

USENIX

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

Non-Interactive Key Exchange from Lattices in the Standard Model

Ying Li and Lei Zhang, *East China Normal University;*
Qiqi Lai, *Shaanxi Normal University*

https://www.usenix.org/conference/usenixsecurity26/presentation/li-ying

This paper is included in the Proceedings of the 35th USENIX Security Symposium.

August 12–14, 2026 • Baltimore, MD, USA

ISBN 978-1-939133-58-8

Open access to the Proceedings of the 35th USENIX Security Symposium is sponsored by



جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

ARTIFACT EVALUATED
USENIX
AVAILABLE

Non-Interactive Key Exchange from Lattices in the Standard Model

Ying Li
East China Normal University
yingli@stu.ecnu.edu.cn

Lei Zhang*
East China Normal University
leizhang@sei.ecnu.edu.cn

Qiqi Lai*
Shaanxi Normal University
laiqq@snnu.edu.cn

Abstract

Existing non-interactive key exchange (NIKE) constructions in the CKS-heavy security model either incur loose security reductions or lack lattice instantiations due to the absence of suitable hash proof systems (HPS). To overcome this limitation, we introduce a new cryptographic primitive, named as *decoupled key encapsulation mechanisms* (decoupled KEMs), which separates recipient-independent ciphertext generation from recipient-specific key derivation. Moreover, we formalize the security of our decoupled KEM as MU-IND-wCCA$^{\text{Corr+}}$, which is a multi-user security notion. Based on this, we give a new framework that any decoupled KEM with MU-IND-wCCA$^{\text{Corr+}}$ implies a NIKE protocol with much tighter security in the CKS-heavy security model. For concrete instantiation, we first construct a specific HPS based on Modulus-LWE in the standard model, and then obtain a decoupled KEM with the desired security notion. Applying our framework, we obtain a lattice-based NIKE secure in the standard model, denoted $\mathsf{NIKE}_{\mathsf{HPS}}$. $\mathsf{NIKE}_{\mathsf{HPS}}$ achieves a linear security reduction loss in the number of users and close to the known optimum, improving upon the quadratic loss of SWOOSH (USENIX Security' 24) under CKS-heavy security. Experiments show that our protocol achieves significant speedups in key generation and shared-key derivation over SWOOSH.

1 Introduction

Key exchange (KE) is a fundamental cryptographic primitive that enables two users to establish a shared key over an insecure communication channel. Classical KE protocols [8, 18, 35, 39, 52, 54] typically rely on interactive exchanges, which implicitly assume synchronous availability and stable network conditions. However, in many practical settings, such as large-scale distributed systems, intermittently connected devices, or bandwidth-constrained environments, these assumptions no longer hold.

Non-interactive key exchange (NIKE) formalizes the ability for users to derive shared keys solely from publicly available information, without interaction. This abstraction aligns with requirements that have been recognized in the design of large-scale and asynchronous systems, including secure group messaging and asynchronous key establishment [4, 16]. More broadly, NIKE captures the notion of non-interactive key derivation that underlies a range of higher-level cryptographic primitives, such as broadcast encryption [12].

Most existing NIKE constructions are rooted in Diffie-Hellman (DH) or factoring-related assumptions [19, 24], whose security is threatened by the advent of quantum computing. In particular, Shor's algorithm [53] renders discrete logarithms and integer factorization solvable in polynomial time. Moreover, lattice reduction techniques [42] and generic search speedups [29], further narrow the security margin of classical cryptographic assumptions. These developments have catalyzed a shift toward post-quantum cryptography. Lattice-based constructions stand out as one of the few known approaches capable of supporting NIKE while offering conjectured resistance against both classical and quantum attacks.

1.1 Related Work

Since the introduction of DH KE [19], often regarded as the first instantiation of NIKE, the study of NIKE protocols has made significant progress.

Security Models. Early work by Cash, Kiltz, and Shoup [15] formalized the CKS security for NIKE in both the honest key registration (HKR) and dishonest key registration (DKR) settings, and proposed a concrete NIKE construction based on the twin DH problem. The two settings differ in how public keys are generated and registered: in the HKR setting, all public keys are generated honestly together with their corresponding private keys, whereas in the DKR setting, an adversary may register arbitrary public keys without knowing the associated private keys. Importantly, security established in the HKR setting can be generically lifted to the DKR setting using standard transformations. These typically rely

*Lei Zhang is the corresponding author.

on augmenting public-key registration with key-validation mechanisms, such as zero-knowledge proofs of knowledge of the corresponding private keys. Consequently, many existing NIKE constructions first establish security in the HKR setting and subsequently extend it to the DKR setting with moderate additional effort. Following this well-established approach, we focus on the HKR setting.

Building on the CKS framework, Freire, Hofheinz, Kiltz, and Paterson [24] refined NIKE security notions by introducing the CKS-light, CKS-heavy and m-CKS-heavy variants (under both HKR and DKR settings). CKS-heavy is strictly stronger than CKS-light, as it allows adaptive user corruptions. Although CKS-light security can be upgraded to CKS-heavy, the reduction incurs a quadratic loss in the number of users, namely $\mu^2/2$, where μ denotes the total number of users in the system. The m-CKS-heavy notion further allows multiple test queries, leading to an additional reduction loss of $\mu^2 t/2$, where t is the number of test queries. In practice, security against a single adaptive test query-as captured by CKS-heavy-is sufficient to preclude meaningful attacks, making it a natural and well-balanced target for NIKE design.

Beyond the CKS-style security notions, Hesse et al. [37] introduced an alternative notion for NIKE, termed v-semi-adaptive security. In this model, a test query is allowed only if the total number of distinct user identities involved across all corruption, reveal, and test queries is bounded by v. This notion is stronger than CKS-light and facilitates tighter security reductions in practical constructions. However, it is strictly weaker than m-CKS-heavy security and not directly comparable to CKS-heavy security, as it permits multiple test queries but enforces a strictly more restrictive form of adaptivity with respect to when and how such queries may be issued.

NIKE from Lattices. Lattice-based cryptography has made significant progress in recent years, garnering considerable attention within the post-quantum cryptography community. Notably, in the third round of the National Institute of Standards and Technology (NIST) Post-Quantum Cryptography (PQC) standardization process, lattice-based schemes such as the public-key encryption (PKE) and key encapsulation mechanisms (KEM) of CRYSTALS-Kyber [13], along with the digital signature CRYSTALS-Dilithium [22] and Falcon [23], were selected for standardization. These milestones underscore the promising potential of lattice-based cryptography.

While lattice-based cryptography has achieved significant milestones in standardization, lattice-based KE protocols have not yet been directly standardized by NIST. However, it is widely recognized that such protocols can be constructed using reconciliation mechanisms [21, 49] or lattice-based PKE/KEM [13]. Despite the development of several such protocols [9], most remain interactive, and practical implementations of lattice-based NIKE protocols are still lacking.

Recent research has investigated various approaches to lattice-based NIKE constructions, with the most compelling candidates relying on standard hardness assumptions such as SIS or LWE. However, SIS-based NIKE protocols are inherently infeasible due to conflicting dimension requirements [20]. LWE-based constructions avoid this issue, but as observed by Guo et al. [30,31], when both parties' public keys are LWE samples, reconciliation is needed to handle LWE noise. This reconciliation typically introduces interaction, thereby violating the non-interactive nature of NIKE.

Despite these challenges, Gajland et al. [25] proposed SWOOSH, a Module-LWE–based NIKE protocol achieving CKS-light security in the random oracle model (ROM). Beyond standard security notions, SWOOSH additionally considers *semi-malicious correctness* [5], where an adversary, after obtaining the public parameters, may issue quantum queries to a quantum random oracle [11] and exploit quantum search techniques to find two distinct key pairs output by NGen algorithm (see Defination 9) that induce a correctness failure of the NIKE. This correctness-oriented adversarial model, which targets lattice-noise–induced failures, is strictly stronger than those typically adopted in existing lattice-based public-key cryptographic schemes. As such attacks are not commonly incorporated into standard correctness analyses, which usually assume honest key generation, we likewise restrict our attention to classical honest correctness.

We note that Hesse et al. [37] presented a NIKE construction that incorporates an LWE-based tagged NIKE component, but achieves full non-interactivity by combining it with a matrix DH–based non-interactive tag exchange. Consequently, the overall protocol relies on classical hardness assumptions and therefore does not provide post-quantum security. Moreover, the security of this construction is established in the v-semi-adaptive security model.

Tight Security Reductions in Lattice-Based NIKE. Tight reduction is crucial for ensuring both security and efficiency in cryptographic schemes by achieving security guarantees without excessively large parameters. While many public-key cryptographic schemes [27, 32, 37, 51] have successfully achieved tight reductions, this is more challenging for NIKE protocols. In NIKE, compromising one private key can expose shared keys, leading to a reduction loss factor lower-bounded by $O(\mu^2)$ [7] or $O(\mu)$ [36]. Currently, $\mu/2$ represents the optimal reduction bound achievable without increasing the public and private key lengths [6]. As discussed above, SWOOSH achieves CKS-light security with a tight reduction. However, transforming a CKS-light NIKE protocol to CKS-heavy security would incur an additional $\mu^2/2$ loss in reduction tightness. Consequently, constructing a lattice-based NIKE protocol that attains CKS-heavy security with a reduction loss approaching $O(\mu)$ remains a challenge.

Generic Constructions. Hesse et al. [36] proposed a generic NIKE framework based on hash proof system (HPS) [17], more precisely HPS for subset membership problem (SMP). Throughout the rest of the paper, we use HPS to mean an HPS for SMP. However, applying this framework to lattice-based NIKE protocols presents a significant challenge due to the

Table 1: Comparison of NIKE constructions with lattice-based or lattice-instantiable realizations.

Protocol	Model	Assumption	PQ	SL
SWOOSH [25]	ROM	Module-LWE	✓	$\mu^{2\dagger}$
Hesse et al. [36]	SM	MDDH‡	×	$\mu/2$
Ours	SM	Module-LWE	✓	μ

†: Under CKS-light security, SWOOSH admits a constant reduction loss of factor 2. When analyzed under CKS-heavy security for comparison, the corresponding reduction loss is μ^2. ‡: Hesse et al.'s construction is originally based on the Matrix Decisional Diffie-Hellman (MDDH) assumption.

inherent errors in LWE samples, which prevent the correct establishment of a shared key. In fact, several lattice-based HPS constructions have been proposed, all of which achieve only relaxed correctness. These include quasi-adaptive HPS (QA-HPS) [33], approximate HPS [10, 38], and probabilistic QA-HPS (pr-QA-HPS) [34], collectively referred to as relaxed HPS. A secure lattice-based NIKE protocol requires a standard HPS that ensures (perfect) correctness, making these relaxed constructions unsuitable. Although reconciliation mechanisms can be used to address errors introduced by LWE samples, known methods are usually interactive and therefore incompatible with NIKE. Allowing a super-polynomial modulus-to-noise ratio makes correctness easier, but at a high efficiency cost. Thus, building a secure and correct lattice-based NIKE with practical polynomial modulus-to-noise ratios remains a challenging problem.

Besides the challenge mentioned above, there is an increasing need to explore more generic constructions that align with current trends in cryptographic standardization. In particular, the ongoing PQC standardization efforts emphasize KEM-based designs for KE protocols [6, 43, 44]. However, the challenge with using standardized KEMs, such as CRYSTALS-Kyber, for NIKE is that their structure inherently involves interaction during the KE process. This interaction creates a barrier to directly applying KEMs for NIKE constructions. Therefore, it is worthwhile to investigate KEM-based frameworks that can facilitate efficient NIKE constructions while maintaining alignment with standardization trends.

1.2 Our Contribution

Prior lattice-based NIKE constructions under the CKS-heavy model either suffer loose reductions [25] or cannot be directly instantiated in the lattice setting due to the lack of supporting HPS [36]. Using KEMs to build NIKE has emerged as a natural approach for post-quantum settings, but existing lattice-based KEMs require the sender to perform encapsulation online using the recipient's public key and transmit the resulting ciphertext. These limitations motivate the need for NIKE constructions that achieve tight reductions, practical efficiency and standard-model security. In this work, we make the following contributions.

A Decoupled-KEM View of NIKE. We introduce a new abstraction, *decoupled key encapsulation mechanisms* (decoupled KEMs), which separate ciphertext generation from recipient-specific key derivation. Concretely, we formalize KEMs whose encapsulation algorithm can be decomposed into two phases: (i) a recipient-independent phase that outputs a precomputable ciphertext together with an internal state, and (ii) a recipient-dependent phase that derives the shared key once the recipient's public key is known.

Building on it, we present a generic NIKE construction from decoupled KEMs. The construction enables ciphertext precomputation and achieves non-interactivity by exploiting a simple asymmetry between users based on identity ordering. To formalize security, we define a multi-user security notion for decoupled KEMs, denoted multi-user IND-weak-CCA with corruption (MU-IND-wCCA$^{\text{Corr+}}$), and prove that any decoupled KEM satisfying this notion yields a NIKE protocol secure under CKS-heavy security. This result establishes decoupled KEMs as a principled and general foundation for constructing NIKE.

Constructing Decoupled KEMs from HPS. We show that HPS provide a natural and powerful tool for instantiating decoupled KEMs. In particular, we construct a decoupled KEM, denoted D-KEM$_{\mathsf{HPS}}$, from any HPS.

We prove that (i) the correctness of the underlying HPS directly implies the correctness of the resulting decoupled KEM, and (ii) the ε-smoothness of the HPS implies MU-IND-wCCA$^{\text{Corr+}}$ security, with a reduction loss linear in the number of users. As a result, our framework yields a new NIKE construction, denoted NIKE$_{\mathsf{HPS}}$.

A Lattice-based HPS and NIKE Instantiation. We present a concrete Module-LWE–based HPS that satisfies standard correctness and ε-smoothness in the standard model (SM), employing a carefully designed rounding mechanism. The construction admits a tight reduction to the Module-LWE assumption, which directly translates into tight security guarantees for both D-KEM$_{\mathsf{HPS}}$ and the resulting NIKE$_{\mathsf{HPS}}$.

Our NIKE construction is post-quantum secure, proven under CKS-heavy security in the SM, and achieves a reduction loss of μ. This bound is only a constant factor away from the near-optimal $\mu/2$ reduction loss achieved by Hesse et al. [36]. By comparison, the reduction loss of SWOOSH [25] is a factor of 2 under CKS-light security and suffers an additional μ^2 loss when lifted to CKS-heavy security. A comparison between our instantiation and existing NIKE constructions, including lattice-based and lattice-instantiable protocols, is summarized in Table 1, where PQ indicates post-quantum security and SL denotes security loss.

We also conduct a comprehensive experimental evaluation of correctness and efficiency. Across all parameter sets considered (Tables 2), NIKE$_{\mathsf{HPS}}$ achieves correctness error probabilities in the range 2^{-102} to 2^{-85}, which are practically acceptable in lattice-based cryptographic schemes [3]. Efficiency measurements at NIST security Level 1 show that our protocol significantly outperforms both SWOOSH and *Hesse-*

Lattice (the lattice-based instantiation of Hesse et al. [36] obtained using our HPS construction). Under 128-bit security with NTT-based acceleration, it achieves speedups of approximately 2.36× and 1.99× in shared-key derivation, respectively. Public–private key generation costs are comparable to SWOOSH and achieve an approximately 3.63× speedup over SWOOSH. Likewise, under 192-bit security with NTT-based acceleration, our shared-key derivation achieves speedups of about 1.78× over Hesse-Lattice and about 1.85× over SWOOSH under its parameter set L3-S1 (where SWOOSH provides only 145-bit security).

2 Preliminaries

2.1 Notations

Let λ be a security parameter. A function $\mathfrak{n} : \mathbb{N} \to \mathbb{R}$ is called *negligible*, written $\mathfrak{n}(\lambda)$, if for any positive polynomial $p(\cdot)$ there exists an integer N such that for all $n > N$, it holds that $\mathfrak{n}(n) < \frac{1}{p(n)}$. For $n \in \mathbb{N}$, we write $[n] = \{1, 2, \cdots, n\}$.

An algorithm $\mathcal{A}$ is a *probabilistic polynomial-time* (PPT) algorithm if its running time $T(\mathcal{A})$ is polynomial in λ. We write $y \leftarrow_\$ \mathcal{A}(x)$ to denote the output of $\mathcal{A}$ on input x, where $\mathcal{A}$ may use internal randomness. If $\mathcal{A}$ is deterministic, we write $y \leftarrow \mathcal{A}(x)$ for its (deterministic) output. For a finite set $\mathbb{S}$, $x \leftarrow_\$ \mathbb{S}$ denotes sampling x uniformly at random from $\mathbb{S}$. For a distribution χ, $x \leftarrow_\$ \chi$ denotes sampling according to χ.

For two probability distributions $\mathcal{X}$ and $\mathcal{Y}$ with the same support Ω, the statistical distance between them is defined as $\Delta(\mathcal{X}, \mathcal{Y}) = \frac{1}{2}\sum_{x\in\Omega} |\Pr[\mathcal{X} = x] - \Pr[\mathcal{Y} = x]|$. If $\Delta(\mathcal{X}, \mathcal{Y}) \leq \mathfrak{n}(\lambda)$, we say that $\mathcal{X}$ and $\mathcal{Y}$ are statistically indistinguishable and denote it by $\mathcal{X} \approx_s \mathcal{Y}$. If $\Delta(\mathcal{X}, \mathcal{Y}) \leq \varepsilon$, we denote $\mathcal{X} \equiv_\varepsilon \mathcal{Y}$ and say $\mathcal{X}$ and $\mathcal{Y}$ are ε-*close*. If $|\Pr[\mathcal{D}(\mathcal{X}) = 1] - \Pr[\mathcal{D}(\mathcal{Y}) = 1]| \leq \mathfrak{n}(\lambda)$ for all PPT distinguishers $\mathcal{D}$, we say $\mathcal{X}$ and $\mathcal{Y}$ are computationally distinguishable and denote it by $\mathcal{X} \approx_c \mathcal{Y}$.

This paper also involves modular multiplication and modular addition operations over matrices and vectors. Lowercase bold letters, such as $\mathbf{a} = (a_1, a_2, \cdots, a_m)^T \in \mathbb{R}^m$, denote column vectors, where a_i represents the i-th entry of $\mathbf{a}$. Uppercase bold letters, such as $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_n) \in \mathbb{R}^{m\times n}$, denote matrices, where $\mathbf{a}_i$ is the i-th column of $\mathbf{A}$. For a vector $\mathbf{a}$ or a matrix $\mathbf{A}$, we denote its transpose by $\mathbf{a}^T$ or $\mathbf{A}^T$, respectively. Boldface $\mathbf{0}$ denotes the all-zero vector.

2.2 Lattice Background

We adopt a lattice-based framework and rely on the Module-LWE assumption [14, 26, 50], where instances are naturally expressed as vectors of ring elements over a polynomial ring [41, 46], which supports efficient arithmetic.

Definition 1 (Cyclotomic Ring). *Let $k > 2$, $m = 2^k$, $N = \varphi(m) = 2^{k-1}$. Let $f(X) = \Phi_m(X)$ be the m-th cyclotomic polynomial. We have $f(X) = X^N + 1$. Define the cyclotomic integer ring of degree N as $\mathcal{R} = \mathbb{Z}[X]/(f(X))$. For an integer modulus $q \geq 2$ (typically prime), define $\mathcal{R}_q = \mathcal{R}/q\mathcal{R} \cong (\mathbb{Z}/q\mathbb{Z})[X]/(f(X))$, i.e., polynomials are reduced modulo $f(X)$ and coefficients are reduced modulo q.*

Elements in $\mathcal{R}_q$ can be expressed as $f = \sum_{i=1}^{N} a_i X^{i-1} \in \mathcal{R}_q$, where the coefficients $a_i \in [0, q-1]$. Let the ℓ_∞-norm for $f \in \mathcal{R}_q$ and $\mathbf{f} = (f_1, f_2, \cdots, f_m) \in \mathcal{R}_q^m$ be defined as

$$\|f\|_\infty = \max_{i\in[N]} |a_i|\,,\ \|\mathbf{f}\|_\infty = \max_{i\in[m]} \|f_i\|_\infty.$$

Definition 2 (Module-LWE Problem). *Let $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^N + 1)$ and let χ be an error distribution over $\mathcal{R}_q$. The Module-LWE problem is to distinguish between the following two distributions $(\mathbf{A}, \mathbf{A}^T\mathbf{s} + \mathbf{e})$ and $(\mathbf{A}, \mathbf{b})$, where $\mathbf{A} \leftarrow_\$ \mathcal{R}_q^{n\times m}$, $\mathbf{s} \leftarrow_\$ \chi^n$, $\mathbf{e} \leftarrow_\$ \chi^m$, and $\mathbf{b} \leftarrow_\$ \mathcal{R}_q^m$.*

Definition 3 (Module-LWE Assumption). *The Module-LWE assumption holds if, for any adversary $\mathcal{B}$, the following advantage, denoted as $\mathsf{Adv}^{\mathrm{MLWE}}_{N,q,n,m,\chi}(\mathcal{B})$, i.e., the advantage of $\mathcal{B}$ in solving the Module-LWE problem, is negligible in the security parameter λ :*

$$\mathsf{Adv}^{\mathrm{MLWE}}_{N,q,n,m,\chi}(\mathcal{B}) = \big|\Pr\left[\mathcal{B}(\mathbf{A}, \mathbf{A}^T \cdot \mathbf{s} + \mathbf{e}) = 1\right] - \Pr\left[\mathcal{B}(\mathbf{A}, \mathbf{b}) = 1\right]\big| \leq \mathfrak{n}(\lambda).$$

Theoretical analyses of Module-LWE constructions often assume the secret $\mathbf{s}$ and the noise $\mathbf{e}$ to be sampled either from a discrete Gaussian distribution [14, 26, 41, 45, 48] or from a centered binomial distribution (CBD) [13]. The latter is typically more efficient and easier to implement in a constant-time manner. Accordingly, we adopt the CBD and write $\mathbf{v} \leftarrow_\$ \mathsf{Cbd}(n, N, \beta)$ to denote sampling a vector $\mathbf{v} \in \mathcal{R}^n$ whose n components are polynomials of degree $< N$, and whose coefficients are independently drawn from a CBD with support $\{-\beta, \ldots, \beta\}$. When used in computations modulo q, we view $\mathbf{v}$ as an element of $\mathcal{R}_q^n$ via coefficient reduction. In particular, this sampling ensures $\|\mathbf{v}\|_\infty \leq \beta$.

2.3 Hash Proof Systems

Definition 4 (SMP). *An SMP, denoted as $\mathsf{SMP} = \mathsf{SSetup}$, consists of a PPT algorithm with the following properties.*

- *$spp \leftarrow_\$ \mathsf{SSetup}(1^\lambda)$: On input 1^λ, it selects a language parameter ρ and outputs a language description $spp = (X, L, R)$ based on ρ, where X is a set of instances, $L \subset X$ is a set of valid instances, and R is an efficiently computable relation such that for every $x \in L$, there exists a witness w satisfying $(x, w) \in R$.*

Additionally, for any (X, L, R) output by SSetup, it must be possible to efficiently sample: 1) elements x uniformly at random from $X \setminus L$, and 2) pairs (x, w) such that x is sampled uniformly at random from L and $(x, w) \in R$.

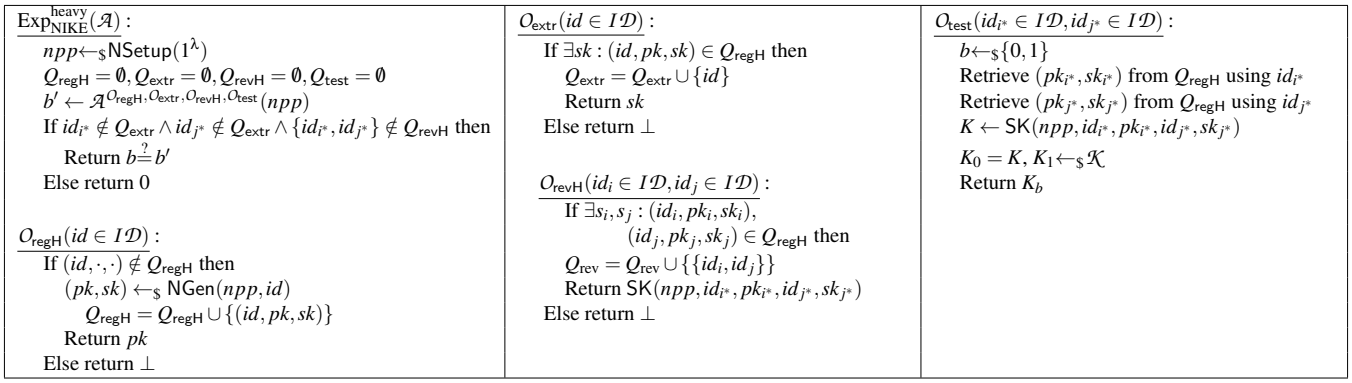

Figure 1: Security experiment for CKS-heavy secure NIKE. We use i, j to distinguish different users, where $i \neq j$. The symbol $\cdot$ in a tuple $(id, \cdot, \cdot)$ denotes an arbitrary value.

Definition 5 (SMP Assumption [34, 36]). *The SMP assumption holds if, for any adversary $\mathcal{B}$, the following advantage, denoted as $\mathsf{Adv}^{\mathrm{SMP}}(\mathcal{B})$, i.e., the advantage of $\mathcal{B}$ in distinguishing the two distributions x and x', is negligible in λ :*

$$\mathsf{Adv}^{\mathrm{SMP}}(\mathcal{B}) = \left|\Pr\left[\mathcal{B}(spp, x) = 1\right] - \Pr\left[\mathcal{B}(spp, x') = 1\right]\right| \leq \mathfrak{n}(\lambda),$$

where $spp = (X, L, R) \leftarrow_{\$} \mathsf{SSetup}(1^\lambda)$, $x \leftarrow_{\$} L$, and $x' \leftarrow_{\$} X \setminus L$.

Definition 6 (HPS [32, 34]). *An HPS, denoted as $\mathsf{HPS} = (\mathsf{HSetup}, \alpha_{(\cdot)}, \mathsf{Pub}, \mathsf{Priv})$, consists of the following algorithms.*

- *$hpp \leftarrow_{\$} \mathsf{HSetup}(1^\lambda)$: On input 1^λ, it runs $\mathsf{SSetup}(1^\lambda)$ to generate a language description $spp = (X, L, R)$ based on a language parameter ρ. Then selects a hash key space $\mathcal{HSK}$ from which keys can be sampled efficiently and uniformly at random, a projection key space $\mathcal{HPK}$, a hash value space Π such that there exists a family of hash functions $\mathcal{H} = \{h_{hsk} : X \to \Pi \mid hsk \in \mathcal{HSK}\}$; finally, outputs system parameters $hpp = (spp, \mathcal{HSK}, \mathcal{HPK}, \Pi, \mathcal{H}, \rho)$.*
- *$hpk \leftarrow \alpha_\rho(hpp, hsk)$: α_ρ is an efficiently computable mapping algorithm indexed by ρ. On input hpp and a hash key $hsk \in \mathcal{HSK}$, it outputs a projection key $hpk \in \mathcal{HPK}$.*
- *$\pi \leftarrow \mathsf{Pub}(hpp, hpk, x, w)$: On input hpp, $hpk = \alpha_\rho(hsk)$ and a pair $(x, w) \in R$, the public evaluation algorithm outputs a hash value $\pi \in \Pi$.*
- *$\pi \leftarrow \mathsf{Priv}(hpp, hsk, x)$: On input hpp, hsk and an instance $x \in X$, the private evaluation algorithm outputs a hash value $\pi = h_{hsk}(x) \in \Pi$ using the hash key hsk.*

Definition 7 (Correctness of HPS). *An HPS scheme $\mathsf{HPS} = (\mathsf{HSetup}, \alpha_{(\cdot)}, \mathsf{Pub}, \mathsf{Priv})$ is said to be correct if, for any $hpp = (spp, \mathcal{HSK}, \mathcal{HPK}, \Pi, \mathcal{H}, \rho) \leftarrow_{\$} \mathsf{HSetup}(1^\lambda)$, the hash values computed by Pub and Priv agree except with negligible probability. That is,*

$$\Pr\left[\pi \neq \pi' \;\middle|\; \begin{array}{l} \pi = \mathsf{Pub}(hpp, hpk, x, w) \\ \pi' = \mathsf{Priv}(hpp, hsk, x) \end{array}\right] \leq \mathfrak{n}(\lambda),$$

where $hsk \leftarrow_{\$} \mathcal{HSK}$, $hpk \leftarrow \alpha_\rho(hsk)$ and $(x, w) \in R$.

Definition 8 (Smoothness). *An HPS scheme $\mathsf{HPS} = (\mathsf{HSetup}, \alpha_{(\cdot)}, \mathsf{Pub}, \mathsf{Priv})$ for SMP is said to be ε-smooth if for any $hpp = (spp, \mathcal{HSK}, \mathcal{HPK}, \Pi, \mathcal{H}, \rho) \leftarrow_{\$} \mathsf{HSetup}(1^\lambda)$, the statistical difference between the two distributions $(x, hpk, \mathsf{Priv}(hpp, hsk, x))$ and (x, hpk, π) is at most ε, such that $\{(x, hpk, \mathsf{Priv}(hpp, hsk, x))\} \equiv_\varepsilon \{(x, hpk, \pi)\}$, where $hsk \leftarrow_{\$} \mathcal{HSK}$, $hpk \leftarrow \alpha_\rho(hpp, hsk)$, $x \leftarrow_{\$} X \setminus L$ and $\pi \leftarrow_{\$} \Pi$.*

2.4 Non-Interactive Key Exchange

Definition 9 (NIKE). *A NIKE protocol, denoted as $\mathsf{NIKE} = (\mathsf{NSetup}, \mathsf{NGen}, \mathsf{SharedKey})$, with identity space $\mathcal{ID}$ and shared key space $\mathcal{K}$, consists of the following algorithms.*

- *$npp \leftarrow_{\$} \mathsf{NSetup}(1^\lambda)$: On input 1^λ, it outputs the system parameters npp.*
- *$(pk_i, sk_i) \leftarrow_{\$} \mathsf{NGen}(npp, id_i)$: On input npp and a user's identity id_i, it outputs a public-private key pair (pk_i, sk_i).*
- *$K_{ij} \leftarrow \mathsf{SK}(npp, id_i, pk_i, id_j, sk_j)$: It is run by a user with identity id_j. On input npp, another user's identity id_i and corresponding public key pk_i, and the user's own identity id_j and corresponding private key sk_j, it outputs a shared key $K_{ij} \in \mathcal{K}$, or $\perp$ if $id_i = id_j$.*

Definition 10 (Correctness of NIKE). *A NIKE protocol, denoted by $\mathsf{NIKE} = (\mathsf{NSetup}, \mathsf{NGen}, \mathsf{SK})$, is said to be* correct *if, for any $npp \leftarrow_{\$} \mathsf{NSetup}(1^\lambda)$, and for any two users with identities id_i and id_j and public-private key pairs $(pk_i, sk_i) \leftarrow_{\$}$*

$\mathsf{NGen}(npp, id_i)$, $(pk_j, sk_j) \leftarrow_{\$} \mathsf{NGen}(npp, id_j)$, *the shared keys computed by the two users agree except with negligible probability. That is, the correctness error probability is*

$$\Pr\left[K_{ij} \neq K_{ji} \;\middle|\; \begin{array}{l} K_{ij} = \mathsf{SK}(npp, id_i, pk_i, id_j, sk_j) \\ K_{ji} = \mathsf{SK}(npp, id_j, pk_j, id_i, sk_i) \end{array}\right] \leq \mathfrak{n}(\lambda).$$

Security Model for NIKE. Security of NIKE protocols is typically formalized under the CKS family of models, most notably CKS-light, CKS-heavy, and *m*-CKS-heavy, which differ in the number and types of oracle queries available to the adversary. Although these variants are polynomially equivalent [24], they provide different levels of adversarial flexibility and query adaptivity. In this work, we consider the HKR formulation of the CKS-heavy security model, for which standard transformations to the corresponding DKR variant are known. For brevity, we refer to the HKR CKS-heavy model simply as CKS-heavy.

Figure 1 defines the formal security experiment $\mathrm{Exp}^{\mathrm{heavy}}_{\mathrm{NIKE}}(\mathcal{A})$ for the CKS-heavy model. In this experiment, an adversary $\mathcal{A}$ is allowed to adaptively issue multiple queries to the oracles O_{regH}, O_{extr}, and O_{revH}, as well as a single query to the O_{test} oracle. We note that a valid O_{test} query requires that the involved identities have previously been submitted to O_{regH}. If this has not already been done, it is assumed that the challenger will perform the necessary O_{regH} queries first. At the end of the experiment, $\mathcal{A}$ outputs a bit b' as its guess for the challenge bit b. If $b' = b$ and a specified winning condition $id_{i^*} \notin Q_{\mathsf{extr}} \wedge id_{j^*} \notin Q_{\mathsf{extr}} \wedge \{id_{i^*}, id_{j^*}\} \notin Q_{\mathsf{revH}}$ is satisfied, then $\mathcal{A}$ is said to win the experiment. The advantage that $\mathcal{A}$ wins is denoted by $\mathsf{Adv}^{\mathrm{heavy}}_{\mathrm{NIKE}}(\mathcal{A})$.

Definition 11 (CKS-heavy Security)**.** *A NIKE protocol* $\mathsf{NIKE} = (\mathsf{NSetup}, \mathsf{NGen}, \mathsf{SK})$ *is said to be CKS-heavy secure if* $\mathsf{Adv}^{\mathrm{heavy}}_{\mathrm{NIKE}}(\mathcal{A})$ *is negligible, i.e.,*

$$\mathsf{Adv}^{\mathrm{heavy}}_{\mathrm{NIKE}}(\mathcal{A}) = \left|\Pr[\mathrm{Exp}^{\mathrm{heavy}}_{\mathrm{NIKE}}(\mathcal{A}) \Rightarrow 1] - \tfrac{1}{2}\right| \leq \mathfrak{n}(\lambda).$$

3 Generic Construction of NIKE from Decoupled KEM

We introduce the notion of decoupled KEM and leverage it as a building block for constructing NIKE protocols.

3.1 Definition of Decoupled KEM

A decoupled KEM scheme, denoted as $\mathsf{D\text{-}KEM} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$, with identity space $\mathcal{ID}$ and shared key space $\mathcal{K}$, consists of the following algorithms.

- $kpp \leftarrow_{\$} \mathsf{KSetup}(1^\lambda)$: On input 1^λ, it outputs the system parameters kpp.
- $(p, s) \leftarrow_{\$} \mathsf{KGen}(kpp, id)$: On input kpp and a user's identity $id \in \mathcal{ID}$, it outputs a public-private key pair (p, s).
- $(C, K) \leftarrow_{\$} \mathsf{Encap}(kpp, p)$: On input kpp and a public key p, this algorithm proceeds in two stages, invoking two sub-algorithms sequentially:
  1. $(C, r) \leftarrow_{\$} \mathsf{Encap}_1(kpp)$: On input kpp, this first-stage algorithm outputs a ciphertext C and an internal secret r.
  2. $K \leftarrow \mathsf{Encap}_2(kpp, p, C, r)$: On input kpp, p, C and r generated in Encap_1, this second-stage algorithm outputs a shared key $K \in \mathcal{K}$.
- $K \leftarrow \mathsf{Decap}(kpp, C, s)$: On input kpp, C and s, it outputs a shared key $K \in \mathcal{K}$.

Definition 12 (Correctness of Decoupled KEM)**.** *A decoupled KEM scheme* $\mathsf{D\text{-}KEM} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$ *is said to be correct if, for any* $kpp \leftarrow_{\$} \mathsf{KSetup}(1^\lambda)$*, and for any public-private key pair* $(p, s) \leftarrow_{\$} \mathsf{KGen}(kpp, id)$*, the shared keys computed by* Encap *and* Decap *are identical with all but negligible probability. That is,*

$$\Pr\left[K \neq K' \;\middle|\; \begin{array}{l} (C, K) = \mathsf{Encap}(kpp, p) \\ K' = \mathsf{Decap}(C, s) \end{array}\right] \leq \mathfrak{n}(\lambda).$$

Security Model for Decoupled KEM. The standard notion of indistinguishability under chosen-ciphertext attack (IND-CCA) for KEMs is defined in a single-user, single-challenge setting. This notion guarantees that, given a challenge ciphertext and a challenge key, an adversary with access to a decapsulation oracle cannot distinguish whether the challenge key is the actual shared key encapsulated in the ciphertext or a uniformly random key. However, the widely accepted security models for KE protocols are fundamentally multi-user and permit adversaries to corrupt the private keys of certain users. Consequently, an IND-CCA-secure KEM is insufficient for constructing KE protocols that are secure in such a setting.

To address this, a natural extension is the Multi-User IND-CCA model with corruption queries under the single-challenge setting, denoted as MU-IND-CCA^{Corr}. This model allows the adversary to interact with $\mu \geq 1$ users, corrupt a subset of them, and issue a single challenge query. In the challenge query, the adversary receives a challenge ciphertext and a challenge key, and must distinguish whether the key is the actual encapsulated shared key or a random value. The MU-IND-CCA^{Corr} model supports corruption and challenge capabilities as those considered in the CKS-heavy setting, including multiple reveal queries per user, which correspond to decapsulation oracle queries in MU-IND-CCA^{Corr}. However, security in the standard MU-IND-CCA^{Corr} model is insufficient for constructing a NIKE protocol secure under CKS-heavy. This insufficiency stems from two factors. First, in the CKS-heavy model the adversary may only register honest users and reveal session keys established between honest parties. Consequently, the corresponding decapsulation queries are restricted to well-formed ciphertexts derived from honest protocol executions, and the adversary is not permitted to

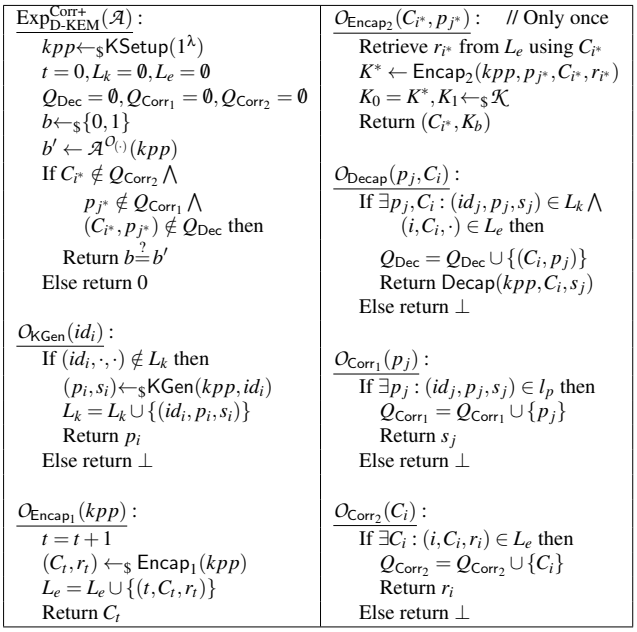

Figure 2: Security experiment for MU-IND-wCCA$^{\mathrm{Corr+}}$ secure decoupled KEM. We use $O_{(\cdot)}$ to denote the adversary's access to the oracles $O_{\mathsf{KGen}}, O_{\mathsf{Encap}_1}, O_{\mathsf{Encap}_2}, O_{\mathsf{Decap}}, O_{\mathsf{Corr}_1}$.

submit malformed ciphertexts. Second, the unique structure of our decoupled KEM requires a stronger or more tailored security model.

To enable the reduction from NIKE to the underlying decoupled KEM, we introduce a tailored variant, denoted as MU-IND-wCCA$^{\mathrm{Corr+}}$. This model is formalized through a security experiment $\mathrm{Exp}^{\mathrm{Corr+}}_{\text{D-KEM}}(\mathcal{A})$, as illustrated in Figure 2. In this experiment, an adversary $\mathcal{A}$ is allowed to adaptively issue multiple queries to the oracles O_{KGen}, O_{Encap_1}, O_{Decap}, O_{Corr_1} and O_{Corr_2}, as well as a single query to the O_{Encap_2} oracle. We note that a valid O_{Encap_2} query requires that the involved ciphertext and public key have been generated by O_{Encap_1} and O_{KGen}, respectively. At the end of the experiment, $\mathcal{A}$ outputs a bit b' as its guess for the challenge bit b. If $b'=b$ and a specified winning condition $C_i \notin Q_{\mathsf{Corr}_2} \wedge p_j \notin Q_{\mathsf{Corr}_1} \wedge (C_i, p_j) \notin Q_{\mathsf{Dec}}$ is satisfied, then $\mathcal{A}$ is said to win the experiment. The advantage of $\mathcal{A}$ in winning this experiment is denoted as $\mathsf{Adv}^{\mathrm{Corr+}}_{\text{D-KEM}}(\mathcal{A})$.

Definition 13 (MU-IND-wCCA$^{\mathrm{Corr+}}$ Security)**.** *A decoupled KEM scheme* $\mathsf{D\text{-}KEM} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$ *is said to be* MU-IND-wCCA$^{\mathrm{Corr+}}$ *secure if* $\mathsf{Adv}^{\mathrm{Corr+}}_{\text{D-KEM}}(\mathcal{A})$ *is negligible, i.e.,*

$$\mathsf{Adv}^{\mathrm{Corr+}}_{\text{D-KEM}}(\mathcal{A}) = \left|\Pr[\mathrm{Exp}^{\mathrm{Corr+}}_{\text{D-KEM}}(\mathcal{A}) \Rightarrow 1] - \tfrac{1}{2}\right| \leq \mathfrak{n}(\lambda).$$

Compared to the standard MU-IND-CCA^{Corr} model (see Appendix A), where the adversary may decapsulate arbitrary ciphertexts via a decapsulation oracle and obtain a user's private key via a corruption oracle, our model restricts the decapsulation oracle to ciphertexts generated by the challenger. This restriction yields a weaker form of CCA security, which nevertheless suffices for proving the security of NIKE under the CKS-heavy setting. In addition, we retain the corruption functionality through the oracle O_{Corr_1} and further strengthens the adversary's capability by introducing an additional oracle, O_{Corr_2}. This new oracle grants access to the internal secret associated with a selected ciphertext. This extension clearly results in a strictly stronger adversarial model. The inclusion of O_{Corr_1} and O_{Corr_2} allows us to accurately simulate the corruption oracle O_{Extr} in the CKS-heavy security model for NIKE.

Another key difference between MU-IND-CCA^{Corr} model and our MU-IND-wCCA$^{\mathrm{Corr+}}$ model lies in how a challenge ciphertext and challenge key are generated. In MU-IND-CCA^{Corr}, a single oracle O_{Encap} is used to produce both. In contrast, our model achieves the same functionality by splitting the process across two oracles: O_{Encap_1} and O_{Encap_2}. Specifically, we grant the adversary the ability to query O_{Encap_1} and O_{Encap_2} separately. When invoked sequentially, these oracles are functionally equivalent to a single call to O_{Encap} in the MU-IND-CCA^{Corr} model. This separation is essential to ensure compatibility with the NIKE security proof: O_{Encap_1} is used to simulate part of a user's public and private key components, whereas O_{Encap_2} is responsible for simulating responses to the O_{test} oracle in CKS-heavy.

3.2 The Generic Construction

Let $\mathsf{D\text{-}KEM} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$ be a decoupled KEM scheme with identity space $\mathcal{ID}$ and shared key space $\mathcal{K}$. The construction of a NIKE protocol $\mathsf{NIKE} = (\mathsf{NSetup}, \mathsf{NGen}, \mathsf{SK})$ with identity space $\mathcal{ID}$ and shared key space $\mathcal{K}$ from D-KEM proceeds as follows.

- $npp \leftarrow_{\$} \mathsf{NSetup}(1^\lambda)$: On input 1^λ, it runs $\mathsf{KSetup}(1^\lambda)$ to generate the parameters kpp, which are directly used as the NIKE system parameters npp.
- $(pk, sk) \leftarrow_{\$} \mathsf{NGen}(npp, id)$: On input npp and a user's identity $id \in \mathcal{ID}$, it executes $\mathsf{KGen}(npp, id)$ to generate the key pair (p, s) corresponding to the D-KEM scheme, invokes $\mathsf{Encap}_1(npp)$ to obtain the ciphertext C and internal secret r, and sets the public key of the user to be $pk = (p, C)$ and the private key to be $sk = (s, C, r)$[1].
- $K_{ij} \leftarrow \mathsf{SK}(npp, id_i, pk_i, id_j, sk_j)$: On input npp, another user's identity $id_i \in \mathcal{ID}$ and public key $pk_i = (p_i, C_i)$, and the user's own identity $id_j \in \mathcal{ID}$ and private key $sk_j = (s_j, C_j, r_j)$, the algorithm proceeds as follows:
  - If $id_i < id_j$, run $\mathsf{Encap}_2(npp, p_i, C_j, r_j)$ to compute the shared key $K_{i,j} \in \mathcal{K}$.

[1] C is included in the private key as well, since the correctness of r depends on its structural relation to C.

– Else run $\mathsf{Decap}(npp, C_i, s_j)$ to derive the shared key $K_{ij} \in \mathcal{K}$.

Theorem 1. *If* $\mathsf{D\text{-}KEM} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$ *is correct, then the derived NIKE protocol* $\mathsf{NIKE} = (\mathsf{NSetup}, \mathsf{NGen}, \mathsf{SK})$ *is also correct.*

For the complete proof, refer to Appendix B.1.

Theorem 2. *Let* $\mathsf{NIKE} = (\mathsf{NSetup}, \mathsf{NGen}, \mathsf{SK})$ *be a decoupled KEM-based NIKE. Let* $\mathcal{A}_1$ *be an adversary that breaks the CKS-heavy security of* NIKE *with advantage* $\mathsf{Adv}^{\text{heavy}}_{\text{NIKE}}(\mathcal{A}_1)$. *Then, there exists an adversary* $\mathcal{A}_2$, *which invokes* $\mathcal{A}_1$ *as a subroutine and breaks the* MU-IND-wCCA$^{\text{Corr+}}$ *security of the underlying decoupled KEM scheme* D-KEM *with advantage* $\mathsf{Adv}^{\text{Corr+}}_{\text{D-KEM}}(\mathcal{A}_2)$, *such that* $\mathsf{Adv}^{\text{heavy}}_{\text{NIKE}}(\mathcal{A}_1) \leq \mathsf{Adv}^{\text{Corr+}}_{\text{D-KEM}}(\mathcal{A}_2)$. *The running time of* $\mathcal{A}_2$ *is almost the same as that of* $\mathcal{A}_1$.

Proof. To establish the security of the NIKE protocol, we construct an adversary $\mathcal{A}_2$ that interacts with the challenger $\mathcal{C}$ in the experiment $\text{Exp}^{\text{Corr+}}_{\text{D-KEM}}$ for the underlying decoupled KEM scheme, while faithfully simulating the experiment $\text{Exp}^{\text{heavy}}_{\text{NIKE}}$ for $\mathcal{A}_1$. By leveraging the success of $\mathcal{A}_1$, $\mathcal{A}_2$ attempts to break the security of the decoupled KEM.

Let $b \in \{0,1\}$ be the challenge bit selected by $\mathcal{C}$ in the experiment $\text{Exp}^{\text{Corr+}}_{\text{D-KEM}}$ for the underlying decoupled KEM scheme. Upon receiving the system parameters kpp from $\mathcal{C}$, $\mathcal{A}_2$ initializes the simulation environment for $\mathcal{A}_1$ as follows: $\mathcal{A}_2$ sets the system parameters of the NIKE protocol as $npp = kpp$, then gives npp to $\mathcal{A}_1$. $\mathcal{A}_2$ also maintains three initially empty sets: Q_{regH}, Q_{extr}, and Q_{revH}. For every oracle query issued by $\mathcal{A}_1$, $\mathcal{A}_2$ simulates the response using its access to $\mathcal{C}$'s oracles as follows:

$O_{\mathsf{regH}}(id)$ query: If a tuple (id, p, C) already exists in Q_{regH} for identity id, $\mathcal{A}_2$ returns $\perp$. Otherwise, $\mathcal{A}_2$ issues $O_{\mathsf{KGen}}(kpp, id)$ and $O_{\mathsf{Encap}_1}(kpp)$ to $\mathcal{C}$ to obtain (p, C), adds (id, p, C) to Q_{regH}, and returns (p, C).

$O_{\mathsf{extr}}(id)$ query: If no tuple (id, p, C) exists in Q_{regH} for identity id, $\mathcal{A}_2$ returns $\perp$. Otherwise, $\mathcal{A}_2$ issues $O_{\mathsf{Corr}_1}(p)$ and $O_{\mathsf{Corr}_2}(C)$ to $\mathcal{C}$ to obtain (s, r), adds id to Q_{extr}, and returns (s, r).

$O_{\mathsf{revH}}(id_i, id_j)$ query: If either (id_i, p_i, C_i) or (id_j, p_j, C_j) does not exist in Q_{regH}, $\mathcal{A}_2$ returns $\perp$. If $id_i < id_j$, $\mathcal{A}_2$ queries $O_{\mathsf{Decap}}(p_i, C_j)$ to $\mathcal{C}$ to get K, adds $\{id_i, id_j\}$ to Q_{revH} and returns K. Else, $\mathcal{A}_2$ queries $O_{\mathsf{Decap}}(p_j, C_i)$ to $\mathcal{C}$ to get K, adds $\{id_i, id_j\}$ to Q_{revH} and returns K.

$O_{\mathsf{test}}(id_{i^*}, id_{j^*})$ query: If $id_{i^*} < id_{j^*}$, $\mathcal{A}_2$ queries $O_{\mathsf{Encap}_2}(C_{j^*}, p_{i^*})$ to $\mathcal{C}$ to get K_b and returns it. Else, $\mathcal{A}_2$ queries $O_{\mathsf{Encap}_2}(C_{i^*}, p_{j^*})$ to get K_b and returns it.

Finally, $\mathcal{A}_1$ outputs its guess b', which $\mathcal{A}_2$ forwards to $\mathcal{C}$ as his own guess. $\mathcal{C}$ then outputs the result $b \stackrel{?}{=} b'$, and $\mathcal{A}_2$ adopts this as his final output as well.

During the above simulation, if $\mathcal{A}_2$ does not abort, then he perfectly simulates the experiment $\text{Exp}^{\text{heavy}}_{\text{NIKE}}$ for $\mathcal{A}_1$. Notably, by construction, if $\mathcal{C}$ does not abort, then $\mathcal{A}_2$ will not abort either. Therefore, whenever $\mathcal{A}_1$ wins and $\mathcal{C}$ does not abort, $\mathcal{A}_2$ also wins. It follows that

$$\begin{aligned}\mathsf{Adv}^{\text{Corr+}}_{\text{D-KEM}}(\mathcal{A}_2) &= \left|\Pr\left[\text{Exp}^{\text{Corr+}}_{\text{D-KEM}}(\mathcal{A}_2) \Rightarrow 1\right] - \tfrac{1}{2}\right| \\ &\geq \left|\Pr\left[\text{Exp}^{\text{heavy}}_{\text{NIKE}}(\mathcal{A}_1) \Rightarrow 1\right] - \tfrac{1}{2}\right| \\ &= \mathsf{Adv}^{\text{heavy}}_{\text{NIKE}}(\mathcal{A}_1).\end{aligned}$$

□

4 Generic Construction of D-KEM from HPS

We construct a decoupled KEM from an HPS, and formalize a generic framework for HPS-based decoupled KEMs, denoted $\mathsf{D\text{-}KEM}_{\mathsf{HPS}}$.

4.1 The Construction of $\mathsf{D\text{-}KEM}_{\mathsf{HPS}}$

Let $\mathsf{HPS} = (\mathsf{HSetup}, \alpha_{(\cdot)}, \mathsf{Pub}, \mathsf{Priv})$ be a ε-smooth HPS associated with an SMP. We construct a decoupled KEM scheme, denoted by $\mathsf{D\text{-}KEM}_{\mathsf{HPS}} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$, with identity space $\mathcal{ID}$ and shared key space $\mathcal{K}$, built upon the underlying HPS as follows:

- $kpp \leftarrow_{\$} \mathsf{KSetup}(1^\lambda)$: On input 1^λ, it first runs $\mathsf{HSetup}(1^\lambda)$ to generate $hpp = (X, L, R, \mathcal{HSK}, \mathcal{HPK}, \Pi, \mathcal{H}, \rho)$, then selects an identity space $\mathcal{ID} = \{0,1\}^*$ and sets shared key space $\mathcal{K} = \Pi$. Finally, it outputs $kpp = \{X, L, R, \mathcal{HSK}, \mathcal{HPK}, \mathcal{H}, \rho, \mathcal{ID}, \mathcal{K}\}$.
- $(p, s) \leftarrow_{\$} \mathsf{KGen}(kpp, id)$: On input kpp and a user's identity $id \in \mathcal{ID}$, it selects $s \leftarrow_{\$} \mathcal{HSK}$, generates $p = \alpha_\rho(kpp, s)$, and outputs (p, s) for the user.
- $(C, K) \leftarrow_{\$} \mathsf{Encap}(kpp, p)$: On input kpp and p, it proceeds the following two stages by invoking two sub-algorithms sequentially.
  1. $(C, r) \leftarrow_{\$} \mathsf{Encap}_1(kpp)$: On input kpp, this first-stage algorithm samples a pair $(x, w) \in R$, sets $C = x$, $r = w$ and outputs (C, r).
  2. $K \leftarrow \mathsf{Encap}_2(kpp, p, C, r)$: On input kpp, p, C and r, this second-stage algorithm runs $\mathsf{Pub}(kpp, p, C, r)$ to generate a hash value $\pi \in \mathcal{K}$, and outputs $K = \pi$.
- $K \leftarrow \mathsf{Decap}(kpp, C, s)$: On input kpp, C and s, it runs $\mathsf{Priv}(kpp, s, C)$ to generate a hash value $\pi \in \mathcal{K}$, and outputs $K = \pi$.

Theorem 3. *If* $\mathsf{HPS} = (\mathsf{HSetup}, \alpha_{(\cdot)}, \mathsf{Pub}, \mathsf{Priv})$ *is correct, then the derived decoupled KEM scheme* $\mathsf{D\text{-}KEM}_{\mathsf{HPS}} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$ *is also correct.*

The proof of Theorem 3 is in Appendix B.2.

Theorem 4. *Let* $\mathsf{HPS} = (\mathsf{HSetup}, \alpha_{(\cdot)}, \mathsf{Pub}, \mathsf{Priv})$ *be an* ε*-smooth HPS associated with an SMP* $\mathsf{SMP} = \mathsf{SSetup}$*, where* ε *is negligible in* λ*. If the* SMP *assumption holds for* SMP*, then the decoupled KEM scheme* $\mathsf{D\text{-}KEM}_{\mathsf{HPS}} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$ *described in Section 4.1 is* $\text{MU-IND-wCCA}^{\text{Corr+}}$ *secure. More precisely, for any adversary* $\mathcal{A}$ *that makes at most* μ *queries to the oracle* O_{Encap_1} *and achieves advantage* $\mathsf{Adv}^{\text{Corr+}}_{\mathsf{D\text{-}KEM}_{\mathsf{HPS}}}(\mathcal{A})$ *in the experiment* $\mathsf{Exp}^{\text{Corr+}}_{\mathsf{D\text{-}KEM}_{\mathsf{HPS}}}$*, we have* $\mathsf{Adv}^{\text{Corr+}}_{\mathsf{D\text{-}KEM}_{\mathsf{HPS}}}(\mathcal{A}) \le \mu \cdot \left(\mathsf{Adv}^{\text{SMP}}(\mathcal{B}) + \varepsilon\right)$*, where* $\mathsf{Adv}^{\text{SMP}}(B)$ *is the advantage defined in Definition 5.*

We prove the $\text{MU-IND-wCCA}^{\text{Corr+}}$ security of our $\mathsf{D\text{-}KEM}_{\mathsf{HPS}}$ scheme through a sequence of games. A detailed proof is deferred to Appendix B.3

Theorem 5. *Let* $\mathsf{HPS} = (\mathsf{HSetup}, \alpha_{(\cdot)}, \mathsf{Pub}, \mathsf{Priv})$ *be an* ε*-smooth HPS associated with an SMP* $\mathsf{SMP} = \mathsf{SSetup}$*, where* ε *is negligible. Let* $\mathsf{D\text{-}KEM}_{\mathsf{HPS}}$ *be the decoupled KEM constructed from the HPS as in Section 4.1, and let* $\mathsf{NIKE}_{\mathsf{HPS}} = (\mathsf{NSetup}, \mathsf{NGen}, \mathsf{SK})$ *be the NIKE protocol obtained from* $\mathsf{D\text{-}KEM}_{\mathsf{HPS}}$ *via the generic transformation in Section 3.2. If the SMP assumption holds for* SMP*, then for any adversary* $\mathcal{A}_1$ *that makes at most* μ *queries to the oracle* O_{regH} *and breaks the CKS-heavy security of* $\mathsf{NIKE}_{\mathsf{HPS}}$ *with advantage* $\mathsf{Adv}^{\text{heavy}}_{\mathsf{NIKE}_{\mathsf{HPS}}}(\mathcal{A}_1)$*, there exists an adversary* $\mathcal{A}_2$ *against the* $\text{MU-IND-wCCA}^{\text{Corr+}}$ *security of* $\mathsf{D\text{-}KEM}_{\mathsf{HPS}}$ *such that their advantages satisfy* $\mathsf{Adv}^{\text{heavy}}_{\mathsf{NIKE}_{\mathsf{HPS}}}(\mathcal{A}_1) \le \mathsf{Adv}^{\text{Corr+}}_{\mathsf{D\text{-}KEM}_{\mathsf{HPS}}}(\mathcal{A}_2) \le \mu \cdot \left(\mathsf{Adv}^{\text{SMP}}(\mathcal{B}) + \varepsilon\right)$*. The running time of* $\mathcal{A}_1$ *is almost the same as that of* $\mathcal{A}_2$*.*

The proof is a direct composition of Theorems 2 and 4.

5 Instantiation of NIKE from Lattices in the Standard Model

We first present a concrete lattice-based instantiation of an HPS, whose security relies on the hardness of the Module-LWE problem. Building on this HPS, we further instantiate $\mathsf{D\text{-}KEM}_{\mathsf{HPS}}$ and subsequently $\mathsf{NIKE}_{\mathsf{HPS}}$.

5.1 HPS from Lattices

Module-LWE-based SMP. We begin by instantiating an SMP. At a high level, the security of the resulting SMP follows from the hardness of Module-LWE problem: distinguishing valid SMP instances from random ones is equivalent to distinguishing Module-LWE samples from uniform.

The resulting SMP, denoted by $\mathsf{SMP} = \mathsf{SSetup}$, is defined as follows:

- $spp \leftarrow_{\$} \mathsf{SSetup}(1^\lambda)$: On input 1^λ, it first selects $N = 2^{k-1}$ for $k = O(\lambda)$ with $k > 2$, integers $n, m > 0$, a prime number q and an integer $\beta_1 > 0$, thereby fixing a CBD χ_{β_1} over the ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^N+1)$. Next, it samples $\mathbf{A} \leftarrow_{\$} \mathcal{R}_q^{n \times m}$ as the language parameter, and then defines a language description as follows:

$$X = \mathcal{R}_q^m,$$
$$L = \{\mathbf{x} \in \mathcal{R}_q^m \mid \exists \mathbf{r} \in \chi_{\beta_1}^n, \mathbf{e} \in \chi_{\beta_1}^m, \mathbf{x}^T = \mathbf{r}^T \cdot \mathbf{A} + \mathbf{e}^T\},$$
$$R = \{(\mathbf{x}, (\mathbf{r}, \mathbf{e})) \in \mathcal{R}_q^m \times (\chi_{\beta_1}^n \times \chi_{\beta_1}^m) \mid \mathbf{x}^T = \mathbf{r}^T \cdot \mathbf{A} + \mathbf{e}^T\}.$$

  Finally, it outputs $spp = (X, L, R)$.

Theorem 6. *Let* $\mathcal{B}_1$ *be any PPT adversary against the SMP* $\mathsf{SMP} = \mathsf{SSetup}$ *instantiated as above. Then there exists a PPT adversary* $\mathcal{B}_2$ *against the Module-LWE problem such that* $\mathsf{Adv}^{\text{SMP}}(\mathcal{B}_1) \le \mathsf{Adv}^{\text{MLWE}}_{N,q,n,m,\chi_{\beta_1}}(\mathcal{B}_2)$*, where* $\mathcal{B}_2$ *runs in essentially the same time as* $\mathcal{B}_1$*.*

Proof. Adversary $\mathcal{B}_2$ is given a Module-LWE challenge $(\mathbf{A}, \mathbf{b})$, where $\mathbf{A} \in \mathcal{R}_q^{n \times m}$ is uniformly random and $\mathbf{b} \in \mathcal{R}_q^m$ is either uniformly random or of the form $\mathbf{b} = \mathbf{A}^T\mathbf{s} + \mathbf{e}$ for $\mathbf{s} \leftarrow_{\$} \chi_{\beta_1}^n$ and $\mathbf{e} \leftarrow_{\$} \chi_{\beta_1}^m$. $\mathcal{B}_2$ sets the language parameter $\rho = \mathbf{A}$ and constructs the corresponding language description $spp = (X, L, R)$ according to SSetup algorithm defined above, then forwards $(spp, \mathbf{b})$ to $\mathcal{B}_1$ and outputs whatever bit $\mathcal{B}_1$ outputs.

By construction, $\mathbf{b} \in L$ if and only if $(\mathbf{A}, \mathbf{b})$ is a valid Module-LWE sample. Therefore, $\mathcal{B}_1$ distinguishes samples drawn from L and $X \setminus L$ with exactly the same advantage as distinguishing Module-LWE samples from uniform. The claimed bound on the advantage and the running time follows immediately. □

HPS Construction. Given the above SMP, we construct an HPS $\mathsf{HPS} = (\mathsf{HSetup}, \alpha_{(\cdot)}, \mathsf{Pub}, \mathsf{Priv})$, as illustrated in Figure 3. In the construction, the hash key space is defined as $\mathcal{HSK} = \chi_{\beta_2}^m$, where χ_{β_2} is a CBD over the ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^N+1)$ with support $\{-\beta_2, \cdots, \beta_2\}$. Pub and Priv algorithms employ a reconciliation function Rec [25], which applies coefficient-wise rounding to a ring element in $\mathcal{R}_q$ and maps it to an N-bit binary string as the shared key.

We now show that our HPS $\mathsf{HPS} = (\mathsf{HSetup}, \alpha_{(\cdot)}, \mathsf{Pub}, \mathsf{Priv})$ constructed above is correct and ε-smooth.

Theorem 7 (Correctness)**.** *Let* $hpp = (X, L, R, \mathcal{HSK}, \mathcal{HPK}, \Pi, \mathcal{H}, \mathbf{A}) \leftarrow_{\$} \mathsf{HSetup}(1^\lambda)$*. Assume that* $4m\beta_1\beta_2 N^2 \le 2^{-\omega(\lambda)}q$*. Then the hash values computed by* Pub *and* Priv *agree except with negligible probability. That is,*

$$\Pr\left[\pi \neq \pi' \;\middle|\; \begin{array}{l} \pi = \mathsf{Pub}(hpp, \mathbf{hpk}, \mathbf{x}, (\mathbf{r}, \mathbf{e})) \\ \pi' = \mathsf{Priv}(hpp, \mathbf{hsk}, \mathbf{x}) \end{array}\right] \le \mathrm{n}(\lambda),$$

where $\mathbf{hsk} \leftarrow_{\$} \mathcal{HSK}$*,* $\mathbf{hpk} = \alpha_{\mathbf{A}}(\mathbf{hsk})$ *and* $(\mathbf{x} = \mathbf{A}^T \cdot \mathbf{x} + \mathbf{e}, (\mathbf{r}, \mathbf{e})) \in R$*.*

Proof. In our construction, $\mathsf{Pub}(hpp, \mathbf{hpk}, \mathbf{x}, (\mathbf{r}, \mathbf{e})) = \mathsf{Rec}(\mathbf{r}^T \cdot \mathbf{A} \cdot \mathbf{hsk})$, $\mathsf{Priv}(hpp, \mathbf{hsk}, \mathbf{x}) = \mathsf{Rec}(\mathbf{r}^T \cdot \mathbf{A} \cdot \mathbf{hsk} + \mathbf{e}^T \cdot \mathbf{hsk})$. The two inputs to the function Rec differ by the additive

$\underline{\mathsf{HSetup}(1^\lambda)}$:
$(X, L, R) \leftarrow_\$ \mathsf{SSetup}(1^\lambda)$ // The algorithm implicitly samples $\mathbf{A}$
$\mathcal{HSK} = \chi_{\beta_2}^m, \mathcal{HPK} = \mathcal{R}_q^n, \Pi = \{0,1\}^N$
$\mathcal{H} = \{h_{\mathbf{hsk}} : \mathcal{R}_q^m \to \{0,1\}^N | \mathbf{hsk} \in \mathcal{HSK}\}$
$hpp = (X, L, R, \mathcal{HSK}, \mathcal{HPK}, \Pi, \mathcal{H}, \mathbf{A})$
Return hpp

$\underline{\alpha_{\mathbf{A}}(hpp, \mathbf{hsk})}$:
Parse $\mathbf{hsk} \in \mathcal{HSK}$ // $\mathbf{hsk} \leftarrow_\$ \mathsf{Cbd}(m, N, \beta_2)$
$\mathbf{hpk} = \mathbf{A} \cdot \mathbf{hsk}$
Return $\mathbf{hpk}$

$\underline{\mathsf{Pub}(hpp, \mathbf{hpk}, \mathbf{x}, w)}$:
Parse $w = (\mathbf{r}, \mathbf{e})$ // $\mathbf{r} \leftarrow_\$ \mathsf{Cbd}(n, N, \beta_1), \mathbf{e} \leftarrow_\$ \mathsf{Cbd}(m, N, \beta_1)$
$\pi = \mathsf{Rec}(\mathbf{r}^T \cdot \mathbf{hpk}) \in \{0,1\}^N$
Return π

$\underline{\mathsf{Priv}(hpp, \mathbf{hsk}, \mathbf{x})}$:
$\pi = \mathsf{Rec}(\mathbf{x}^T \cdot \mathbf{hsk}) \in \{0,1\}^N$
Return π

$\underline{\mathsf{Rec}(k)}$:
Parse $k = \sum_{i=1}^N a_i X^{i-1} \in \mathcal{R}_q$
For $i \in \{1, \cdots, N\}$:
$\pi_i = \mathsf{Rnd}(a_i) \in \{0,1\}$
Return $\pi \in \{0,1\}^N$

$\underline{\mathsf{Rnd}(a_i)}$:
Parse $a_i \in \mathbb{Z}_q$
If $\frac{q}{4} \le a_i \le \frac{3q}{4}$ then
Return 1
Else return 0

$\underline{\mathsf{Cbd}(t, N, h)}$:
For $i \in [t]$:
For $j \in [N]$:
For $v \in [h]$:
$b_v, c_v \leftarrow_\$ \{0,1\}$
$a_j = \sum_{v=1}^h (b_v - c_v)$
$f_i = \sum_{j=1}^N a_j X^{j-1}$
Return $\mathbf{f} = (f_1, \cdots, f_t) \in \mathcal{R}_q^t$

Figure 3: HPS from lattices.

noise term $\delta = \mathbf{e}^T \cdot \mathbf{hsk} = \sum_{i=1}^m e_i \cdot hsk_i$. For every $i \in [m]$ and $j \in [N]$, let $e_{i,j}$ and $hsk_{i,j}$ denote the j-th coefficients of the i-th elements in the vectors $\mathbf{e}$ and $\mathbf{hsk}$, respectively. By construction, $e_{i,j}$ and $hsk_{i,j}$ are independent and bounded as $|e_{i,j}| \le \beta_1$, $|hsk_{i,j}| \le \beta_2$. Then we can get

$$\begin{aligned}\left\|\delta\right\|_\infty = \left\|\mathbf{e}^T \cdot \mathbf{hsk}\right\|_\infty &= \left\|\sum_{i=1}^m e_i \cdot hsk_i\right\|_\infty \\ &\le \sum_{i=1}^m \left\|e_i \cdot hsk_i\right\|_\infty \le m\beta_1\beta_2 N.\end{aligned}$$

The above $\|\delta\|_\infty$ upper-bounds the maximum coefficient-wise perturbation introduced by the additive noise term δ. In particular, each coefficient of $\mathbf{r}^T \cdot \mathbf{A} \cdot \mathbf{hsk}$ can be shifted by at most $\|\delta\|_\infty$. According to our construction, a mismatch between the two outputs of Rec can occur only if some coefficient of $\mathbf{r}^T \cdot \mathbf{A} \cdot \mathbf{hsk}$ lies in the danger interval, namely an interval such that adding the noise term δ may shift the coefficient across a rounding boundary $\{\frac{q}{4}, \frac{3q}{4}\}$. In other words, the danger interval is $S^* = \left[\frac{q}{4} \pm ||\delta||_\infty\right] \cup \left[\frac{3q}{4} \pm ||\delta||_\infty\right]$.

Let $u = \mathbf{r}^T \cdot \mathbf{A} \cdot \mathbf{hsk} \in \mathcal{R}_q$. For any fixed coefficient index $j \in [N]$, the probability that u_j falls within the danger interval S^* is bounded by $\Pr[u_j \in S^*] \le \frac{|S^*|}{q} = \frac{4m\beta_1\beta_2 N}{q}$.

By the union bound over all N coefficients,

$$\begin{aligned}&\Pr\left[\pi \ne \pi' \,\middle|\, \begin{array}{l}\pi = \mathsf{Rec}(\mathbf{r}^T \cdot \mathbf{A} \cdot \mathbf{hsk}), \\ \pi' = \mathsf{Rec}(\mathbf{r}^T \cdot \mathbf{A} \cdot \mathbf{hsk} + \mathbf{e}^T \cdot \mathbf{hsk})\end{array}\right] \\ &\le \frac{4m\beta_1\beta_2 N^2}{q} \le 2^{-\omega(\lambda)}.\end{aligned}$$

□

Theorem 8. *Let* $hpp = (X, L, R, \mathcal{HSK}, \mathcal{HPK}, \Pi, \mathcal{H}, \mathbf{A}) \leftarrow_\$ \mathsf{HSetup}(1^\lambda)$ *be the public parameters. Assume that* $q \le 2^{-\omega(\lambda)}(2\beta_2 + 1)^m$. *Then the HPS* HPS *is* ε*-smooth for negligible* ε *in* λ. *That is*

$$\left\{(\mathbf{x}, \mathbf{hpk}, \mathsf{Priv}(hpp, \mathbf{hsk}, \mathbf{x})) \,\middle|\, \begin{array}{c}\mathbf{hsk} \leftarrow_\$ \chi_{\beta_2}^m \\ \mathbf{hpk} = \alpha_{\mathbf{A}}(\mathbf{hsk}),\ \mathbf{x} \leftarrow_\$ X \setminus L\end{array}\right\}$$
$$\approx_s \left\{(\mathbf{x}, \mathbf{hpk}, \pi) \,\middle|\, \begin{array}{c}\mathbf{hsk} \leftarrow_\$ \chi_{\beta_2}^m,\ \pi \leftarrow_\$ \{0,1\}^N \\ \mathbf{hpk} = \alpha_{\mathbf{A}}(\mathbf{hsk}),\ \mathbf{x} \leftarrow_\$ X \setminus L\end{array}\right\}.$$

Proof. By the definition of the private evaluation algorithm in our instantiation, it holds that $\mathsf{Priv}(hpp, \mathbf{hsk}, \mathbf{x}) = \mathsf{Rec}(\mathbf{x}^T \cdot \mathbf{hsk})$. That is

$$\left\{(\mathbf{x}, \mathbf{hpk}, \mathsf{Priv}(hpp, \mathbf{hsk}, \mathbf{x})) \,\middle|\, \begin{array}{c}\mathbf{hsk} \leftarrow_\$ \chi_{\beta_2}^m \\ \mathbf{hpk} = \alpha_{\mathbf{A}}(\mathbf{hsk}),\ \mathbf{x} \leftarrow_\$ X \setminus L\end{array}\right\}$$
$$\equiv \left\{(\mathbf{x}, \mathbf{hpk}, \mathsf{Rec}(\mathbf{x}^T \cdot \mathbf{hsk})) \,\middle|\, \begin{array}{c}\mathbf{hsk} \leftarrow_\$ \chi_{\beta_2}^m \\ \mathbf{hpk} = \alpha_{\mathbf{A}}(\mathbf{hsk}),\ \mathbf{x} \leftarrow_\$ X \setminus L\end{array}\right\}$$

Let $u \leftarrow_\$ \mathcal{R}_q$. Since Rec is a deterministic reconciliation fuction, statistical distance does not increase under applying Rec [47, Proposition 2.2]. Therefore,

$$\begin{aligned}&\Delta\left(\left\{\mathsf{Rec}(\mathbf{x}^T \cdot \mathbf{hsk}) \,\middle|\, \begin{array}{c}\mathbf{hsk} \leftarrow_\$ \chi_{\beta_2}^m \\ \mathbf{x} \leftarrow_\$ X \setminus L\end{array}\right\}, \left\{\mathsf{Rec}(u) \mid u \leftarrow_\$ \mathcal{R}_q\right\}\right) \\ &= \Delta\left(\left\{\mathsf{Rec}(\mathbf{x}^T \cdot \mathbf{hsk}) \,\middle|\, \begin{array}{c}\mathbf{hsk} \leftarrow_\$ \chi_{\beta_2}^m \\ \mathbf{x} \leftarrow_\$ X \setminus L\end{array}\right\}, \left\{\pi \mid \pi \leftarrow_\$ \{0,1\}^N\right\}\right) \\ &\le \Delta\left(\left\{\mathbf{x}^T \cdot \mathbf{hsk} \,\middle|\, \begin{array}{c}\mathbf{hsk} \leftarrow_\$ \chi_{\beta_2}^m \\ \mathbf{x} \leftarrow_\$ X \setminus L\end{array}\right\}, \left\{u \mid u \leftarrow_\$ \mathcal{R}_q\right\}\right).\end{aligned}$$

Fix $\mathbf{x} \in X \setminus L$ and define the auxiliary (pre-reconciliation) map $g_{\mathbf{x}} : \mathcal{HSK} \to \mathcal{R}_q$ by $g_{\mathbf{x}}(\mathbf{hsk}) = \mathbf{x}^T \cdot \mathbf{hsk} = \sum_{i=1}^m x_i \cdot hsk_i$. For fixed $\mathbf{x}$ the funtion $g_{\mathbf{x}}(\mathbf{hsk})$ is exactly the generalized knapsack function over $\mathcal{R}_q$ analyzed in [47, Definition 4.1]. Consequently, the generalized knapsack analysis yields

$$\begin{aligned}&\Delta\left(\left\{\mathbf{x}^T \cdot \mathbf{hsk} \,\middle|\, \begin{array}{c}\mathbf{hsk} \leftarrow_\$ \chi_{\beta_2}^m \\ \mathbf{x} \leftarrow_\$ X \setminus L\end{array}\right\}, \left\{u \mid u \leftarrow_\$ \mathcal{R}_q\right\}\right) \\ &\le 1/2 \cdot \sqrt{(1 + q/(2\beta_2 + 1)^m)^N - 1}.\end{aligned}$$

Table 2: Parameter settings for NIKE protocols.

Parameter	Description	Level 1									Level 3				
		SWOOSH					Hesse-Lattice/Ours				SWOOSH	Hesse-Lattice/Ours			
		L1-S1†	L1-S2	L1-S3	L1-S4	L1-S5	L1-S6	L1-S7	L1-S8	L1-S9	L3-S1†	L3-S2	L3-S3	L3-S4	L3-S5
N	Dimension of $\mathcal{R}_q$	256	256	256	256	256	256	256	256	256	256	256	256	256	256
$n^{\ddagger}$	Rows of matrix $\mathbf{A}$	-	-	-	-	-	27	28	26	27	-	34	34	35	33
$m^{\ddagger}$	Columns of matrix $\mathbf{A}$	32	48	47	47	46	31	31	30	30	53	37	36	39	35
β_1	Bound of χ_{β_1}	1	2	2^4+2	2^9	2^{11}	2	1	2^7	2	1	2^{11}	2^{13}	2^7	2^{15}
$\beta_2^{\ddagger}$	Bound of χ_{β_2}	-	-	-	-	-	2^{14}	$2^{14}+2^{13}$	$2^{14}+2^{13}$	$2^{14}+2^{13}$	-	2^{15}	$2^{15}+2^{14}+2^{12}$	2^{15}	2^{16}
q	Prime modulus	$2^{214}-255$					$2^{140}-2^9\times 77+1$				$2^{214}-255$	$2^{140}-2^9\times 77+1$			

†: L1-S1 corresponds to SWOOSH instantiated with its originally recommended parameters. L3-S1 denotes the SWOOSH parameter set that maximizes the estimated bit security under its originally recommended modulus q. Under the CKS-heavy model with $\mu = 2^{30}$ users, L1-S1 and L3-S1 fail to achieve 128-bit and 192-bit security, respectively. ‡: SWOOSH does not involve the parameters n and β_2.

Combining the above inequalities, we obtain $\Delta\left(\left\{\mathsf{Rec}(\mathbf{x}^T\cdot\mathbf{hsk}) \;\middle|\; \begin{matrix}\mathbf{hsk}\leftarrow_\$ \chi_{\beta_2}^m \\ \mathbf{x}\leftarrow_\$ X\setminus L\end{matrix}\right\}, \left\{\pi \;\middle|\; \pi\leftarrow_\$ \{0,1\}^N\right\}\right) \le \varepsilon$, where $\varepsilon = 1/2\cdot\sqrt{(1+q/(2\beta_2+1)^m)^N-1}$.

Finally, statistical distance is preserved when taking joint distributions with identically distributed side information. Therefore, adjoining the (identically distributed) pair $(\mathbf{x},\mathbf{hpk})$ to both distributions yields

$$\left\{(\mathbf{x},\mathbf{hpk},\mathsf{Rec}(\mathbf{x}^T\cdot\mathbf{hsk})) \;\middle|\; \begin{matrix}\mathbf{hsk}\leftarrow_\$ \chi_{\beta_2}^m \\ \mathbf{hpk}=\alpha_{\mathbf{A}}(\mathbf{hsk}),\ \mathbf{x}\leftarrow_\$ X\setminus L\end{matrix}\right\}$$
$$\equiv_\varepsilon \left\{(\mathbf{x},\mathbf{hpk},\pi) \;\middle|\; \begin{matrix}\mathbf{hsk}\leftarrow_\$ \chi_{\beta_2}^m,\ \pi\leftarrow_\$ \{0,1\}^N \\ \mathbf{hpk}=\alpha_{\mathbf{A}}(\mathbf{hsk}),\ \mathbf{x}\leftarrow_\$ X\setminus L\end{matrix}\right\}.$$

Under the assumption $q \le 2^{-\omega(\lambda)}(2\beta_2+1)^m$, we have $\varepsilon \le 1/2\sqrt{(1+2^{-\omega(\lambda)})^N-1}$, which is negligible in λ. □

5.2 D-KEM$_{\mathsf{HPS}}$ and NIKE$_{\mathsf{HPS}}$ from Lattices

Starting from our lattice-based HPS, we can obtain a decoupled KEM D-KEM$_{\mathsf{HPS}}$ via the HPS-to-KEM transformation in Section 4, and then obtain a lattice-based NIKE protocol NIKE$_{\mathsf{HPS}}$ via the generic KEM-to-NIKE construction.

We note that, although the witness for $\mathbf{x}\in L$ in Figure 3 is defined as $w=(\mathbf{r},\mathbf{e})$, the public evaluation algorithm Pub only needs $\mathbf{r}$. This simplification does not affect correctness or security. We therefore define the internal secret output by Encap_1 in D-KEM$_{\mathsf{HPS}}$ as $\mathbf{r}$ (without $\mathbf{e}$). Accordingly, a user's private and public keys in NIKE$_{\mathsf{HPS}}$ are $sk=(\mathbf{hsk},\mathbf{r},\mathbf{x})$ and $pk=(\mathbf{hpk},\mathbf{x})$. The concrete instantiation of NIKE$_{\mathsf{HPS}}$ used in our implementation is provided in Appendix C.1.

6 Evaluation

We evaluate both the security and efficiency of our NIKE$_{\mathsf{HPS}}$ protocol by conducting a comparative assessment against existing lattice-based NIKE constructions, including SWOOSH and Hesse-Lattice.

6.1 Parameter Selection

Our evaluation considers parameter sets targeting widely used post-quantum security regimes inspired by the NIST PQC standardization process [1]. Specifically, we focus on configurations aligned with NIST security Levels 1 and 3, corresponding to 128-bit and 192-bit classical security, respectively. We emphasize that concrete security is estimated using state-of-the-art lattice attacks and may slightly exceed or fall below the nominal targets (e.g., 129-bit security for a nominal 128-bit setting). Following the analysis in Appendix C.2, the selected parameter sets are summarized in Table 2. Parameter sets are denoted by L1-Si or L3-Si, where the prefix indicates the target security level and i distinguishes different configurations within the same level.

The original SWOOSH construction [25] recommends a lattice-based instantiation with an approximately 214-bit modulus. In contrast, our construction attains comparable or higher concrete security with a substantially smaller modulus of about 140 bits. As discussed later, due to its larger reduction loss, a 140-bit modulus is insufficient for SWOOSH, whereas the 214-bit recommendation is consistent with its original analysis. Nevertheless, under our concrete-security estimation methodology, SWOOSH instantiated with its recommended parameters (L1-S1) does not achieve NIST security Level 1 without further parameter adjustments.

Accordingly, we retune the SWOOSH and focus our evaluation on Security Level 1, consistent with the original evaluation in [25]. Moreover, under the CKS-heavy model, scaling SWOOSH to Level 3 would require a prohibitively large modulus. In particular, accounting for the reduction loss in its security proof and translating it to the CKS-heavy setting, the modulus q would need to be at least 262 bits, resulting in substantial computational and public-key overheads.

For this reason, we do not include an explicit SWOOSH instantiation targeting Level 3 in our evaluation. Instead, we show that even under its originally recommended 214-bit modulus, which is far below the 262 bits required to target Level 3 under the CKS-heavy setting and therefore falls short of security Level 3, SWOOSH remains less efficient than

Table 3: Correctness error probabilities and security level.

Setting	SWOOSH		Hesse-Lattice		Ours		
	$E^{\dagger}$	$B^{\dagger}$	E	B	E	B	$\varepsilon^{\dagger}$
L1-S1	2^{-191}	58	-	-	-	-	-
L1-S2	2^{-188}	**127**	-	-	-	-	-
L1-S3	2^{-182}	**126**	-	-	-	-	-
L1-S4	2^{-172}	**131**	-	-	-	-	-
L1-S5	2^{-168}	**130**	-	-	-	-	-
L1-S6	-	-	2^{-102}	128	2^{-102}	**128**	2^{-159}
L1-S7	-	-	2^{-102}	132	2^{-102}	**131**	2^{-168}
L1-S8	-	-	2^{-95}	129	2^{-95}	**129**	2^{-160}
L1-S9	-	-	2^{-101}	129	2^{-101}	**128**	2^{-160}
L3-S1	2^{-190}	145	-	-	-	-	-
L3-S2	-	-	2^{-90}	193	2^{-90}	**192**	2^{-223}
L3-S3	-	-	2^{-88}	196	2^{-88}	**196**	2^{-227}
L3-S4	-	-	2^{-94}	193	2^{-94}	**192**	2^{-239}
L3-S5	-	-	2^{-85}	193	2^{-85}	**193**	2^{-224}

†: E denotes the honest-correctness error probability, B denotes the achieved security level in bits, ε denotes the achieved ε-smoothness of our construction/Hesse-Lattice.

our protocol instantiated to target security Level 3. For completeness, we also report the maximum concrete security that SWOOSH can achieve under this parameter setting.

6.2 Correctness and Security

Using these parameter sets, we evaluate the honest-correctness error and concrete security of three lattice-based NIKE constructions: (i) SWOOSH, (ii) Hesse-Lattice, and (iii) our $\mathsf{NIKE}_{\mathsf{HPS}}$. Security is analyzed in the CKS-heavy model. Results are summarized in Table 3.

Comparison of Security. We perform a comparative evaluation of the security of the lattice-based NIKE constructions against lattice attacks, including primal uSVP, BDD, dual, and dual-hybrid attacks. Security estimates are computed using SageMath and the Lattice Estimator [2]. To account for the high lattice dimension, we also consider sieving-based basis reduction with parallelization and vector-level optimizations, as well as algebraic attacks based on Gröbner basis techniques, including the Arora–GB attack.

Under the CKS-heavy model, the three constructions incur different reduction losses (see Table 1), and these losses depend directly on the number of users μ. Consequently, specifying μ is necessary for a meaningful security evaluation. Many real-world systems involve a large user base. For instance, Facebook reports over 10^8 active users. Accordingly, we set $\mu = 2^{30} \approx 10^9$, a regime in which reduction loss becomes a dominant factor in determining effective security.

Fixing $\mu = 2^{30}$, Table 3 shows that the original SWOOSH parameter set L1-S1, which uses an approximately 214-bit modulus q, does not meet security Level 1, indicating that the original parameters are unsuitable for large-scale deployments. By increasing the matrix dimension m or enlarging the CBD bound β_1, Level 1 can be attained. However, Level 3 remains unattainable with a 214-bit modulus. Under the CKS-heavy setting, the concrete-security bound of SWOOSH is constrained not only by the Module-LWE hardness but also by the statistical term $4\beta_1 N/q$, which dominates in large-scale settings. Achieving Level 3 security for $\mu = 2^{30}$ with $N = 256$ and $\beta_1 \geq 1$ would require a modulus of at least 262 bits. Moreover, instantiating SWOOSH with the 140-bit modulus employed in this work renders Level 1 unachievable, since $4\beta_1 N/q < 2^{-187}$ would require a modulus larger than 198 bits, close to the originally recommended 214-bit modulus.

In contrast, with the 140-bit modulus, our $\mathsf{NIKE}_{\mathsf{HPS}}$ and Hesse-Lattice achieve the intended security levels by appropriately tuning matrix dimensions m, n and the CBD bound β_2: sets L1-S5–L1-S9 attain Level 1, and L3-S2–L3-S5 attain Level 3. These results illustrate that our construction scales efficiently to large user bases and outperforms SWOOSH in achieving high concrete security under CKS-heavy setting.

Comparison of Correctness. For our construction and Hesse-Lattice, the correctness error probability is upper bounded by $4m\beta_1\beta_2 N^2/q$. Consequently, under identical parameter settings, the two constructions exhibit the same correctness error probability. In contrast, SWOOSH admits a bound of $4m\beta_1^2 N^2/q$, independent of β_2. Across all parameter settings considered, our $\mathsf{NIKE}_{\mathsf{HPS}}$ achieves correctness error probabilities ranging from 2^{-102} to 2^{-85}, which are practically acceptable in lattice-based cryptographic schemes [3, 13, 28].

To match the correctness error probabilities of SWOOSH, the modulus q needs to be increased accordingly in our construction[2]. As an illustrative example, we consider the Level 1 settings from L1-S2 to L1-S5[3]. Specifically, using the 214-bit modulus recommended for SWOOSH and setting $(N, n, m, \beta_1, \beta_2)$ to $(256, 42, 48, 1, 2^{10} + 2^8)$, $(256, 41, 46, 2, 2^{11})$, $(256, 40, 45, 2^7, 2^{11})$, and $(256, 40, 46, 2^{11}, 2^{11} + 2^6)$, respectively, our construction achieves correctness error probabilities of 2^{-180}, 2^{-178}, 2^{-172}, and 2^{-168}, respectively. These values are comparable to those of SWOOSH while still maintaining Level 1 security. Under these matched settings, our construction reduces the public-key size by 7.45% and accelerates NSetup, NGen, and SK by 14.47%, 15.51%, and 12.25%, respectively, compared with SWOOSH. Compared with Hesse-Lattice under the same parameters, our construction reduces the computation time of SK by approximately 50% while maintaining comparable correctness and security.

Similarly, our current parameter sets targeting Level 3 security already achieve practically acceptable correctness, while further reducing the correctness error probability can likewise be achieved by increasing the modulus q.

[2] Note that our construction already achieves practically acceptable correctness under the parameter settings using a 140-bit modulus. The matched settings instead illustrate the flexibility of trading a larger modulus for lower correctness error probabilities.

[3] We exclude the set L1-S1 since SWOOSH does not achieve the corresponding target security level under that setting.

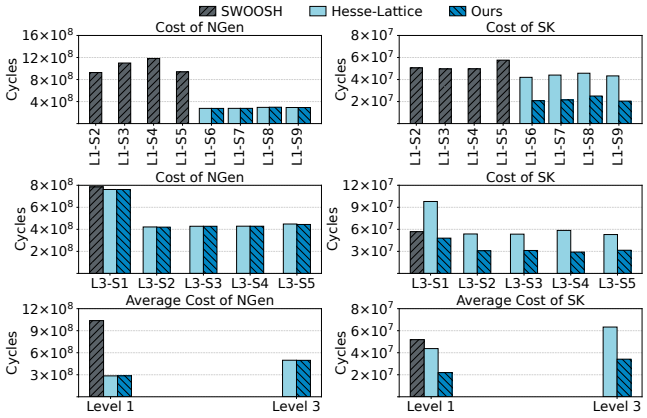

Figure 4: Computational cost comparison with NTT-based acceleration.

6.3 Efficiency Evaluation

We evaluate the efficiency of $\mathsf{NIKE}_{\mathsf{HPS}}$ under the parameter sets in Tables 2 on a Windows laptop with a 13th Gen Intel Core i7-13700H, and 16 GB RAM. All reported costs are averaged over 1,000 executions. Table 5 in Appendix D reports the estimated theoretical computational costs of the considered NIKE protocols, providing a reference for interpreting the experimental results.

Baseline Implementation. We establish a baseline by implementing three NIKE protocols over $\mathcal{R}_q$ using naive schoolbook polynomial multiplication, with large-modulus arithmetic handled by the GNU Multiple Precision (GMP) library. Public and secret keys are stored explicitly as vectors of polynomials, and all matrix–vector products and inner products are computed with straightforward nested loops. For our protocol, the SK cost is averaged over both identity orderings.

Table 4 reports the performance under the considered parameter settings. At security Level 1, our $\mathsf{NIKE}_{\mathsf{HPS}}$ yields public keys that are 2.51× smaller than those of SWOOSH, and speeds up shared-key derivation (SK) by 1.82×, with key generation (NGen) further accelerated by 2.78×, on average. For the 192-bit target (Level 3), SWOOSH instantiated with L3-S1 provides only about 145-bit security. In comparison, our $\mathsf{NIKE}_{\mathsf{HPS}}$ using parameter sets L3-S2–L3-S5 (achieving $\geq$ 192-bit security) has 2.30× smaller public keys and shared-key derivation is 1.26× faster. Compared with Hesse-Lattice at the same level, our shared-key derivation remains roughly 2.00× faster, on average.

NTT-based Acceleration. To improve efficiency, we adopt Number Theoretic Transform (NTT)-based polynomial arithmetic following standard practice [13, 25], reducing the cost of polynomial multiplication from $O(N^2)$ to $O(N \log N)$. In our implementation, the public matrix $\mathbf{A}$ and key pairs (pk, sk) are generated directly in the NTT domain. Public keys are encoded and transmitted in the NTT domain as well, avoiding unnecessary domain conversions. With NTT acceleration, Figure 4 reports the computational costs of public-private key generation and shared-key derivation for three constructions.

Table 4: Performance comparison using naive schoolbook polynomial multiplication.

Level	Protocol	Setting	Cycles			PK
			NSetup	NGen	SK	(KB)
1	SWOOSH	L1-S2	43 154 887	11 605 229 568	69 409 768	639.02
		L1-S3	41 692 336	13 262 287 019	73 601 209	625.70
		L1-S4	41 400 559	14 338 919 854	73 258 248	625.65
		L1-S5	39 623 065	13 965 226 540	78 058 931	612.38
		Avg	**41 467 712**	**13 292 915 745**	**73 582 039**	**625.69**
	Hesse-Lattice	L1-S6	13 886 533	4 823 133 822	82 430 348	251.94
		L1-S7	14 377 430	4 773 701 251	79 902 017	256.25
		L1-S8	13 082 480	5 048 041 862	88 760 320	243.23
		L1-S9	13 604 107	4 525 400 589	78 608 064	247.61
		Avg	**13 737 638**	**4 792 569 381**	**82 425 187**	**249.76**
	Ours	L1-S6	13 910 055	4 868 043 281	40 536 358	251.90
		L1-S7	14 343 525	4 726 062 128	39 130 444	256.27
		L1-S8	12 703 627	5 000 729 279	43 429 291	243.26
		L1-S9	13 201 608	4 526 306 932	38 513 234	247.58
		Avg	**13 539 704**	**4 780 285 405**	**40 402 332**	**249.75**
3	SWOOSH	L3-S1†	52 752 403	12 593 202 517	71 089 502	705.62
	Hesse-Lattice	L3-S2	21 130 467	8 247 136 108	114 456 332	308.37
		L3-S3	20 338 551	8 493 875 942	111 447 991	304.05
		L3-S4	22 688 603	8 709 029 573	115 513 036	321.41
		L3-S5	19 190 172	7 709 854 180	108 003 792	295.35
		Avg	**20 836 948**	**8 289 973 951**	**112 355 288**	**307.30**
	Ours	L3-S2	20 911 231	8 175 829 586	56 168 134	308.43
		L3-S3	20 310 654	8 488 085 950	56 622 102	304.01
		L3-S4	22 637 464	8 712 269 251	58 128 926	321.41
		L3-S5	19 069 103	7 700 630 366	54 040 821	295.34
		Avg	**20 732 113**	**8 269 203 788**	**56 239 996**	**307.30**

†:L3-S1 is included only to evaluate SWOOSH's efficiency at the highest security level attainable with SWOOSH's recommended modulus q.

At Level 1, averaged over multiple experimental runs, shared-key derivation in our construction takes 21 937 640 cycles, compared with 51 830 921 cycles for SWOOSH and 43 761 097 cycles for Hesse-Lattice, corresponding to speedups of 2.36× and 1.99×, respectively. Moreover, our key generation is 3.63× faster than that of SWOOSH. For the 192-bit security target (Level 3), SWOOSH instantiated with L3-S1 still requires 56 794 121 cycles for shared-key derivation. In contrast, our construction achieves $\geq$ 192-bit security with only 30 676 044 cycles, yielding an approximately 1.85× speedup while offering stronger security guarantees.

The gap is dominated by protocol-level overhead rather than polynomial arithmetic. SWOOSH incurs extra hashing overhead, which was originally implemented using cSHAKE [40], and performs more ring operations during shared-key derivation (see Table 5), and it also operates over a larger modulus q. In contrast, our construction and Hesse-Lattice perform the same core operations for key generation, but shared-key derivation in Hesse-Lattice requires evaluating both Priv and Pub, whereas in our protocol each party evaluates only one of them.

7 Conclusion

We have presented a lattice-based HPS and its instantiation into a decoupled KEM, which in turn yields a NIKE protocol, $\mathsf{NIKE}_{\mathsf{HPS}}$. Building on this construction, we also obtain a lattice-based variant of the Hesse et al. [36] framework, denoted *Hesse-Lattice*, enabling direct performance and security comparisons. Our theoretical and empirical evaluations show that $\mathsf{NIKE}_{\mathsf{HPS}}$ achieves high correctness guarantees and the intended concrete security levels while remaining computationally efficient. In particular, shared-key derivation in our protocol is substantially faster than in SWOOSH and moderately higher than in Hesse-Lattice.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. Ying Li and Lei Zhang are supported in part by the National Natural Science Foundation of China (No. 62372177). Qiqi Lai is supported in part by the National Natural Science Foundation of China (No. 62172266) and the Key Research and Development Program of Shaanxi under Grant 2024GXZDCYL-01-15. This work is also supported in part by the Graduate International Conference Special Fund of East China Normal University

Ethical Considerations

Our work involves no human subjects or per-sonal data. All evaluations are based on analytical bench-marking and parameter analysis. Therefore, the direct risk ofconducting the research is minimal. The main stakeholdergroups include security researchers, standardization commu-nities, system architects, developers, end users, and possibleadversaries. The work may positively support future research, parameter evaluation, and standardization. Potential negativeimpacts are indirect, such as premature deployment, futurere-parameterization if stronger attacks emerge, or possibleuse of the published details by attackers. We mitigate theserisks by presenting the work as a research result rather than adeployment-ready system, providing explicit security and correctness analyses, and recommending further cryptanalytic, implementation, and system-level validation before practicaluse.

Open Science

We have open-sourced the full implementation of $\mathsf{NIKE}_{\mathsf{HPS}}$, including parameter settings, and core algorithms. All artifacts are available on Zenodo at https://zenodo.org/records/20588261.

References

[1] Gorjan Alagic, Daniel Apon, David Coope, Quynh Dang, Thinh Dang, John Kelsey, Jacob Lichtinger, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, Daniel Smith-Tone, and Yi-Kai Liu. Status report on the third round of the nist post-quantum cryptography standardization process. Technical Report NIST IR 8413, National Institute of Standards and Technology, July 2022. https://doi.org/10.6028/NIST.IR.8413-upd1.

[2] Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. Cryptology ePrint Archive, Paper 2015/046, 2015. https://eprint.iacr.org/2015/046.

[3] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key Exchange—A new hope. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 327–343, Austin, TX, August 2016. USENIX Association. https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/alkim.

[4] Joël Alwen, Sandro Coretti, Yevgeniy Dodis, and Yiannis Tselekounis. Security analysis and improvements for the ietf mls standard for group messaging. In *Advances in Cryptology – CRYPTO 2020*, pages 248–277, Cham, 2020. Springer International Publishing. https://doi.org/10.1007/978-3-030-56784-2_9.

[5] Andris Ambainis, Mike Hamburg, and Dominique Unruh. Quantum security proofs using semi-classical oracles. In *Advances in Cryptology – CRYPTO 2019*, pages 269–295, Cham, 2019. Springer International Publishing. https://doi.org/10.1007/978-3-030-26951-7_10.

[6] Christoph Bader, Dennis Hofheinz, Tibor Jager, Eike Kiltz, and Yong Li. Tightly-secure authenticated key exchange. In *Theory of Cryptography*, pages 629–658, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-46494-6_26.

[7] Christoph Bader, Tibor Jager, Yong Li, and Sven Schäge. On the impossibility of tight cryptographic reductions. In *Advances in Cryptology – EUROCRYPT 2016*, pages 273–304, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-49896-5_10.

[8] Manuel Barbosa, Kai Gellert, Julia Hesse, and Stanislaw Jarecki. Bare pake: Universally composable key exchange from just passwords. In *Advances in Cryptology – CRYPTO 2024*, pages 183–217, Cham, 2024.

Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-68379-4_6.

[9] Hugo Beguinet, Céline Chevalier, David Pointcheval, Thomas Ricosset, and Mélissa Rossi. Get a cake: Generic transformations from key encaspulation mechanisms to password authenticated key exchanges. In *Applied Cryptography and Network Security*, pages 516–538, Cham, 2023. Springer Nature Switzerland. http://doi.org/10.1007/978-3-031-33491-7_19.

[10] Fabrice Benhamouda, Olivier Blazy, Léo Ducas, and Willy Quach. Hash proof systems over lattices revisited. In *Public-Key Cryptography – PKC 2018*, pages 644–674, Cham, 2018. Springer International Publishing. https://doi.org/10.1007/978-3-319-76581-5_22.

[11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In *Advances in Cryptology – ASIACRYPT 2011*, pages 41–69, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-25385-0_3.

[12] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *Advances in Cryptology – CRYPTO 2005*, pages 258–275, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. https://doi.org/10.1007/11535218_16.

[13] Joppe Bos, Leo Ducas, Eike Kiltz, T Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehle. Crystals - kyber: A cca-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 353–367, London, UK, 2018. IEEE. https://doi.org/10.1109/EuroSP.2018.00032.

[14] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. STOC '13, page 575–584, New York, NY, USA, 2013. Association for Computing Machinery. https://doi.org/10.1145/2488608.2488680.

[15] David Cash, Eike Kiltz, and Victor Shoup. The twin diffie-hellman problem and applications. In *Advances in Cryptology – EUROCRYPT 2008*, volume 4965, pages 127–145, 2008. https://doi.org/10.1007/978-3-540-78967-3_8.

[16] Katriel Cohn-Gordon, Cas Cremers, Luke Garratt, Jon Millican, and Kevin Milner. On ends-to-ends encryption: Asynchronous group messaging with strong security guarantees. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 1802–1819, New York, NY, USA, 2018. Association for Computing Machinery. https://doi.org/10.1145/3243734.3243747.

[17] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *Advances in Cryptology — EUROCRYPT 2002*, pages 45–64, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-46035-7_4.

[18] Hannah Davis and Felix Günther. Tighter proofs for the sigma and tls 1.3 key exchange protocols. In *Applied Cryptography and Network Security*, pages 448–479, Cham, 2021. Springer International Publishing. https://doi.org/10.1007/978-3-030-78375-4_18.

[19] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976. https://doi.org/10.1109/TIT.1976.1055638.

[20] Jintai Ding, Kevin Schmitt, and Zheng Zhang. A key exchange based on the short integer solution problem and the learning with errors problem. In *Codes, Cryptology and Information Security*, pages 105–117, Cham, 2019. Springer International Publishing. https://doi.org/0.1007/978-3-030-16458-4_8.

[21] Jintai Ding, Xiang Xie, and Xiaodong Lin. A simple provably secure key exchange scheme based on the learning with errors problem. Cryptology ePrint Archive, Paper 2012/688, 2012. https://ia.cr/2012/688.

[22] Léo Ducas, Eike Kiltz, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, Feb. 2018. https://doi.org/10.13154/tches.v2018.i1.238-268.

[23] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. Falcon: Fast-fourier lattice-based compact signatures over NTRU. *Submission to the NIST's post-quantum cryptography standardization process*, 36:1–75, 2018. https://www.di.ens.fr/~prest/Publications/falcon.pdf.

[24] Eduarda S. V. Freire, Dennis Hofheinz, Eike Kiltz, and Kenneth G. Paterson. Non-interactive key exchange. In *Public-Key Cryptography – PKC 2013*, pages 254–271, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-36362-7_17.

[25] Phillip Gajland, Bor de Kock, Miguel Quaresma, Giulio Malavolta, and Peter Schwabe. SWOOSH: Efficient lattice-based non-interactive key exchange. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 487–504, Philadelphia, PA, August 2024. USENIX Association. https://www.usenix.org/conference/usenixsecurity24/presentation/gajland.

[26] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, page 197–206, New York, NY, USA, 2008. Association for Computing Machinery. https://doi.org/10.1145/1374376.1374407.

[27] Kristian Gjøsteen and Tibor Jager. Practical and tightly-secure digital signatures and authenticated key exchange. In *Advances in Cryptology – CRYPTO 2018*, pages 95–125, Cham, 2018. Springer International Publishing. https://doi.org/10.1007/978-3-319-96881-0_4.

[28] Lewis Glabush, Patrick Longa, Michael Naehrig, Chris Peikert, Douglas Stebila, and Fernando Virdia. FrodoKEM: A CCA-secure learning with errors key encapsulation mechanism. Cryptology ePrint Archive, Paper 2025/1861, 2025. https://eprint.iacr.org/2025/1861.

[29] Lov K. Grover. Quantum mechanics helps in searching for a needle in a haystack. *Phys. Rev. Lett.*, 79:325–328, Jul 1997. https://link.aps.org/doi/10.1103/PhysRevLett.79.325.

[30] Siyao Guo, Pritish Kamath, Alon Rosen, and Katerina Sotiraki. Limits on the efficiency of (ring) lwe based non-interactive key exchange. In *Public-Key Cryptography – PKC 2020: 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4–7, 2020, Proceedings, Part I*, page 374–395, Berlin, Heidelberg, 2020. Springer-Verlag. https://doi.org/10.1007/978-3-030-45374-9_13.

[31] Siyao Guo, Pritish Kamath, Alon Rosen, and Katerina Sotiraki. Limits on the efficiency of (ring) lwe-based non-interactive key exchange. *Journal of Cryptology*, 35:1–24, 2022. http://dx.doi.org/10.1007/s00145-021-09406-y.

[32] Shuai Han, Tibor Jager, Eike Kiltz, Shengli Liu, Jiaxin Pan, Doreen Riepel, and Sven Schäge. Authenticated key exchange and signatures with tight security in the standard model. In *Advances in Cryptology – CRYPTO 2021*, pages 670–700, Cham, 2021. Springer International Publishing. https://doi.org/10.1007/978-3-030-84259-8_23.

[33] Shuai Han, Shengli Liu, and Dawu Gu. Almost tight multi-user security under adaptive corruptions & leakages in the standard model. In *Advances in Cryptology – EUROCRYPT 2023*, pages 132–162, Cham, 2023. Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-30620-4_5.

[34] Shuai Han, Shengli Liu, Zhedong Wang, and Dawu Gu. Almost tight multi-user security under adaptive corruptions from lwe in the standard model. In *Advances in Cryptology – CRYPTO 2023*, pages 682–715, Cham, 2023. Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-38554-4_2.

[35] Sven Hebrok, Tim Leonhard Storm, Maximilian Radoy Felix Matthias Cramer, and Juraj Somorovsky. STEK sharing is not caring: Bypassing TLS authentication in web servers using session tickets. In *34rd USENIX Security Symposium (USENIX Security 25)*, Seattle, WA, USA, 2025. USENIX Association. https://www.usenix.org/conference/usenixsecurity25/presentation/hebrok.

[36] Julia Hesse, Dennis Hofheinz, and Lisa Kohl. On tightly secure non-interactive key exchange. In *Advances in Cryptology – CRYPTO 2018*, pages 65–94, Cham, 2018. Springer International Publishing. https://doi.org/10.1007/978-3-319-96881-0_3.

[37] Julia Hesse, Dennis Hofheinz, Lisa Kohl, and Roman Langrehr. Towards tight adaptive security of non-interactive key exchange. In *Theory of Cryptography*, pages 286–316, Cham, 2021. Springer International Publishing. https://doi.org/10.1007/978-3-030-90456-2_10.

[38] Shaoquan Jiang, Guang Gong, Jingnan He, Khoa Nguyen, and Huaxiong Wang. PAKEs: New framework, new techniques and more efficient lattice-based constructions in the standard model. In *Public-Key Cryptography – PKC 2020*, pages 396–427, Cham, 2020. Springer International Publishing. https://doi.org/10.1007/978-3-030-45374-9_14.

[39] Jonathan Katz and Michael Rosenberg. Latke: A framework for constructing identity-binding PAKEs. In *Advances in Cryptology – CRYPTO 2024*, pages 218–250, Cham, 2024. Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-68379-4_7.

[40] John Kelsey, Shu jen Chang, and Ray Perlner. Sha-3 derived functions: cshake, kmac, tuplehash, and parallelhash. Technical Report NIST IR 8413, National Institute of Standards and Technology, December 2016. https://doi.org/10.6028/NIST.SP.800-185.

[41] Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, 75:565–599, 2015. https://doi.org/10.1007/s10623-014-9938-4.

[42] Arjen K. Lenstra, Jr. Hendrik W. Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982. https://www.cs.cmu.edu/afs/cs/academic/class/15451-f11/www/lectures/lect1129_LLL.pdf.

[43] Jiahao Liu, Yi Wang, Rongmao Chen, Xinyi Huang, Jinshu Su, and Moti Yung. Universally composable subversion-resilient authenticated key exchange. Cryptology ePrint Archive, Paper 2025/1521, 2025. https://eprint.iacr.org/2025/1521.

[44] Xiangyu Liu, Shengli Liu, Dawu Gu, and Jian Weng. Two-pass authenticated key exchange with explicit authentication and tight security. In *Advances in Cryptology – ASIACRYPT 2020*, pages 785–814, Cham, 2020. Springer International Publishing. https://doi.org/10.1007/978-3-030-64834-3_27.

[45] Vadim Lyubashevsky. Lattice signatures without trapdoors. In *Advances in Cryptology – EUROCRYPT 2012*, pages 738–755, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-29011-4_43.

[46] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In *Advances in Cryptology – CRYPTO 2022*, pages 71–101, Cham, 2022. Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-15979-4_3.

[47] Daniele Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Comput. Complex.*, 16(4):365–411, December 2007. https://doi.org/10.1007/s00037-007-0234-9.

[48] Chris Peikert. Limits on the hardness of lattice problems in ℓ_p norms. *computational complexity*, 17:300–351, 2008. https://doi.org/10.1007/s00037-008-0251-3.

[49] Chris Peikert. Lattice cryptography for the internet. In *Post-Quantum Cryptography*, pages 197–219, Cham, 2014. Springer International Publishing. https://doi.org/10.1007/978-3-319-11659-4_12.

[50] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing*, STOC '05, page 84–93, New York, NY, USA, 2005. Association for Computing Machinery. https://doi.org/10.1145/1060590.1060603.

[51] Tsunekazu Saito, Keita Xagawa, and Takashi Yamakawa. Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In *Advances in Cryptology – EUROCRYPT 2018*, pages 520–551, Cham, 2018. Springer International Publishing. https://doi.org/10.1007/978-3-319-78372-7_17.

[52] Peter Schwabe, Douglas Stebila, and Thom Wiggers. Post-quantum tls without handshake signatures. CCS '20, New York, NY, USA, 2020. Association for Computing Machinery. https://doi.org/10.1145/3372297.3423350.

[53] Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134. IEEE, 1994. https://doi.org/10.1109/SFCS.1994.365700.

[54] Jipeng Zhang, Junhao Huang, Lirui Zhao, Donglong Chen, and Çetin Kaya Koç. ENG25519: Faster TLS 1.3 handshake using optimized x25519 and ed25519. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 6381–6398, Philadelphia, PA, August 2024. USENIX Association. https://www.usenix.org/conference/usenixsecurity24/presentation/zhang-jipeng.

A KEM with MU-IND-CCA^{Corr} Security

In this section, we provide the standard definition of a key encapsulation mechanism (KEM), specify its correctness property, and formalize its IND-CCA security in the multi-user setting.

Definition 14 (KEM)**.** *A KEM scheme, denoted as* $\mathsf{KEM} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$*, with identity space* $\mathcal{ID}$ *and shared key space* $\mathcal{K}$*, consists of four polynomial-time algorithms:*

- $kpp \leftarrow_{\$} \mathsf{KSetup}(1^{\lambda})$*: On input* 1^{λ}*, it outputs the system parameters* kpp*.*
- $(p,s) \leftarrow_{\$} \mathsf{KGen}(kpp, id)$*: On input* kpp *and a user's identity, it outputs a public-private key pair* (p,s)*.*
- $(C,K) \leftarrow_{\$} \mathsf{Encap}(kpp, p)$*: On input* kpp *and a public key* p*, it outputs a ciphertext* C *and an shared key* $K \in \mathcal{K}$*.*
- $K \leftarrow \mathsf{Decap}(kpp, C, s)$*: On input* kpp*,* C*, and a private key* s*, it outputs the shared key* $K \in \mathcal{K}$*.*

Definition 15 (Correctness for KEM)**.** *A KEM scheme* $\mathsf{KEM} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$ *is said to be correct if, for any* $kpp \leftarrow_{\$} \mathsf{KSetup}(1^{\lambda})$*, and for any public-private key pair is generated as* $(p,s) \leftarrow_{\$} \mathsf{KGen}(kpp, id)$*, the shared keys*

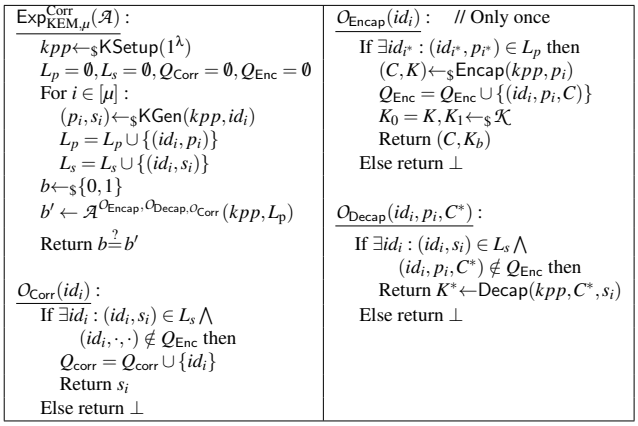

Figure 5: Security experiment for MU-IND-CCA^{Corr} secure KEM.

computed by Encap *and* Decap *agree except with negligible probability, that is,*

$$\Pr\left[K \neq K' \;\middle|\; \begin{array}{l} (C,K) = \mathsf{Encap}(kpp, p) \\ K' = \mathsf{Decap}(C, s) \end{array} \right] \leq \mathfrak{n}(\lambda).$$

In a multi-user setting, where an adversary can exploit multiple users' public keys and ciphertexts, KEM must achieve CCA-level security to mitigate potential threats arising from decapsulation queries.

Figure 5 defines the security experiment for the MU-IND-CCA^{Corr} model in a multi-user setting with $\mu \in \mathbb{N}^+$. In this model, an adversary $\mathcal{A}$ is allowed to adaptively issue a series of oracle queries to the oracles O_{Corr} and O_{Decap}, as well as a single query to O_{Encap}. At the end of the experiment, $\mathcal{A}$ outputs a bit b' as its guess for the challenge bit b. If $b' = b$, then $\mathcal{A}$ is said to win the experiment. The advantage of $\mathcal{A}$ in winning this experiment is denoted as $\mathsf{Adv}^{\mathrm{Corr}}_{\mathsf{KEM},\mu}(\mathcal{A})$.

Definition 16 (MU-IND-CCA^{Corr} Security)**.** *A KEM scheme* $\mathsf{KEM} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$ *is said to be* MU-IND-CCA^{Corr} *secure if* $\mathsf{Adv}^{\mathrm{Corr}}_{\mathsf{KEM},\mu}(\mathcal{A})$ *is negligible, i.e.,*

$$\mathsf{Adv}^{\mathrm{Corr}}_{\mathsf{KEM},\mu}(\mathcal{A}) = \left| \Pr[\mathsf{Exp}^{\mathrm{Corr}}_{\mathsf{KEM},\mu}(\mathcal{A}) \Rightarrow 1] - \tfrac{1}{2} \right|.$$

B Security Proofs

B.1 The Proof of Theorem 1

Proof. Let $\mathsf{D\text{-}KEM} = (\mathsf{KSetup}, \mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$ be a decoupled KEM scheme that satisfies correctness. Fix any public parameters $npp \leftarrow_{\$} \mathsf{NSetup}(1^\lambda)$ and any two users' identities id_i, id_j. For each $k \in \{i, j\}$, the user's key pair is generated as: $(p_k, s_k) \leftarrow_{\$} \mathsf{KGen}(npp, id_k)$, $(C_k, r_k) \leftarrow_{\$} \mathsf{Encap}_1(npp)$, and define the public and private keys as $pk_k = (p_k, C_k)$ and $sk_k = (s_k, C_k, r_k)$, respectively. We distinguish two cases based on the identity ordering.

Case 1: $id_i < id_j$. The shared keys computed by the two users are: $K_{ij} = \mathsf{Encap}_2(npp, p_i, C_j, r_j), K_{ji} = \mathsf{Decap}(npp, C_j, s_i)$. By the correctness of the underlying decoupled KEM, we have

$$\Pr\left[K_{ij} \neq K_{ji} \;\middle|\; \begin{array}{l} K_{ij} = \mathsf{Encap}_2(npp, p_i, C_j, r_j) \\ K_{ji} = \mathsf{Decap}(npp, C_j, s_i) \end{array} \right] \leq \mathfrak{n}(\lambda).$$

Case 2: $id_j \leq id_i$. The shared keys are: $K_{ij} = \mathsf{Decap}(npp, C_i, s_j)$, $K_{ji} = \mathsf{Encap}_2(npp, p_j, C_i, r_i)$. Again, the correctness of the KEM implies

$$\Pr\left[K_{ij} \neq K_{ji} \;\middle|\; \begin{array}{l} K_{ij} = \mathsf{Decap}(npp, C_i, s_j) \\ K_{ji} = \mathsf{Encap}_2(npp, p_j, C_i, r_i) \end{array} \right] \leq \mathfrak{n}(\lambda).$$

In both cases, the probability that the two users derive different shared keys is negligible, which completes the proof. □

B.2 The Proof of Theorem 3

Proof. Let a $\mathsf{HPS} = (\mathsf{HSetup}, \alpha_{(\cdot)}, \mathsf{Pub}, \mathsf{Priv})$ be a correct HPS scheme. Fix any public parameters $kpp \leftarrow_{\$} \mathsf{KSetup}(1^\lambda)$ and any two users' identities id_i and id_j. For each $k \in \{i, j\}$, the user's key pair is generated as: $s_k \leftarrow_{\$} \mathcal{HSK}$, $p_k = \alpha_{\mathsf{p}}(s_k)$, where p_k and s_k are the public and private keys associated with id_k, respectively.

Without loss of generality, assume that the user with id_i runs Encap, while the user with id_j runs Decap to derive their shared keys. In this case, the keys computed by them are $K = \mathsf{Pub}(kpp, p_j, C_i, r_i)$ and $K' = \mathsf{Priv}(kpp, s_j, C_i)$, where (C_i, r_i) is the output of Encap_1 executed by the user with id_i. By the correctness of the underlying HPS, we have

$$\Pr\left[K \neq K' \;\middle|\; \begin{array}{l} K = \mathsf{Pub}(kpp, p_j, C_i, r_i) \\ K' = \mathsf{Priv}(kpp, s_j, C_i) \end{array} \right] \leq \mathfrak{n}(\lambda).$$

This shows that the probability of deriving different shared keys is negligible, completing the proof. □

B.3 The Proof of Theorem 4

Proof. We prove the MU-IND-wCCA$^{\mathrm{Corr+}}$ security of our $\mathsf{D\text{-}KEM}_{\mathsf{HPS}}$ scheme through a sequence of games $\mathbf{G}_0$ to $\mathbf{G}_4$. Let $\Pr[\mathbf{G}_i]$ denote the probability that $\mathcal{A}$ wins in Game $\mathbf{G}_i$.

Game $\mathbf{G}_0$: At the beginning of this game, the challenger $\mathcal{C}$ sets the system parameters $kpp = \{X, L, R, \mathcal{HSK}, \mathcal{HPK}, \mathcal{H}, \mathsf{p}, \mathcal{ID}, \mathcal{K} = \Pi\}$ according to the original construction, and gives kpp to $\mathcal{A}$. $\mathcal{C}$ also initializes $t = 0$, maintains five initially empty sets: L_k, L_e, Q_{Dec}, Q_{Corr_1}, and Q_{Corr_2}, randomly selects a challenge bit $b \in \{0, 1\}$ and an index $I \in [\mu]$. For every oracle query issued by $\mathcal{A}$, $\mathcal{C}$ does the following.

$O_{\mathsf{KGen}}(id_i)$ query: If L_k contains an entry for identity id_i, $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ runs $\mathsf{KGen}(kpp, id_i)$ to obtain $s_i \in \mathcal{HSK}$ and $p_i = \alpha_{\mathsf{p}}(s_i)$, adds (id_i, p_i, s_i) to L_k, and returns p_i.

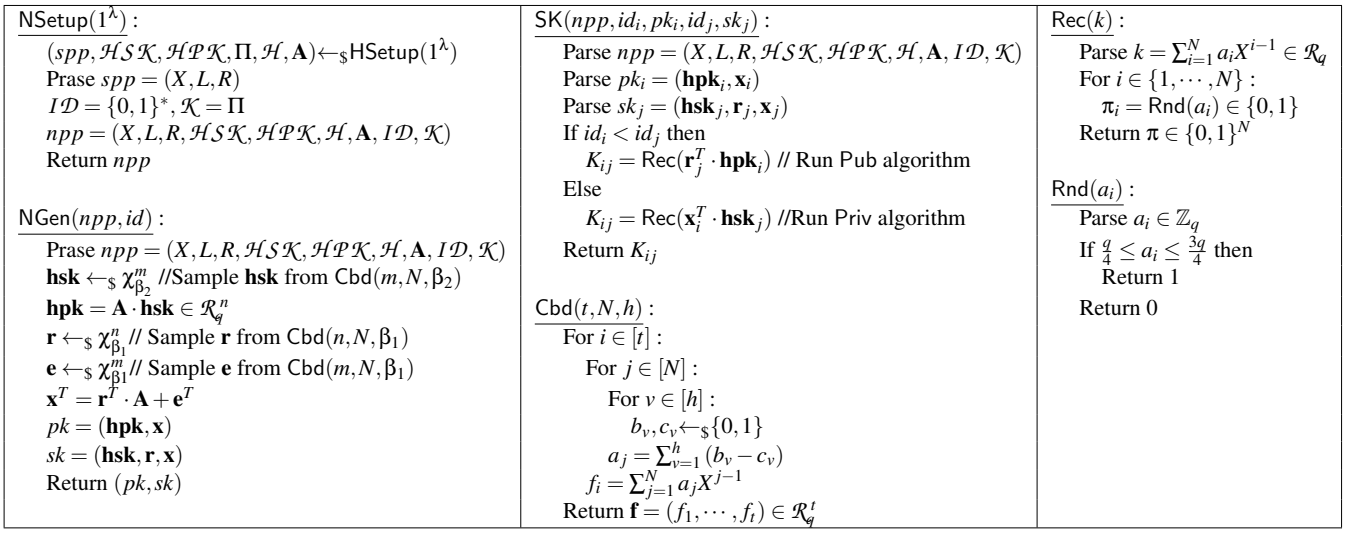

Figure 6: $\mathsf{NIKE}_{\mathsf{HPS}}$ from lattices.

$O_{\mathsf{Encap}_1}(kpp)$ query: $\mathcal{C}$ sets $t = t+1$, runs $\mathsf{Encap}_1(kpp)$ to generate $(C_t, r_t) \in R$, adds (t, C_t, r_t) to L_e, and returns C_t.

$O_{\mathsf{Encap}_2}(C_{i^*}, p_{j^*})$ query: $\mathcal{C}$ retrieves r_{i^*} from L_e using C_{i^*}, runs $\mathsf{Encap}_2(kpp, p_{j^*}, C_{i^*}, r_{i^*})$ to get $K_0 = \mathsf{Pub}(kpp, p_{j^*}, C_{i^*}, r_{i^*})$, samples $K_1 \leftarrow_\$ \mathcal{K}$, and returns (C_{i^*}, K_b).

$O_{\mathsf{Decap}}(p_j, C_i)$ query: If $(id_j, p_j, s_j) \notin L_k \vee (i, C_i, \cdot) \notin L_e$, $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ adds (C_i, p_j) to Q_{Dec}, and returns $\mathsf{Decap}(kpp, C_i, s_j)$.

$O_{\mathsf{Corr}_1}(p_j)$ query: If no tuple (id_j, p_j, s_j) exists in L_k for public key p_j, $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ adds p_j to Q_{Corr_1}, and returns s_j.

$O_{\mathsf{Corr}_2}(C_i)$ query: If no tuple (i, C_i, r_i) exists in L_e for ciphertext C_i, $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ adds C_i to Q_{Corr_2}, and returns r_i.

Finally, upon receiving $\mathcal{A}$'s guess b', $\mathcal{C}$ outputs 1 if and only if $b' = b$ under the specified winning condition $C_{i^*} \notin Q_{\mathsf{Corr}_2} \wedge p_{j^*} \notin Q_{\mathsf{Corr}_1} \wedge (C_{i^*}, p_{j^*}) \notin Q_{\mathsf{Dec}}$. According to Definition 13, this game is identical to the original experiment $\mathsf{Exp}^{\text{Corr+}}_{\text{D-KEM}_{\mathsf{HPS}}}$. Hence, we have $\mathsf{Adv}^{\text{Corr+}}_{\text{D-KEM}_{\mathsf{HPS}}}(\mathcal{A}) = \left|\Pr[\mathbf{G}_0] - \frac{1}{2}\right|$.

Game $\mathbf{G}_1$: It is the same as $\mathbf{G}_0$, except that when responding to $O_{\mathsf{Encap}_2}(C_{i^*}, p_{j^*})$, $\mathcal{C}$ retrieves s_{j^*} from L_k using p_{j^*}, runs $\mathsf{Decap}(kpp, C_{i^*}, s_{j^*})$ to obtain $K_0 = \mathsf{Priv}(kpp, s_{j^*}, C_{i^*})$. By the correctness of the $\text{D-KEM}_{\mathsf{HPS}}$ scheme, we have $\Pr[\mathbf{G}_1] = \Pr[\mathbf{G}_0]$.

Game $\mathbf{G}_2$: It is identical to $\mathbf{G}_1$, except that an abort event is introduced. Let Abort denote the event that the game aborts, and $\neg$Abort its complement. If the random index I chosen by $\mathcal{C}$ does not match the index i^* associated with the query $O_{\mathsf{Encap}_2}(C_{i^*}, p_{j^*})$ issued by $\mathcal{A}$, i.e., $I \neq i^*$, then the game aborts. Since $\mathcal{A}$ can issue at most μ queries to O_{Encap_1}, the probability that $\mathcal{C}$ guesses correctly is $\Pr[I = i^*] = \frac{1}{\mu}$. That is, $\Pr[\neg\text{Abort}] = \frac{1}{\mu}, \Pr[\text{Abort}] = 1 - \frac{1}{\mu}$. If the game does not abort, $\mathbf{G}_2$ proceeds exactly as $\mathbf{G}_1$. Otherwise, it returns 1 with probability $\frac{1}{2}$. Hence, we have $\Pr[\mathbf{G}_2] = \frac{1}{\mu} \cdot \Pr[\mathbf{G}_1] + (1 - \frac{1}{\mu}) \cdot \frac{1}{2}$.

Game $\mathbf{G}_3$: It is identical to $\mathbf{G}_2$, except that during the t-th O_{Encap_1} query, if $t = I$, $\mathcal{C}$ samples $C_I \leftarrow_\$ X \setminus L$, adds $(I, C_I, \perp)$ to L_e, and returns C_I to $\mathcal{A}$. Any non-negligible distinguishing advantage of $\mathcal{A}$ between games $\mathbf{G}_2$ and $\mathbf{G}_3$ would directly yield a non-negligible advantage for an adversary $\mathcal{B}$ against the SMP assumption, by embedding its challenge instance at index I. Specifically, if $C_I \in L$, the simulation corresponds to game $\mathbf{G}_2$; whereas if $C_I \in X \setminus L$, the simulation corresponds to game $\mathbf{G}_3$. By Definition 5, we have $|\Pr[\mathbf{G}_2] - \Pr[\mathbf{G}_3]| \le \mathsf{Adv}^{\text{SMP}}(\mathcal{B})$.

Game $\mathbf{G}_4$: This game is identical to $\mathbf{G}_3$, except that when $\mathcal{A}$ issues the $O_{\mathsf{Encap}_2}(C_{i^*}, p_{j^*})$ query, $\mathcal{C}$ replaces the shared key $K_0 = \mathsf{Priv}(kpp, s_{j^*}, C_{i^*})$ with a uniformly random key $K_0 \leftarrow_\$ \mathcal{K}$. By the ε-smooth property of the underlying HPS, for the given $C_{i^*} = C_I \in X \setminus L$, the two distributions $(C_{i^*}, p_{j^*}, K_0 = \mathsf{Priv}(kpp, s_{j^*}, C_{i^*}))$ and $(C_{i^*}, p_{j^*}, K_0 \leftarrow_\$ \mathcal{K})$ are ε-close. Hence, we have $\left|\Pr[\mathbf{G}_3] - \Pr[\mathbf{G}_4]\right| \le \varepsilon$. In $\mathbf{G}_4$, the shared key in the O_{Encap_2} query has been replaced with a uniformly random value K_b ($b \in \{0,1\}$). As a result, $\mathcal{A}$ wins with probability $\frac{1}{2}$, i.e., $\Pr[\mathbf{G}_4] = \frac{1}{2}$.

By combining the bounds derived in the previous games up to $\mathbf{G}_4$, we obtain

$$\mathsf{Adv}^{\text{Corr+}}_{\text{D-KEM}_{\mathsf{HPS}}}(\mathcal{A}) \le \mu \cdot \left(\mathsf{Adv}^{\text{SMP}}(\mathcal{B}) + \varepsilon\right),$$

which completes the proof. □

C Details of $\mathsf{NIKE}_{\mathsf{HPS}}$

In this section, we describe the full construction of $\mathsf{NIKE}_{\mathsf{HPS}}$, and then detail the parameter-selection methodology and theoretical rationale underlying our experimental evaluation.

C.1 Construction of $\mathsf{NIKE}_{\mathsf{HPS}}$

We present a concrete instantiation of $\mathsf{NIKE} = (\mathsf{NSetup}, \mathsf{NGen}, \mathsf{SK})$ with the identity space $\mathcal{ID}$ and key space $\mathcal{K}$ under the Module-LWE assumption. Our construction is obtained by transforming the lattice-based HPS into $\mathsf{D\text{-}KEM}_{\mathsf{HPS}}$ and then deriving $\mathsf{NIKE}_{\mathsf{HPS}}$, as illustrated in Figure 6.

C.2 Detailed Parameter Selection Rationale

Dimension of $\mathcal{R}_q$. We fix $N = 256$ for all instantiations so that reconciliation directly produces a 256-bit shared key. Smaller values of N would require finer-grained rounding to encode the same key length, leading to higher correctness error and increased computational cost.

Matrix A. SWOOSH is formulated with a matrix $\mathbf{A} \in \mathcal{R}_q^{m \times m}$. In contrast, both our $\mathsf{NIKE}_{\mathsf{HPS}}$ and Hesse-Lattice employ a matrix $\mathbf{A} \in \mathcal{R}_q^{n \times m}$ with $m > n$. We emphasize that n is only meaningful for our construction and that of Hesse-Lattice.

χ_{β_1} and χ_{β_2}. SWOOSH samples all secrets and errors from a single CBD with bound β, which corresponds to β_1 in our notation. In contrast, our $\mathsf{NIKE}_{\mathsf{HPS}}$ and Hesse-Lattice employ two CBDs with bounds β_1 and β_2. After fixing β_1, the parameter β_2 is selected to balance correctness and smoothness, following the bounds derived in Theorems 7 and 8.

Modulus q. Concrete security is estimated by reducing Module-LWE over $\mathcal{R}_q$ to standard LWE and evaluating the resulting instances using lattice attack cost models [13, 25]. For fixed N, n, m, and β_1, smaller values of q generally increase the hardness of the underlying lattice problem, while larger q improve correctness at the expense of higher computational cost and a larger reduction loss stemming from the ε-smooth requirement.

D Theoretical Computational Cost

Table 5: Theoretical computational cost of NIKE protocols

Protocol	NGen	SK $(id_i)^\dagger$	SK (id_j)
SWOOSH [25]‡	$2m^2T_m + 2m^2T_a$	$mT_m + (m-1)T_a + T_h$	$mT_m + (m-1)T_a + T_h$
Hesse-Lattice [36]‡	$2nmT_m + (2nm-n)T_a$	$(n+m)T_m + (n+m-1)T_a$	$(n+m)T_m + (n+m-1)T_a$
Ours‡	$2nmT_m + (2nm-n)T_a$	$nT_m + (n-1)T_a$	$mT_m + (m-1)T_a$

†: For protocols where SK cost depends on user ordering (SWOOSH and ours), we index parties by $id_i < id_j$.
‡: T_h is the cost of a hash evaluation; T_m and T_a are the costs of polynomial modular multiplication and addition over $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^N+1)$, respectively.

Table 5 summarizes the theoretical computational costs of SWOOSH, Hesse-Lattice, and our protocol. In all three constructions, shared-key derivation (SK) also need to perform rounding of the shared key, however, its runtime is negligible. We therefore omit this cost for all protocols in the theoretical analysis.