in [5] and adapted to the case of lattices in [9]. Let $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ be defined as above. Let k be a positive integer where $k + 1$ denotes the module rank, q be a prime denoting the ciphertext modulus and p be a prime denoting the plaintext modulus with $p < q$. Let χ_f, χ_g, χ_r be distributions over R_q. Finally, let H and G be two hash functions, seen as random oracles. Our resulting complete scheme is given in Figure 3 and provide below some details.

At first, the key generation samples an invertible matrix $\{f_{ij}\}_{i,j\in[k]} \leftarrow \chi_f^{k\times k}$ with $\mathbf{F}$ having the following structure to ensure invertibility:

$$\mathbf{F} = \begin{bmatrix} pf_{11}+1 & pf_{12} & pf_{13} & \cdots & pf_{1k} \\ f_{21} & pf_{22}+1 & pf_{23} & \cdots & pf_{2k} \\ f_{31} & f_{32} & pf_{33}+1 & \cdots & pf_{3k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_{k1} & f_{k2} & f_{k3} & \cdots & pf_{kk}+1 \end{bmatrix} \tag{12}$$

The diagonal entries $pf_{ii} + 1$ ensure that $\mathbf{F}$ is invertible modulo q with high probability.

Then, the key difference from the original probabilistic Module-NTRU encryption scheme of Bai et al. [2] is the deterministic derivation of the encryption randomness. While the initial scheme samples $\mathbf{r} \leftarrow \chi_r^k$ freshly for each encryption, we here compute $\mathbf{r} = G(H(pk), m)$ deterministically from the message, using hash functions $G : \{0,1\}^* \to \chi_r^k$ and $H : \{0,1\}^* \to \{0,1\}^{256}$. This modification, initially proposed in [5], and adapted to the case of lattices in [9], ensures the same message always produces the same ciphertext under a given public key. This is essential for applying the HBE transformation s described in Section 3.

- **KeyGen**$(R, q, p, k, \chi_f, \chi_g)$:
  1. Sample $\{f_{ij}\}_{i,j\in[k]} \leftarrow \$\chi_f^{k\times k}$ and construct $\mathbf{F}$ by Eq. (12) until $\mathbf{F}$ is invertible in $R_q^{k\times k}$
  2. Sample $\mathbf{g} \leftarrow \$\chi_g^k$
  3. Compute $\mathbf{h} = \mathbf{F}^{-1} \cdot \mathbf{g} \bmod q$
  4. Return $pk := \mathbf{h} \in R_q^k, \quad sk := \det(\mathbf{F})$
- **Encryption**$(pk = \mathbf{h}, m)$:
  1. Compute $h_{pk} = H(pk)$
  2. Derive randomness $\mathbf{r} = G(h_{pk}, m) \in \chi_r^k$
  3. Return $c = p \cdot \mathbf{h}^T\mathbf{r} + m \bmod q$
- **Decryption**$(sk = (\det(\mathbf{F}), \mathbf{h}), c)$:
  1. Return $m = c \cdot \det(\mathbf{F}) \bmod p$

Fig. 3. Deterministic Module-NTRU-based (OW-CPA-secure encryption scheme)

As we will see in our implementation section, the resulting encryption scheme possesses two key properties:

1. **Compact ciphertexts**: Unlike standard Module-LWE schemes that require rank-$(k+1)$ ciphertexts, our scheme achieves rank-1 ciphertexts (single polynomial) by using the Module-NTRU structure. This results in ciphertext sizes of approximately 1024 bytes for NIST Level 3 security [2].
2. **Efficient Module-NTRU structure**: Operating over polynomial rings $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ with module rank k, the scheme enables efficient NTT-based multiplication for fast operations [14].

Correctness. The correctness proof is identical to that of Bai et al. [2], since our deterministic derivation of the encryption randomness does not affect the correctness argument.

Security. Under the Module-NTRU assumptions and modeling G and H as random oracles, the base scheme achieves OW-CPA security. More precisely, we have the following theorem.

Theorem 3 (OW-CPA Security of Base Scheme). *The deterministic encryption scheme given in Figure 3 achieves OW-CPA security in the random oracle model, under the* $\mathsf{MNTRU}_{R_q,k,\chi_\mathbf{F},\chi_\mathbf{g}}$ *and the* $\mathsf{MLWE}_{R_q,k,\chi_s,\chi_e}$ *assumptions.*

Proof. The proof follows the same principle as the one given in [2], except that we have to manage the random oracle for both G and H. More precisely, assume $\mathcal{A}$ can break the OW-CPA security of our scheme with non-negligible advantage. The goal of the proof is to construct a simulator $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Search MLWE problem. Hence, $\mathcal{B}$ receives Search MLWE samples $(\mathbf{a}_i, b_i)$ and simulates the public key pk by setting $\mathbf{h}_i = \mathbf{a}_i/p \pmod q$. By the Decisional MNTRU assumption, $\mathbf{h}_i$ is computationally indistinguishable from a genuine public key.

To recover $\mathbf{s}$, $\mathcal{B}$ invokes $\mathcal{A}$ at least k times to collect k linear equations. For each MLWE sample $(\mathbf{a}_i, b_i)$, $\mathcal{B}$ proceeds as follows to generate the challenge ciphertext.

1. Select a message $m_i \in R_p$.
2. Request the random oracle H on input $\mathbf{h}_i$ to get a random value $\mathbf{r}_{\mathbf{h}_i}$.
3. Request the random oracle G on input $r_{\mathbf{h}_i} \| m_i$ to get a random value $\mathbf{r}_{m_i}$.
4. Constructs a ciphertext $c_i = p \cdot \mathbf{h}_i^T \mathbf{r}_{m_i} + m_i \pmod q$

As we are in the random oracle model, the adversary is not able to distinguish $\mathbf{r}_{m_i}$ from a true random value in χ_r^k. Hence, a ciphertext in our scheme is exactly similar to a ciphertext in [2]. We can then obviously proceed as in the initial paper.

After all the executions with $\mathcal{A}$, if the obtained linear system is non-singular, we can recover the secret by linear algebra, hence breaking the Search MLWE problem. The probability of seeing a non-singular matrix of dimension k over R_q is given in Lemma 3.7 in [2]. We also need to take care of the fact that the OW-CPA adversary $\mathcal{A}$ is seeing samples with the same secret, which mean that we needs to re-randomize the MLWE secret, which is done using a Rényi divergence argument provided in Lemma 3.5 of [2]. □

4.3 Our OW-CCA Scheme

Applying the generic Hash-Before-Encrypt transformation from Section 3 to the deterministic OW-CPA encryption scheme above immediately yields an OW-CCA-secure construction. The resulting scheme is shown in Figure 4.

- **KeyGen**: Same as base scheme (Figure 3)
- **Encryption**$_{HBE}(pk = \mathbf{h}, m)$:
  1. Compute hash: $h = H(m) \in \{0,1\}^{256}$
  2. Augment message: $m' = m \| h$
  3. Derive randomness $\mathbf{r} = G(H(pk), m') \in \chi_r^k$
  4. Return $c = p \cdot \mathbf{h}^T \mathbf{r} + m' \bmod q$
- **Decryption**$_{HBE}(sk = (\det(\mathbf{F}), \mathbf{h}), c)$:
  1. Compute $m^* = (c \cdot \det(\mathbf{F})) \bmod p$
  2. Parse m^* as $(m \| h)$
  3. Recompute hash $h' = H(m)$
  4. If $h' \neq h$, return $\perp$
  5. Return m

Fig. 4. Module-NTRU-based OW-CCA-secure encryption scheme)

Security Arguments. Correctness is obvious, based on the one of the OW-CPA version. Additionally, based on Theorem 2 and Theorem 3, we directly get the following corollary.

Corollary 1 (OW-CCA Security with HBE Transform). *The HBE-transformed scheme given in Figure 4 achieves OW-CCA security in the random oracle model, under the* $\mathsf{MNTRU}_{R_q,k,\chi_{\mathbf{F}},\chi_{\mathbf{g}}}$ *and the* $\mathsf{MLWE}_{R_q,k,\chi_s,\chi_e}$ *assumptions.*

Alternative FO-based transformation. As an alternative to the HBE transformation used in our main construction, the same deterministic OW-CPA encryption scheme can also be transformed into an OW-CCA-secure scheme using the Fujisaki–Okamoto (FO) transform. This provides an alternative generic route to OW-CCA security for our underlying Module-NTRU encryption scheme. We do not use the FO-transformed variant in our implementation or performance evaluation; it is considered here to establish the compatibility of our underlying OW-CPA construction with the FO paradigm.

Corollary 2 (OW-CCA Security with FO Transform). *Let* Π *denote the deterministic OW-CPA encryption scheme given in Figure 3. Applying the Fujisaki–Okamoto (FO) transform to* Π *yields an OW-CCA-secure public-key encryption scheme in the random oracle model, under the corresponding assumptions of the FO transform.*

5 The SPAKE Framework

This section recalls the notion of Single-party Public Key Authenticated Key Exchange (SPAKE) and the ALIKE framework proposed by Coron et al. [17]. We first review the underlying model, then describe the protocol, and finally recall its security guarantees.

Remark. An alternative to building upon the ALIKE framework would have been to use the construction of Fujioka et al. [23], which derives a CK^+-secure authenticated key exchange protocol from an OW-CCA-secure encryption scheme. Our post-quantum OW-CCA encryption scheme is fully compatible with this framework as well.

5.1 Single-party Public Key Authenticated Key Exchange

As defined in [17], a Single-party Public Key Authenticated Key Exchange protocol consists of a key-generation algorithm together with an interactive protocol executed between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$.

Procedures. More formally, we have the following procedures.

– KeyGen is a key generation procedure executed by $\mathcal{P}$, which given a security parameter λ generates a key pair $(\mathsf{sk}, \mathsf{pk})$.

– An interactive protocol executed by both $\mathcal{P}$ and $\mathcal{V}$, with the following algorithms.
  - Commit is executed by $\mathcal{P}$ which on input sk outputs a commitment com and some auxiliary information denoted $\mathsf{aux}_{\mathcal{P}}$.
  - $\mathsf{Challenge}$ is executed by $\mathcal{V}$ which on input $(\mathsf{pk}, \mathsf{com})$ outputs a challenge chal which is sent to $\mathcal{P}$, and some auxiliary information $\mathsf{aux}_{\mathcal{V}}$.
  - $\mathsf{Response}$ is executed by $\mathcal{P}$ which on input $(\mathsf{sk}, \mathsf{chal})$ outputs a response res and a key k.
  - Verif is executed by $\mathcal{V}$ which on input $(\mathsf{chal}, \mathsf{res}, \mathsf{aux})$ outputs k'. If $\mathsf{k}' = \perp$, then it aborts the protocol.

For simplicity, we assume that the public key pk is certified and that $\mathcal{V}$ always checks such a certificate sent by $\mathcal{P}$.

Correctness. Correctness requires that, whenever

$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^{\lambda}),$$

and both parties execute the protocol honestly, they derive the same session key, namely

$$\mathsf{Verif}(\mathsf{chal}, \mathsf{res}, \mathsf{aux}_{\mathcal{V}}) = \mathsf{k}.$$

Security. The protocol is expected to satisfy two security properties [17]. First, unforgeability guarantees that no adversary can impersonate a prover ($\mathcal{P}$) without knowing its private key sk. Second, channel secrecy guarantees that an adversary observing protocol executions cannot recover the established session key k.

Attack model. An adversary $\mathcal{A}$ can be *passive* (eavesdropping only) or *active* (interacting with the prover $\mathcal{P}$). Note that server $\mathcal{V}$ authentication is **not** required, so active attackers can trivially impersonate a reader. Consequently, channel secrecy is considered only against passive adversaries. We first provide the below game, where $\mathcal{C}$ is a challenger, and $\mathcal{A}$ is an adversary.

Security game:

1. $\mathcal{C}$ generates a key pair $(\mathsf{pk}, \mathsf{sk})$ and sends pk to $\mathcal{A}$.
2. Using sk, $\mathcal{C}$ simulates n protocol executions between a prover and a verifier, and sends the protocol transcripts $(T_1, \ldots, T_n)$ to $\mathcal{A}$.
3. $\mathcal{A}$ can request any of the session keys used in the transcripts $(T_1, \ldots, T_n)$.
4. **(Active attack).** $\mathcal{A}$ can engage in up to n protocol executions with $\mathcal{C}$, who plays the role of the prover $\mathcal{P}$. $\mathcal{C}$ answers using sk. At the end of each protocol execution, $\mathcal{A}$ obtains the corresponding session key k.
5. **(Secrecy).** $\mathcal{A}$ outputs one of the session keys matching the protocol transcripts $(T_1, \ldots, T_n)$, not previously revealed in step 3.
6. **(Unforgeability).** $\mathcal{C}$ plays the role of the verifier $\mathcal{V}$ and $\mathcal{A}$ plays the role of the prover $\mathcal{P}$. $\mathcal{A}$ must output res such that $\mathsf{Verif}(\mathsf{chal}, \mathsf{res}, \mathsf{aux}) \neq \perp$.

Definition 8 ([17]). *A SPAKE protocol is said to be* (t, ε)-secure *if no adversary running in at most* t *elementary steps can impersonate a prover* $\mathcal{P}$ *or recover the session key of an observed transaction with probability greater than* ε.

5.2 Description of ALIKE

We now recall the generic ALIKE construction proposed in [17], later standardized as ISO/IEC 29192-4. In Addition to a OW-CCA secure public key encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ as described in Section 2, ALIKE requires the use of a block cipher $\mathsf{B} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. In the security proof [17], the security proof models the block cipher as an ideal cipher using the Ideal Cipher Model (ICM), which is equivalent to the Random Oracle Model under suitable assumptions [18].

Instantiating ALIKE with our lattice-based OW-CCA-secure encryption scheme yields a post-quantum version of the protocol, which we denote by PQ-ALIKE.

Key generation. During KeyGen, the prover $\mathcal{P}$ executes the $\Pi.\mathcal{K}$ algorithm to get a key pair $(\mathsf{sk}_i, \mathsf{pk}_i)$ for the public key encryption scheme. We also assume that the public key is certified by some Certification Authority.

Interactive Protocol. The protocol between two parties $\mathcal{P}$ and $\mathcal{V}$ is given in Figure 5.

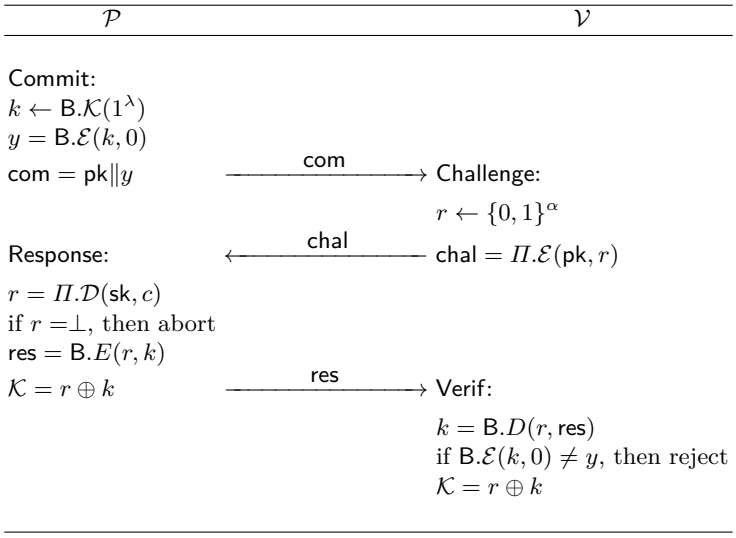

Fig. 5. ALIKE Protocol

5.3 Security

The security analysis of ALIKE established in [17] immediately applies to PQ-ALIKE, since only the underlying public-key encryption scheme is replaced. We therefore recall the two main security results.

Theorem 4 (Active Unforgeability [17]). *The (PQ-)ALIKE scheme is* (t, ϵ)*-secure against unforgeability under active attacks, in the ideal cipher model, assuming that the underlying public-key encryption scheme* Π *is* (t', ϵ')*-OW-CCA secure, where* $t = t' - \mathsf{poly}(\lambda, q, n)$ *and* $\epsilon = \epsilon' + 2 - \beta$*, where* q *is the number of queries to the ideal cipher, and* n *is the number of protocol transcripts.*

Theorem 5 (Passive Channel Secrecy [17]). *The (PQ-)ALIKE scheme is* (t, ϵ)*-passively secure against secrecy, in the ideal cipher model, assuming that* Π *is* (t', ϵ')*-OW-CPA, where* $t = t' - \mathsf{poly}(\lambda, q, n)$ *and* $\epsilon = n \cdot \epsilon' + n \cdot 2^{-\alpha}$*, where* q *is the number of queries to the ideal cipher, and* n *is the number of protocol transcripts.*

These results show that replacing RSA with our lattice-based OW-CCA-secure encryption scheme preserves the original ALIKE security guarantees while providing resistance against quantum adversaries. Consequently, PQ-ALIKE inherits the same provable security framework as ALIKE under the corresponding post-quantum assumptions.

6 Implementations

Our implementation focuses on the Hash-Before-Encryption (HBE) instantiation of the proposed framework. Although the previous section presented both the Fujisaki–Okamoto (FO) and HBE generic transformations from OW-CPA to OW-CCA security, HBE was selected for implementation because it avoids the additional re-encryption step required by FO while providing the level of security required by the ALIKE framework. Combined with our deterministic Module-NTRU encryption scheme, this results in PQ-ALIKE, a lightweight post-quantum SPAKE protocol suitable for resource-constrained environments. This section details our optimization strategies, concrete performance benchmarks on an embedded platform, and the security guarantees provided by PQ-ALIKE. Our benchmarks were conducted on a STM32F4-DISCOVERY board (ARM Cortex-M4 at 120 MHz), demonstrating the efficiency of our implementation in a lightweight environment.

6.1 Parameter Selection and Rationales

To strike an optimal balance between security and performance, we selected parameters compatible with the Number Theoretic Transform (NTT), enabling fast and efficient polynomial operations. The parameter sets, summarized in Table 1, are optimized for NIST-equivalent security levels while minimizing resource consumption.

In addition to NTT compatibility, the choice of the modulus q is driven by both correctness and security considerations. In particular, q is selected such that the noise growth during encryption and decryption remains well below q/2, ensuring a negligible decryption failure probability. At the same time, q directly influences the hardness of the underlying Module-NTRU problems: choosing q

too large may weaken resistance to lattice reduction attacks, while choosing it too small increases the risk of decryption errors. The specific q values used in Table 1 therefore represent a trade-off between these constraints and differ slightly from those in [3] to better match the targeted implementation and NTT efficiency requirements.

Table 1. Parameter sets for the Module-NTRU implementation ($k = 2$, $p = 3$).

Parameter Set	n	k	q	NIST Level
II-NTRU(a)	512	2	32,257	L3
II-NTRU(b)	512	2	51,713	L3
III-NTRU	1024	2	119,809	L5 (future work)

The parameters were chosen to (i) minimize key and ciphertext sizes as public keys and ciphertexts are compact (e.g., 1920 bytes for II_ntru(a)), reducing communication overhead, (ii) optimize cryptographic operations as the modulus q satisfies the condition $n|(q-1)$, ensuring the existence of primitive n-th roots of unity and enabling efficient polynomial multiplication via NTT, while also being chosen to balance noise growth and security requirements, (iii) ensure robust security as decryption failure probability is negligible, and resistance to lattice reduction attacks is based on the assumed hardness of the Module-NTRU problems, (iv) reach a fully plausible security level for real-life applications.

We estimated the concrete security of each parameter set using the lattice estimator of Albrecht et al. [1], evaluating the cost of primal uSVP, BDD, BDD-hybrid, BDD-MITM-hybrid, and DSD attacks under the classical gate cost model ($2^{0.292\beta}$). NIST security levels are defined in terms of classical security: Level 3 requires at least 2^{192} classical operations (equivalent to AES-192 exhaustive key search) and Level 5 requires 2^{256}. Since our scheme is a Module-NTRU construction with rank $k = 2$, the effective lattice dimension input to the estimator is $k \cdot n$. The dominant attack across all parameter sets is the BDD attack. For II-NTRU(a) ($n = 512$, $q = 32{,}257$) the estimator yields $2^{207.1}$ classical operations, and for II-NTRU(b) ($n = 512$, $q = 51{,}713$) it yields $2^{198.8}$, both exceeding the Level 3 threshold by 15.1 and 6.8 bits respectively. For III-NTRU ($n = 1024$, $q = 119{,}809$) the estimator yields $2^{392.1}$, far exceeding the Level 5 threshold. The reduced margin of II-NTRU(b) relative to II-NTRU(a) is expected: a larger q moves closer to the NTRU overstretching boundary while remaining necessary for correct decryption. Both Level 3 variants nonetheless satisfy their claimed security level.

6.2 Optimization Techniques

To achieve state-of-the-art performance on resource-constrained platforms, our implementation leverages several advanced optimization strategies, each tailored to exploit the unique properties of lattice-based cryptography and the target hardware.

Number Theoretic Transform (NTT). The core of our polynomial arithmetic relies on the Number Theoretic Transform (NTT), a specialized variant of the Fast

Fourier Transform (FFT) adapted for finite rings. NTT enables efficient polynomial multiplication in $O(n \log n)$ time, a critical requirement for lattice-based schemes where polynomial operations dominate the computational cost. Our implementation employs a radix-2 NTT with degree-2 splitting, as described in [6], to minimize the number of modular reductions and memory accesses. To further accelerate the transform, we use Barrett reduction for base multiplication, a technique that replaces expensive division operations with multiplications and shifts, as detailed in [4]. This approach is particularly effective on microcontrollers lacking hardware support for 64-bit division. Additionally, we precompute the roots of unity (zetas) and their inverses, storing them in static memory to eliminate redundant calculations during runtime. The NTT butterflies are unrolled in a 4-way loop where beneficial, reducing loop overhead and improving instruction cache utilization. The inverse NTT is optimized by fusing the final normalization step, as suggested in [33], to minimize the number of passes over the polynomial coefficients.

Memory Management and Efficient Sampling. Memory efficiency is critical for embedded systems with limited RAM. To avoid dynamic allocation and heap fragmentation, all working buffers (including those required for NTT, polynomial inversion, and key generation) are statically allocated in the Binary Static Section (BSS). This ensures deterministic memory usage and prevents runtime overhead associated with heap management. For polynomial sampling, we use a hardware-based random number generator (RNG) via the STM32's built-in RNG peripheral, which provides cryptographically secure randomness without software delays. The sampling process follows a rejection sampling approach with constant-time loops, as recommended in [16], to mitigate timing side-channel attacks. The public key is stored in its NTT-transformed form immediately after generation, eliminating the need for repeated forward NTT operations during encryption. This optimization is inspired by the "lazy reduction" technique used in Kyber [9], where precomputed representations of polynomials are reused to minimize redundant computations.

6.3 Concrete Performance Benchmarks

In this section, we provide the concrete performance results we obtained on STM32L4R5 (NUCLEO-L4R5ZI board), starting with the encryption system (see Section 4) and continuing with a complete version of our SPAKE protocol (see Section 5).

Our HBE-based OW-CCA encryption scheme Regarding our HBE-based OW-CCA-secure encryption scheme, Table 2 compares computational costs for key operations, while Table 3 details key and ciphertext sizes. Those performances have been obtained through a mean (min-max) over 100 execution of each process. Key generation is dominated by the polynomial inversion. While this operation is computationally intensive, accounting for the majority of the $\sim$18.8 million cycles required, its impact on real-world performance is mitigated by

the fact that key generation is a one-time operation performed during device initialization, especially in PQ-ALIKE construction. Once generated, keys are reused for many encryption and decryption operations, making the amortized cost negligible in practice.

Since the FO transformation was introduced only as an alternative generic construction and incurs an additional re-encryption during decryption, we benchmark only the HBE instantiation, which is the construction adopted in PQ-ALIKE.

Table 2. Performance comparison on ARM Cortex-M4 (cycles $\times 10^3$). Our scheme runs at 120 MHz. Saber [29] on Cortex-M4 as well as NTRU-HPS-677 [28]. ML-KEM (FIPS 203 [36]) cycle counts measured by wolfSSL [40] on STM32 NUCLEO-F439ZI, Cortex-M4 @ 168 MHz. Cycle counts are frequency-independent.

Scheme	Level	Type	KeyGen	Enc/ Encap	Dec/ Decap
Kyber-768 (v2) [10]	L3	IND-CCA KEM	947	1,113	1,059
Saber [29]	L3	IND-CCA KEM	1,147	1,444	1,543
ML-KEM-768 [40]	L3	IND-CCA KEM	1,095	1,104	1,400
NTRU-HPS-677 [28]	L3	IND-CCA KEM	144,383	956	837
Ours – II-NTRU(a)	L3	OW-CCA PKE	18,803	342	266
Ours – II-NTRU(b)	L3	OW-CCA PKE	18,830	341	266

† ML-KEM cycle counts converted from milliseconds reported in [40] at 168 MHz. Kyber-768 [10] and ML-KEM [36] are based on the same scheme; both are included as they represent optimised implementations from 2019 and 2025 respectively.

Table 3. Key and ciphertext sizes in bytes. Lizard [13] targets 128-bit classical security. ML-KEM sizes are from FIPS 203 [36].

Scheme	NIST Level	Public Key	Secret Key	Ciphertext
Saber [21]	L3	992	2,304	1,088
ML-KEM-768 [36]	L3	1,184	2,400	1,088
Lizard [13]	128-bit cl.	1,622,000	34,000	955
NTRU-HPS-677 [12]	L3	930	1234	930
Ours – II-NTRU(a)	L3	1,920	6,144	1,024
Ours – II-NTRU(b)	L3	2,048	6,144	1,024

PQ-ALIKE over HBE Module-NTRU. The proposed HBE-based OW-CCA encryption scheme is then integrated into PQ-ALIKE to obtain a lightweight post-quantum authenticated key exchange protocol. The performance of the full protocol is summarized in Table 4.

6.4 Comparison

Our implementation demonstrates significant performance and memory advantages compared to leading post-quantum lattice-based schemes optimised for embedded platforms. We benchmark against five proposals: Kyber-768 as implemented by Botros et al. [10] and its NIST-standardised successor ML-KEM

Table 4. PQ-ALIKE protocol phase performance on ARM Cortex-M4 @ 120 MHz (mean over 100 runs). Commit includes key generation. Challenge is performed by the Verifier. Response and Verify are performed by the Prover.

Scheme	Commit (cycles)	Challenge (cycles)	Response (cycles)	Verify (cycles)	Prover Total (cycles)
II-NTRU(a)	∼18,835,099	1,195,936	380,630	39,502	420,132
II-NTRU(b)	∼18,860,584	1,193,336	380,071	39,488	419,559

Note on Commit cycles: The Commit phase consists of a lightweight randomness generation step (7,985 / 7,981 cycles) followed by key generation (∼18.8M cycles). The total Commit cost is therefore dominated by key generation, as reported above. Prover Total combines Response and Verify (≈ 3.50 ms at 120 MHz).

(FIPS 203 [36]) benchmarked by wolfSSL [40] — both included as they represent 2019 and 2025 state-of-the-art optimisations of the same underlying scheme — Saber on ARM Cortex-M4 by Karmakar et al. [29], NTRU-HPS-2048-677 from pqm4 [28], and the lightweight identification protocol by Düzlü et al. [22]. Rudraksh [30] targets NIST Level 1 security exclusively and is implemented on FPGA only; it is therefore excluded from all comparisons.

Kyber-768 cycle counts are taken from Botros et al. [10] at 168 MHz. Saber cycle counts are from Karmakar et al. [29] on Cortex-M4. ML-KEM cycle counts are from wolfSSL benchmarks on an STM32 NUCLEO-F439ZI (Cortex-M4 @ 168 MHz) [40], converted from milliseconds using the platform frequency. NTRU-HPS-2048-677 cycle counts are from the pqm4 project [28] at 168 MHz. Since cycle counts are frequency-independent, differences in clock speed do not affect the validity of these comparisons. Düzlü et al. report cycles at 24 MHz on the same Cortex-M4 platform; the same reasoning applies.

Encryption and Decryption. Our OW-CCA-secure encryption scheme outperforms all compared schemes in both encryption and decryption. Against Kyber-768 [10], we achieve a 69.4% reduction in encryption cycles (341K vs. 1,113K) and a 75% reduction in decryption cycles (265K vs. 1,059K). Against Saber (speed-optimised) [29], our scheme is 4.2× faster in encryption (342K vs. 1,444K) and 5.8× faster in decryption (266K vs. 1,543K). Against ML-KEM-768 [40], the gains are 3.2× in encryption (342K vs. 1,104K) and 5.3× in decryption (266K vs. 1,400K). This efficiency stems from the compact rank-1 ciphertext structure of Module-NTRU, which requires fewer polynomial multiplications than the rank-$(k+1)$ other designs. Düzlü et al. [22] focuses on identification rather than key exchange; direct comparison is therefore not meaningful.

Key Generation. Our key generation requires approximately 18.8M cycles at both security levels, which is higher than LWE/LWR-based schemes such as Kyber and Saber. This is an inherent characteristic of NTRU-family lattices: the standard NTRU-HPS-2048-677 at the same NIST Level 3 requires 143.7M cycles for key generation [28], more than 7.6× our cost. Our Module-NTRU construction therefore achieves significantly faster key generation than classical NTRU while preserving the structural advantages of the NTRU lattice. In the SPAKE

protocol, key generation is performed only once per session by the Prover, while encryption and decryption are repeated across protocol steps. Optimising key generation further remains future work.

Key and Ciphertext Sizes. Our scheme achieves a public key of 1.9 KB for II-NTRU(a) compared to Kyber-768 (1.2 KB) [10] and Saber (1.3 KB) [21]. Ciphertext sizes are competitive: 960 bytes for II-NTRU(a) vs. 1,088 bytes for Kyber-768 and 1,088 bytes for ML-KEM-768 [36]. The larger public key of our scheme is a known characteristic of NTRU-based constructions and remains acceptable for the IoT settings targeted by this work, where computation cost outweighs communication overhead. Düzlü et al.'s protocol [22] offers smaller keys but lacks full AKE functionality.

Memory Usage. In terms of RAM and flash usage, our implementation (208 KB RAM, 53.5 KB flash) is competitive with Botros et al.'s Kyber optimisation (210 KB RAM, 55 KB flash) [10]. Düzlü et al. [22] do not report RAM or flash consumption for their Cortex-M4 implementation. Our memory efficiency is attributed to static buffer allocation and reuse of NTT-transformed keys, which minimises runtime overhead.

Table 5. Communication cost per protocol exchange (bits). Fujioka et al. [23] costs are theoretical (from their Table 1); all others computed as $(\mathrm{pk} + \mathrm{ct}) \times 8$. Only post-quantum instantiations of [23] are included.

Scheme	Assumption	Model	Comm. Cost (bits)
Fujioka et al. [23], Ring-LWE	Ring-LWE	ROM	≈80,600
Fujioka et al. [23], Codes	Code-based	ROM	≈52,320
Fujioka et al. [23], NTRU	NTRU	ROM	≈19,756
Saber [21]	Module-LWR	ROM	16,640
ML-KEM-768 [36]	Module-LWE	ROM	18,176
NTRU-HPS-677 [12]	NTRU	ROM	14,880
Ours – II-NTRU(a)	Module-NTRU	ROM	23,040
Ours – II-NTRU(b)	Module-NTRU	ROM	24,576

Table 5 compares communication costs. Then, NTRU-HPS-677 [12] achieves the lowest cost (14,880 bits); our scheme requires 23,040 bits (II-NTRU(a)) and 24,576 bits (II-NTRU(b)), remaining below all post-quantum instantiations of [23] and uniquely providing a Cortex-M4 implementation based on the HBE transformation rather than the Fujisaki–Okamoto transform.

6.5 Future Directions for Optimization

While our current implementation achieves competitive performance, further optimizations are possible. One promising avenue is the integration of **fast divstep** algorithms, such as those proposed in [7], which could reduce the cost of polynomial inversion by up to 30%. Another direction is the adoption of NTT-friendly primes that enable more efficient modular reduction, as explored in [32]. For

high-security applications, we could also add some masking techniques to protect against side-channel attacks. Finally, the use of hardware acceleration such as custom instructions for NTT or modular arithmetic could provide significant speedups on platforms that support it.

Acknowledgments

We would like to thank anonymous reviewers for their helpful discussions and valuable comments. Third author is funded by the ARES PRD PFS 2022 2027 Programme on Cybersecurity and Cryptology at The University of Bamenda in Cameroon. The work of the second author has been conducted within the Intelligent Cybersecurity for Mobility Systems (ICMS) research chair at Télécom Paris, founded by Ampere, BCG, IRT SystemX, Renault, Solent, Thales, Valeo, and ZF Group, and supported by the Fondation Mines-Télécom.

References

1. M. R. Albrecht, B. R. Curtis, A. Deo, A. Davidson, R. Player, E. W. Postlethwaite, F. Virdia, and T. Wunderer. Estimate all the LWE, NTRU schemes! In D. Catalano and R. De Prisco, editors, *SCN 18*, volume 11035 of *LNCS*, pages 351–367. Springer, Cham, Sept. 2018.
2. S. Bai, H. Jangir, H. Lin, T. Ngo, W. Wen, and J. Zheng. Compact encryption based on module-NTRU problems. In M.-J. Saarinen and D. Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part I*, pages 371–405. Springer, Cham, June 2024.
3. S. Bai, H. Jangir, H. Lin, T. Ngo, W. Wen, and J. Zheng. Compact encryption based on module-NTRU problems. Cryptology ePrint Archive, Report 2024/732, 2024.
4. P. Barrett. Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor. In A. M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 311–323. Springer, Berlin, Heidelberg, Aug. 1987.
5. M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 535–552. Springer, Berlin, Heidelberg, Aug. 2007.
6. D. J. Bernstein. *Fast multiplication and its applications*, page 325–384. Mathematical Sciences Research Institute Publications. Cambridge University Press, 2008.
7. D. J. Bernstein and B.-Y. Yang. Fast constant-time gcd computation and modular inversion. *IACR TCHES*, 2019(3):340–398, 2019.
8. U. Blumenthal, B. Luo, S. O'Melia, G. Torres, and D. A. Wilson. PQuAKE - Post-Quantum Authenticated Key Exchange. Internet-Draft draft-uri-lake-pquake-00, IETF Secretariat, April 2025.
9. J. W. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé. CRYSTALS - Kyber: A CCA-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy*, pages 353–367. IEEE Computer Society Press, Apr. 2018.

10. L. Botros, M. J. Kannwischer, and P. Schwabe. Memory-efficient high-speed implementation of Kyber on Cortex-M4. In J. Buchmann, A. Nitaj, and T. eddine Rachidi, editors, *AFRICACRYPT 19*, volume 11627 of *LNCS*, pages 209–228. Springer, Cham, July 2019.
11. K. Boudgoust, C. Jeudy, A. Roux-Langlois, and W. Wen. Entropic hardness of module-LWE from module-NTRU. In T. Isobe and S. Sarkar, editors, *INDOCRYPT 2022*, volume 13774 of *LNCS*, pages 78–99. Springer, Cham, Dec. 2022.
12. C. Chen, O. Danba, J. Hoffstein, A. Hulsing, J. Rijneveld, J. M. Schanck, P. Schwabe, W. Whyte, Z. Zhang, T. Saito, T. Yamakawa, and K. Xagawa. NTRU. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.
13. J. H. Cheon, D. Kim, J. Lee, and Y. Song. Lizard: Cut off the tail! A practical post-quantum public-key encryption from LWE and LWR. In D. Catalano and R. De Prisco, editors, *SCN 18*, volume 11035 of *LNCS*, pages 160–177. Springer, Cham, Sept. 2018.
14. C. Chuengsatiansup, T. Prest, D. Stehlé, A. Wallet, and K. Xagawa. ModFalcon: Compact signatures based on module-NTRU lattices. In H.-M. Sun, S.-P. Shieh, G. Gu, and G. Ateniese, editors, *ASIACCS 20*, pages 853–866. ACM Press, Oct. 2020.
15. K. Cong, D. Cozzo, V. Maram, and N. P. Smart. Gladius: LWR based efficient hybrid public key encryption with distributed decryption. In M. Tibouchi and H. Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 125–155. Springer, Cham, Dec. 2021.
16. J.-S. Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In Çetin Kaya. Koç and C. Paar, editors, *CHES'99*, volume 1717 of *LNCS*, pages 292–302. Springer, Berlin, Heidelberg, Aug. 1999.
17. J.-S. Coron, A. Gouget, P. Paillier, and K. Villegas. SPAKE: A single-party public-key authenticated key exchange protocol for contact-less applications. In R. Sion, R. Curtmola, S. Dietrich, A. Kiayias, J. M. Miret, K. Sako, and F. Sebé, editors, *FC 2010 Workshops*, volume 6054 of *LNCS*, pages 107–122. Springer, Berlin, Heidelberg, Jan. 2010.
18. J.-S. Coron, J. Patarin, and Y. Seurin. The random oracle model and the ideal cipher model are equivalent. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 1–20. Springer, Berlin, Heidelberg, Aug. 2008.
19. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 13–25. Springer, Berlin, Heidelberg, Aug. 1998.
20. J.-P. D'Anvers, A. Karmakar, S. S. Roy, and F. Vercauteren. Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM. In A. Joux, A. Nitaj, and T. Rachidi, editors, *AFRICACRYPT 18*, volume 10831 of *LNCS*, pages 282–305. Springer, Cham, May 2018.
21. J.-P. D'Anvers, A. Karmakar, S. S. Roy, F. Vercauteren, J. M. B. Mera, M. Van Beirendonck, and A. Basso. SABER. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.
22. S. Düzlü, J. Krämer, T. Pöppelmann, and P. Struck. A lightweight identification protocol based on lattices. In A. Boldyreva and V. Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 95–113. Springer, Cham, May 2023.

23. A. Fujioka, K. Suzuki, K. Xagawa, and K. Yoneyama. Practical and post-quantum authenticated key exchange from one-way secure key encapsulation mechanism. In K. Chen, Q. Xie, W. Qiu, N. Li, and W.-G. Tzeng, editors, *ASIACCS 13*, pages 83–94. ACM Press, May 2013.
24. A. Fujioka, K. Suzuki, K. Xagawa, and K. Yoneyama. Strongly secure authenticated key exchange from factoring, codes, and lattices. *DCC*, 76(3):469–504, 2015.
25. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In M. J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 537–554. Springer, Berlin, Heidelberg, Aug. 1999.
26. D. Hofheinz, K. Hövelmanns, and E. Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In Y. Kalai and L. Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 341–371. Springer, Cham, Nov. 2017.
27. H. Jiang, Z. Zhang, L. Chen, H. Wang, and Z. Ma. IND-CCA-secure key encapsulation mechanism in the quantum random oracle model, revisited. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 96–125. Springer, Cham, Aug. 2018.
28. M. J. Kannwischer, J. Rijneveld, P. Schwabe, and K. Stoffelen. pqm4: Testing and benchmarking NIST PQC on ARM Cortex-M4. Cryptology ePrint Archive, Report 2019/844, 2019.
29. A. Karmakar, J. M. B. Mera, S. S. Roy, and I. Verbauwhede. Saber on ARM CCA-secure module lattice-based key encapsulation on ARM. Cryptology ePrint Archive, Report 2018/682, 2018.
30. S. Kundu, A. Ghosh, A. Karmakar, S. Sen, and I. Verbauwhede. Rudraksh: A compact and lightweight post-quantum key-encapsulation mechanism. *IACR TCHES*, 2025(2):647–680, 2025.
31. A. Langlois and D. Stehlé. Worst-case to average-case reductions for module lattices. *DCC*, 75(3):565–599, 2015.
32. Z. Liu, P. Longa, G. C. C. F. Pereira, O. Reparaz, and H. Seo. Four$\mathbb{Q}$ on embedded devices with strong countermeasures against side-channel attacks. In W. Fischer and N. Homma, editors, *CHES 2017*, volume 10529 of *LNCS*, pages 665–686. Springer, Cham, Sept. 2017.
33. P. Longa and M. Naehrig. Speeding up the number theoretic transform for faster ideal lattice-based cryptography. In S. Foresti and G. Persiano, editors, *CANS 16*, volume 10052 of *LNCS*, pages 124–139. Springer, Cham, Nov. 2016.
34. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, Berlin, Heidelberg, May / June 2010.
35. M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*, pages 427–437. ACM Press, May 1990.
36. National Institute of Standards and Technology. FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard. https://doi.org/10.6028/NIST.FIPS.203, 2024. Federal Information Processing Standards Publication.
37. C. Peikert. Lattice cryptography for the internet. In M. Mosca, editor, *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014*, pages 197–219. Springer, Cham, Oct. 2014.
38. M. Sabt and J. Traoré. Breaking into the KeyStore: A practical forgery attack against android KeyStore. In I. G. Askoxylakis, S. Ioannidis, S. K. Katsikas, and C. A. Meadows, editors, *ESORICS 2016, Part II*, volume 9879 of *LNCS*, pages 531–548. Springer, Cham, Sept. 2016.

39. D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In K. G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 27–47. Springer, Berlin, Heidelberg, May 2011.
40. wolfSSL Inc. Updated post-quantum benchmarks for ML-KEM and ML-DSA on STM32. https://www.wolfssl.com/updated-post-quantum-benchmarks-for-ml-kem-and-ml-dsa-on-stm32/, 2025. STM32 NUCLEO-F439ZI, Cortex-M4 @ 168 MHz.