Swoosh: Practical Lattice-Based Non-Interactive Key Exchange

Phillip Gajland
Max Planck Institute
for Security and Privacy
Ruhr-University Bochum
phillip.gajland@mpi-sp.org

Bor de Kock
NTNU – Norwegian University of
Science and Technology
bor.dekock@ntnu.no

Miguel Quaresma
Max Planck Institute
for Security and Privacy
miguel.quaresma@mpi-sp.org

Giulio Malavolta
Max Planck Institute
for Security and Privacy
giulio.malavolta@mpi-sp.org

Peter Schwabe
Max Planck Institute
for Security and Privacy
Radboud University
peter@cryptojedi.org

ABSTRACT

The advent of quantum computers has generated a wave of interest for post-quantum cryptographic schemes, as a replacement for currently used cryptographic primitives. In this context, lattice-based cryptography has emerged as the leading paradigm to build post-quantum cryptography. However, all viable replacements of the classical Diffie-Hellman key exchange require additional rounds of interactions, thus failing to achieve all the benefits of this protocol. Although earlier work has shown that lattice-based Non-Interactive Key Exchange (NIKE) is theoretically possible, it has been considered too inefficient for real-life applications.

In this work, we provide the first evidence against this folklore belief. We construct a practical lattice-based NIKE whose security is based on the standard module learning with errors (M-LWE) problem in the quantum random oracle model. Our scheme is obtained in two steps: (i) A passively-secure construction that achieves a strong notion of correctness, coupled with (ii) a generic compiler that turns any such scheme into an actively-secure one. To substantiate our efficiency claim, we present an optimised implementation of our construction in Rust and Jasmin, demonstrating its applicability to real-world scenarios. For this we obtain public keys of approximately 220 KBs and the computation of shared keys takes than 12 million cycles on an Intel Skylake CPU at a post-quantum security level of more than 120 bits.

1 INTRODUCTION

A key exchange is a cryptographic primitive that allows two users to agree on a common secret key over an insecure channel, such as the Internet. If the protocol consists of a single, asynchronous message from each party, then we refer to it as a *Non-Interactive Key Exchange* (NIKE). The seminal work of Diffie and Hellman [34] introduced the well-known NIKE scheme that marked the birth of public-key cryptography; each party sends a single group element g^x (or g^y, respectively) and the shared key can be derived by computing $(g^y)^x = (g^x)^y$. From a theoretical stand-point NIKE implies the existence of public key encryption (PKE), key encapsulation mechanism (KEM), and even authenticated key-exchange (AKE) when combining the results of [23] with [41]. Moreover, in practice, the Diffie-Hellman key exchange lies at the heart of protocols such as Transport Layer Security (TLS) [71], the Signal protocol, or the Noise protocol framework [66].

The looming threat of quantum computers, combined with the discovery of efficient quantum algorithms for factoring integers and computing discrete logarithms [75], has motivated the cryptographic community to explore solutions based on new mathematical structures, departing from protocols based on the Diffie-Hellman key exchange. In particular, lattice-based cryptography [70] has emerged as the leading paradigm for constructing *post-quantum* cryptographic schemes. As a result of NIST's recent standardisation process, three, of the four algorithms that were selected for standardisation, are lattice-based [58, 69, 72].

While efficient lattice-based key exchange protocols exist [5, 21, 72], they are all qualitatively different from the standard Diffie-Hellman-style key exchange, in the sense that they require *additional rounds of interaction*. For many applications, where interaction is already built-in, these protocols are perfectly fine substitutes for the Diffie-Hellman (that is not post-quantum secure). However, in many scenarios of interest, the non-interactive nature of NIKE protocols is crucial (we discuss concrete examples in further detail in Section 1.1). Unfortunately, despite almost two decades of research on the subject, an efficient lattice-based NIKE remains elusive. Perhaps more worryingly, a recent work [46] has shown theoretical barriers on the efficiency of lattice-based NIKE, calling into question whether it is even possible to build a practical scheme at all. Thus, the current state of affairs, leaves open the following question:

Is lattice-based *non-interactive* key exchange feasible in practice?

In our work we seek to answer this question in the affirmative, and show that lattice-based NIKE can be made efficient enough to be used in practice, whilst maintaining post-quantum security.

1.1 NIKE vs. KEMs

While the Diffie-Hellman (DH) key exchange happens to be non-interactive, most post-quantum approaches to key exchange are interactive key-encapsulation mechanisms (KEMs). Intuitively, the difference is as follows. In a NIKE, any user A can use their secret key sk_A together with the public key pk_B of a user B to derive a shared key k_{AB}. At the same time, and without interaction with A, user B can compute the same shared key k_{AB} by combining their secret key sk_B with the public key pk_A of user A. However, in a KEM, this key-derivation becomes a two-stage, inherently asymmetric and interactive process. First, A invokes an encapsulation routine that accepts pk_B as input and produces as output the shared key k_{AB}, and a ciphertext ct, which they send to B. User B then invokes the decapsulation routine that takes as input ct and secret key sk_B to produce the same shared secret k_{AB}.

Some protocols employing DH do not actually make use of the non-interactive nature and can thus be migrated to post-quantum KEMs in a straight-forward manner. Probably the most prominent example is TLS, which uses the DH key exchange with ephemeral keys on both sides for forward secrecy and has been updated to offer post-quantum security by using KEMs in multiple papers [15, 22, 64] and real-world deployments [53–55, 78].

Other protocols *do* make use of the non-interactive nature of DH and their migration to post-quantum primitives is thus much more involved. A common pattern in these protocols is that they use static DH keys for authentication. One example is OPTLS by Krawczyk and Wee [52], a proposal that eliminates the need for handshake signatures in TLS. The idea was picked up in the post-quantum setting in the KEMTLS proposal by Schwabe, Stebila, and Wiggers [73]. Like OPTLS, also KEMTLS eliminates the need for handshake signatures, but unlike OPTLS uses static KEM keys for authentication. This comes at the expense of requiring more communication round-trips until full server authentication is achieved. This can be problematic for some protocols like HTTPS that allow the server to send early payload data before the handshake is finished. Similar issues with delayed authentication when moving from DH to KEMs were identified in the migration of WireGuard to the post-quantum setting in [48] and in the the recently proposed post-quantum version of the Noise protocol framework [9].

These examples all manage to migrate from the DH setting to the KEM setting at the cost of further communication round trips, and without having to move to signature-based authentication or engaging even more involved cryptographic primitives. If communicating parties cannot be assumed to be online at the same time, this approach is doomed to fail. A prominent example of precisely this asynchronous communication setting is the Signal secure-messaging protocol and specifically the X3DH protocol [60] that is invoked when a user A starts their communication with a (possibly offline) user B. The X3DH protocol uses a combination of ephemeral, static, and semi-static DH keys to achieve forward secrecy, mutual authentication, and offline deniability without the need for direct interaction between A and B. There have been multiple attempts to migrate X3DH to the post-quantum setting [25, 26, 36, 47, 76] but they all either assume the existence of a reasonably efficient post-quantum NIKE, or fail to achieve the same security and privacy as the pre-quantum version from a single simple asymmetric primitive.

1.2 Our Contributions

In this work, we demonstrate the practical feasibility of lattice-based *non-interactive key exchange*. We propose a new scheme, that we call "Swoosh", based on the hardness of the M-LWE problem. We show a proof of its security, both in the passive and active setting, and provide parameter sets for the former with over 120-bits of security against quantum adversaries (using the best known attacks that incorporate recent advances in lattice cryptanalysis). Our contributions can be succinctly summarised as follows.

(1) We propose a new construction of NIKE based on the hardness of the M-LWE problem. Our construction is based on the standard template [35, 57], but with a new tweak that allows us to prove a strong notion of correctness (which, in turn, is necessary to achieve active security) in the quantum random oracle model (QROM). Somewhat interestingly, our use of the random oracle appears to be different from the Fiat-Shamir [40] and the Fujisaki-Okamoto [42, 43] transformations, and may thus be of independent interest.

(2) We propose a compiler to generically lift a passively secure NIKE to an actively secure scheme, using non-interactive zero-knowledge (NIZK) proofs. While this approach is folklore, to the best of our knowledge it has never appeared explicitly in the literature. Furthermore, the exact notion of passive security needed for the proof to go through, turns out to be surprisingly subtle to identify.

(3) We carefully select parameters for the passively secure NIKE and instantiate the scheme with parameters achieving 120 bits of security against quantum adversaries The resulting scheme we call "Passive-Swoosh" and the full scheme including the NIZK "Swoosh".

(4) We provide an implementation of Passive-Swoosh written in a combination of Rust and Jasmin, and show that the public keys of Passive-Swoosh are smaller than the ones of the smallest parameter set of Classic McEliece [1], an interactive KEM selected for round 4 of the NIST-PQC competition. We also demonstrate that in terms of speed, Passive-Swoosh outperforms CSIDH [30], the only currently known (and realistic) post-quantum NIKE, by orders of magnitude.

1.3 Related work

Post-quantum NIKE. While interactive KEMs appear to be much more efficient in a post-quantum world than NIKEs, this does not mean that there are no previous proposals for post-quantum NIKE. In [19], Boneh and Zhandry show a construction using iO to construct a multiparty NIKE from pseudorandom generators. Given the impractical performance of iO, the result is mainly of theoretical interest. Much more practical was supersingular-isogeny Diffie-Hellman (SIDH) [32, 50]. However, in 2016, this construction was shown to be susceptible to active attacks [44]. This could be solved by employing the Fujisaki-Okamoto transform [42] in the NIST PQC candidate SIKE [49], but this came at the expense of turning the NIKE into an interactive KEM. Another approach to restoring the active security of SIKE was presented in [10]. This approach preserved the non-interactive nature of SIDH, but required many parallel protocol executions and thus massively increased computation time and message sizes. In 2022, all of these approaches based on

SIDH were made obsolete by the Castryck-Decru attack against SIDH [29].

In 2018, Castryck, Lange, Martindale, Panny, and Renes proposed CSIDH, a different approach for constructing an isogeny-based NIKE [30]. CSIDH is not affected by the Castryck-Decru attack, and is arguably the most plausible candidate for practical post-quantum NIKE thus far, although the post-quantum security of concrete parameters is subject of debate [16, 20, 65]. Multiple works have considered the efficient and secure implementation of CSIDH, currently the fastest approach is a variant called CTIDH [11]. We provide a performance comparison of our proposal to CTIDH in Section 6.2.

Lattice-based NIKE. The idea of lattice-based NIKE using the approach we use for Swoosh is not new; in [57] Lyubashevsky calls it *"folklore (since at least 2010)"*. An attempt at selecting parameters was made in [33]. However, the proposed scheme did not formally consider passive security, nor active security. Furthermore, the selected parameters resulted in a correctness error that would not even allow the transformation into an actively secure scheme through the use of NIZK proofs that we use for Swoosh.

In fact, prior to our work, lattice-based NIKE was widely considered impractical and this was even substantiated by theoretical evidence. The work of [46] discovered information theoretic barriers in constructing lattice-based NIKE with non-interactive reconciliations. In particular, they showed that any natural candidate of lattice-based NIKE with polynomial modulus-to-noise ratio would necessarily incur an inverse-polynomial correctness error. However, we stress that our work does not contradict the theorem of [46]. As the authors of [46] observe, non-interactive reconciliation is possible, if we consider (M-)LWE instances with *super-polynomial* modulus-to-noise ratio. This is indeed the regime of parameters that we adopt in our work.

2 TECHNICAL OUTLINE

We give a self-contained overview of our approach for constructing a fast lattice-based NIKE. The following is somewhat informal and glosses over many important details, as it is only intended for an intuitive understanding of our approach. The reader is referred to the respective technical sections for precise statements.

The Basic Blueprint. Before delving into the specifics of our approach, it is useful to recall the folklore construction of lattice-based key exchange between Alice and Bob. Let $\boldsymbol{A}$ be a random public $N \times N$ matrix over some ring $\mathcal{R}_q$ and χ a noise distribution. The protocol proceeds as follows; Alice samples $\vec{s}_1$ and $\vec{e}_1$ from χ^N, and computes her public key as $\vec{s}_1^\top \boldsymbol{A} + \vec{e}_1^\top$. Bob samples an independent $\vec{s}_2$ and $\vec{e}_2$ from χ^N, and computes his public key as $\boldsymbol{A}\vec{s}_2 + \vec{e}_2$. After asynchronously obtaining each other's public keys, Alice and Bob can compute an approximate shared key as

$$\vec{s}_1^\top \left(\boldsymbol{A}\vec{s}_2 + \vec{e}_2\right) \approx \left(\vec{s}_1^\top \boldsymbol{A} + \vec{e}_1^\top\right) \vec{s}_2.$$

A simple calculation shows that the shared keys computed by both parties are identical with the exception of the error terms $\vec{s}_1^\top \vec{e}_2$ and $\vec{e}_1^\top \vec{s}_2$ for Alice and Bob, respectively. To correct these errors, known schemes in the literature are based on encryption or interactive *reconciliation*, which can be realised quite efficiently. However, if we insist on a NIKE protocol, no further interaction is allowed, and Alice and Bob must correct the errors locally. That is, we need to devise a *non-interactive* reconciliation function Rec such that

$$\mathsf{Rec}\left(\vec{s}_1^\top \left(\boldsymbol{A}\vec{s}_2 + \vec{e}_2\right)\right) = \mathsf{Rec}\left(\left(\vec{s}_1^\top \boldsymbol{A} + \vec{e}_1^\top\right) \vec{s}_2\right).$$

Note that, thus far, we have assumed that both Alice and Bob compute their keys according to the specification of the protocol, i.e., we implicitly only considered passive attacks. However, for the security of the final scheme, it will be necessary to handle parties that may behave arbitrarily. In what follows, we show how we tackle these two challenges separately, in a way that preserves the efficiency and security of the scheme.

Challenge I: Non-Interactive Reconciliation. A natural approach for correcting the errors introduced by the noise terms, is to derive the key by *rounding* the coefficients of the resulting ring element. In fact this is the approach that we adopt in this work, however there are still new ideas required to simultaneously achieve all of the following objectives: (i) security from the hardness of the standard module learning with errors (M-LWE) problem, (ii) reducing the correctness error to negligible, and (iii) maintaining the concrete efficiency of the construction. Here, we stress that a negligible correctness error is not just a matter of convenience, but that a non-negligible correctness error translates to an attack against the scheme: Loosely speaking, this is because the attacker can observe whenever the key agreement fails, therefore learning some information about the secret key of the honest party. Let us now focus on making the rounding approach work for non-interactive reconciliation. A simple calculation shows that the error terms cause a correctness error, only when the term $\vec{s}_1^\top \boldsymbol{A}\vec{s}_2$ falls into a *danger interval*

$$S^* = \left[\frac{q}{4} \pm \beta^2 dN\right] \cup \left[\frac{3q}{4} \pm \beta^2 dN\right],$$

where β is a bound on the norm of the noise distribution and d is the degree of $\mathcal{R}_q$. It is tempting to conclude that, if q is sufficiently large, then this event only happens with negligible probability. However, this analysis is imprecise as it does not take into account *adaptive* attacks, where the adversary chooses their secret key intentionally to make this event more likely. To prevent this, and obtain a provably secure scheme, we add a *random shift* $\boldsymbol{r}$ to the term $\vec{s}_1^\top \boldsymbol{A}\vec{s}_2$ to ensure that their sum $\vec{s}_1^\top \boldsymbol{A}\vec{s}_2 + \boldsymbol{r}$ is indeed uniformly distributed in $\mathcal{R}_q$. Note that such $\boldsymbol{r}$ does not need to be kept private, although it is important that it is sampled independently of the keys. Our idea is to sample $\boldsymbol{r}$ as the output of a hash function (modelled as a random oracle) on input the two public keys. This allows us to achieve two goals simultaneously:

- Both parties can recompute the shift $\boldsymbol{r}$ without the need of further interaction.
- We can show that $\vec{s}_1^\top \boldsymbol{A}\vec{s}_2 + \boldsymbol{r}$ is indeed uniformly sampled, even if the adversary has quantum access to the random oracle.

In summary, we are able to build a non-interactive reconciliation mechanism so that the scheme is provably secure (in the passive settings) against the standard M-LWE assumption, in the QROM. In fact, we are also able to show a strong notion of correctness, namely that the adversary cannot cause a reconciliation error, *even if it is allowed to choose both secret keys.* This strong notion of correctness will be useful when lifting the scheme to the active setting.

Challenge II: From Passive to Active Security. The above discussion concerns keys that are guaranteed to be well-formed (passive security). However, in real-world scenarios we have to deal with attackers that can behave arbitrarily. In the stronger notion of *active security* [28, 41] the adversary is given access to various oracles that allow him to register honest keys, register corrupt keys (ones to which he does not know the corresponding public key), or reveal the shared key between an honest key and a corrupted one. Ultimately the adversary wins if he can distinguish between a random key and a shared key, that was derived from two honestly generated key pairs.

In order to prove the active security of our scheme we present a *compiler* that generically lifts our scheme to the active setting using non-interactive zero-knowledge (NIZK) proofs. Here it is crucial that our scheme satisfies the aforementioned strong notion of correctness, since the only thing that the NIZK guarantees is that the keys are in the support of the honest distributions, but otherwise they may be chosen arbitrarily. For technical reasons, we require a NIZK that satisfies the strong property of simulation-sound online-extractability. We refer the reader to Section 5 for more details.

Putting Everything Together. Overall, we obtain a passively secure construction in the QROM assuming the hardness of the Module-LWE (M-LWE) problem (for the active settings, we additionally require a NIZK proof). Compared to Ring-LWE (R-LWE), M-LWE gives us greater flexibility over the choice of parameters, when implementing our scheme. However, this introduces an additional complication: Unlike the case for R-LWE, where single polynomials are considered and their multiplication is commutative, in the case of M-LWE we work with matrices where the matrix multiplication is not generally commutative. For the general case of two parties without predefined roles in a protocol, there is no way to know ahead of time whether to left multiply or right multiply. This means that each public key is effectively duplicated by adding a left multiplied key and right multiplied key. However, we argue that in many cases, when parties have predefined roles in a protocol, such as a server or client, this issue can be resolved (the server could "go right" and the client "left" or vice versa). We defer a more detailed discussion of this to Section 5.3.

Our parameters are selected as to provide more than 120 bits of post-quantum security, taking into account recent advances in lattice cryptanalysis. We work over the ring $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$ with $d = 256$. Along with our public matrix $\boldsymbol{A} \in \mathcal{R}_q^{N \times N}$, where $N = 32$, this gives us a lattice dimension of 8192. In order to reduce the correctness error to reasonable levels, q had to be sufficiently large. We choose $q = 2^{214} - 255$, a prime that is simultanously NTT-friendly and close to a power-of-two making for more efficient field arithmetic. Furthermore, we use ternary noise sampled from a centred binomial distribution, for the sake of efficiency.

Finally, we provide an open-source implementation of Passive-Swoosh in Rust and Jasmin, which employs numerous optimisations rendering competitive benchmarks. Due to the modular fashion of our implementation we note that it can easily be tailored to use different parameters or be incorporated with suitable NIZKs. We defer a more detailed discussion to Section 6.

3 PRELIMINARIES

In this Section we introduce our notation and review some quantum preliminaries along with the relevant lattice-based hardness assumptions.

3.1 Notation

We start by defining some standard notation used throughout the paper.

Sets, Vectors, Polynomials and Norms. For integers a, b, where $a < b$, $[a, b]$ denotes the set $\{a, a+1, \ldots, b\}$. For any positive $\beta \in \mathbb{Z}$, we define the set $[\beta] := \{-\beta, \ldots, -1, 0, 1 \ldots, \beta\}$, and let $x \xleftarrow{\$} \mathcal{S}$ denote the uniform sampling of x from the set $\mathcal{S}$. Let $\mathbb{Z}_q$ denote the ring of integers modulo a prime q. We define $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$ to be the ring of integer polynomials modulo $X^d + 1$, for d a power of 2, and $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$ the ring of integer polynomials modulo $X^d + 1$ where each coefficient is reduced modulo q. We assume that that, for any N, a uniformly sampled N-dimensional square matrix over $\mathcal{R}_q$ is invertible with probability c, for some constant $0 < c \leq 1$. For concreteness, we conservatively set this constant to be $c = 0.5$. Bold upper case letters $\boldsymbol{A}$ and bold lower case letters with arrows $\vec{\boldsymbol{a}}$ denote matrices and column vectors over $\mathcal{R}_q$, respectively; for row vectors we use the transpose $\vec{\boldsymbol{b}}^\top$.

For a polynomial $\boldsymbol{f} \in \mathcal{R}_q$, let $\vec{f} \in \mathbb{Z}_q^d$ denote the coefficient vector of $\boldsymbol{f}$, and $f_i \in \mathbb{Z}_q$ the i^{th} coefficient. However, we denote the constant coefficient by $\tilde{f} := f_0 \in \mathbb{Z}_q$. For an element $f_i \in \mathbb{Z}_q$, we write $|f_i|$ to mean $|f_i \bmod q|$. Let the ℓ_∞ and ℓ_p norms for $\boldsymbol{f} = f_0 + f_1 X + \ldots + f_{d-1} X^{d-1} \in \mathcal{R}_q$ be defined as

$$\|\boldsymbol{f}\|_\infty := \max_{0 \leq i \leq d-1} |f_i| \quad \text{and} \quad \|\boldsymbol{f}\|_p := \sqrt[p]{\sum_{i=0}^{d-1} |f_i|^p},$$

respectively. If $\vec{\boldsymbol{f}} = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_k) \in \mathcal{R}_q^k$, then

$$\left\|\vec{\boldsymbol{f}}\right\|_\infty := \max_{1 \leq i \leq k} \|\boldsymbol{f}_i\|_\infty \quad \text{and} \quad \left\|\vec{\boldsymbol{f}}\right\|_p := \sqrt[p]{\sum_{i=1}^{k} \|\boldsymbol{f}_i\|_p^p}.$$

By default $\left\|\vec{\boldsymbol{f}}\right\| := \left\|\vec{\boldsymbol{f}}\right\|_2$.

Probabilities, Algorithms and Games. The support of a discrete random variable X is defined as

$$\sup(X) := \{x \in \mathbb{R} : \Pr[X = x] > 0\}.$$

Algorithms are denoted by upper-case letters in sans-serif font, such as A and B. Unless otherwise stated all algorithms are probabilistic and $(x_1, \ldots) \xleftarrow{\$} \mathsf{A}(y_1, \ldots)$ is used to denote that A returns $(x_1, \ldots)$ when run on input $(y_1, \ldots)$. When A has oracle access to B during its execution, this is denoted by A^{B}. For a probabilistic algorithm A, the notation $x \in \mathsf{A}(y)$ denotes that x is a possible output of A on input y. We use code-based security games [13], where $\Pr[\mathsf{G} \Rightarrow 1]$ denotes the probability that the final output of game G is 1. The notation $[\![B]\!]$, where B is a Boolean statement, refers to a bit that is 1 if the statement is true and 0 otherwise.

3.2 Quantum Preliminaries

We review some quantum preliminaries as stated in [37].

Qubits, n-qubit States and Measurement. A qubit $|x\rangle := \alpha_0 |0\rangle + \alpha_1 |1\rangle$ is a unit vector in some Hilbert space $\mathcal{H}$. When $\alpha_0 \neq 1$ and $\alpha_1 \neq 1$, we say that $|x\rangle$ is in *superposition*. An n-bit quantum register $|x\rangle := \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$ is a unit vector in $\mathcal{H}^{\otimes n} \cong \mathbb{C}^{2^n}$, that is $\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1$ for $\alpha_i \in \mathbb{C}$. We call the set $\{|0\rangle, |1\rangle, \dots, |2^n-1\rangle\}$ the *computational basis* and say that $|x\rangle$ is *entangled* when $|x\rangle$ cannot be written as the tensor product of single qubits. Unless otherwise stated, measurements are done in the computational basis. After measuring a quantum register $|x\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$ in the computational basis, the state *collapses* and $|x\rangle = \pm |i\rangle$ with probability $|\alpha_i|^2$.

Quantum Algorithms. A quantum algorithm A is a sequence of unitary operations U_i, where unitary operations are defined to map unit vectors to unit vectors, while preserving the normalisation constraint of quantum registers. A quantum oracle algorithm A^{O} is defined analogously, and can additionally query the oracle O before (or after) executing a unitary U_i. As quantum computations need to be reversible, we model an oracle $\mathsf{O} : X \to Y$ by a unitary U_{O} that maps $|x\rangle |y\rangle \mapsto |x\rangle |y \oplus \mathsf{O}(x)\rangle$. For an oracle O, we write $|\mathsf{O}\rangle$ to denote that an algorithm has quantum-access to U_{O}.

Quantum Random Oracle Model. In the random oracle model [12], all parties have access to a uniformly sampled random function H. Since quantum adversaries can evaluate hash functions in superposition, we model quantum adversaries to have quantum access to random oracles [18]. Specifically, we assume that all algorithms have access to the unitary implementing the mapping:

$$|x\rangle |y\rangle \mapsto |x\rangle |y \oplus \mathsf{H}(x)\rangle$$

where H is a uniformly sampled random function.

Query Depth and Query Parallelism. As in the work of [8] we consider the query depth D of an adversary making a total of Q_{H} random oracle queries. This is important in practice because for highly parallel adversaries we have $D \ll Q_{\mathsf{H}}$. By setting $D := Q_{\mathsf{H}}$ we obtain the bounds for sequential adversaries. We will use the following technical Lemma from [8].

Lemma 3.1 (Search in unstructured functions [8, Lem. 2]). *Let* H *be a random function drawn from a distribution such that* $\Pr[\mathsf{H}(x) = 1] \leq \lambda$ *for all* x. *Let* A *be an adversary with query depth* D*, making at most* Q_{H} *many queries to* H. *Then*

$$\Pr\left[\mathsf{H}(x) = 1 : b \xleftarrow{\$} \mathsf{A}^{\mathsf{H}}\right] \leq 4 \cdot (D+2) \cdot (Q_{\mathsf{H}} + 1) \cdot \lambda.$$

3.3 Hardness Assumptions

The security of our scheme relies on the following variant of the Module-Learning With Errors (M-LWE) [56, 70].

Definition 3.2 ($\mathbf{M\text{-}LWE}_{q,n,m,\chi}$). The decisional *Module-Learning With Errors* problem (in its Hermite normal form) with parameters $n, m > 0$ and an error distribution χ over $\mathcal{R}_q$ is defined via the game $\mathbf{M\text{-}LWE}^b_{q,n,m,\chi}$ depicted in Figure 1. Here, $\mathbf{M\text{-}LWE}^b_{q,n,m,\chi}$ is parameterised by a bit b. We define A's advantage in $\mathbf{M\text{-}LWE}^b_{q,n,m,\chi}$ as

$$\mathrm{Adv}^{\mathbf{M\text{-}LWE}}_{q,n,m,\chi}(\mathsf{A}) := \left| \begin{array}{l} \Pr[\mathbf{M\text{-}LWE}^{0,\mathsf{A}}_{q,n,m,\chi} \Rightarrow 1] \\ -\Pr[\mathbf{M\text{-}LWE}^{1,\mathsf{A}}_{q,n,m,\chi} \Rightarrow 1] \end{array} \right|,$$

and say that $\mathbf{M\text{-}LWE}_{q,n,m,\chi}$ is ϵ-hard for all adversaries A satisfying $\mathrm{Adv}^{\mathbf{M\text{-}LWE}}_{q,n,m,\chi}(\mathsf{A}) \leq \epsilon$.

```
Game M-LWE^b_{q,n,m,χ}                Oracle RoR(b)                      / Once
$b' \xleftarrow{\$} \mathsf{A}^{\mathsf{RoR}(b)}$     if $b = 0$ :
return $[\![b = b']\!]$                    $\boldsymbol{A} \xleftarrow{\$} \mathcal{R}_q^{n \times m}$
                                        $\vec{\boldsymbol{s}} \xleftarrow{\$} \chi^m$
                                        $\vec{\boldsymbol{e}} \xleftarrow{\$} \chi^n$
                                        return $(\boldsymbol{A}, \boldsymbol{A}\vec{\boldsymbol{s}} + \vec{\boldsymbol{e}})$
                                    elseif $b = 1$ :
                                        $\boldsymbol{A} \xleftarrow{\$} \mathcal{R}_q^{n \times m}$
                                        $\vec{\boldsymbol{u}} \xleftarrow{\$} \mathcal{R}_q^n$
                                        return $(\boldsymbol{A}, \vec{\boldsymbol{u}})$
```

Figure 1: Game defining $\mathbf{M\text{-}LWE}^b_{q,n,m,\chi}$ with adversary A.

Theoretic treatments of LWE-based schemes typically consider the modulus to be polynomial in n and χ to be the discrete Gaussian on $D_{\mathbb{Z},\alpha \cdot q}$ over $\mathbb{Z}$ with mean 0 and standard deviation $\sigma = \alpha \cdot q / \sqrt{2\pi}$ for some $\alpha < 1$. For these choices the work of [24, 70] showed that if $\alpha q > 2\sqrt{n}$ then worst-case GapSVP-$\tilde{O}(n/\alpha)$ reduces to average-case LWE. As such, many early implementations sampled from a discrete Gaussian distribution, which turns out to be either fairly inefficient [22] or vulnerable to timing attacks [27, 39, 67]. Furthermore, the performance of the best known attacks against LWE-based schemes does not depend on the exact distribution of noise, but rather on the standard deviation (and potentially the entropy). This motivates the use of noise distributions that we can easily, efficiently, and securely sample from. One example is the centred binomial distribution used by CRYSTALS-Kyber [72] and in [5].

4 DEFINITIONS

In this section we present a formal definition of a non-interactive key exchange along with its security notions. A precise definition of non-interactive zero-knowledge proofs can be found in Appendix A.1.

4.1 Non-Interactive Key Exchange

Following the work of [28, 41], we formally define a non-interactive key exchange (NIKE). Through the use of IDs, the security model proposed in [28] abstracts away all considerations concerning certification and public-key infrastructure.

Definition 4.1 (Non-Interactive Key Exchange). A non-interactive key exchange NIKE is defined as a tuple NIKE := (Stp, Gen, SdK) of the following PPT algorithms. Furthermore, we define an identity space $\mathcal{IDS}$ and a shared key space $\mathcal{SKS}$.

$par \xleftarrow{\$} \mathsf{Stp}(1^\lambda)$**:** Given the security parameter 1^λ (encoded in unary), the probabilistic setup algorithm returns a set of system parameters par.

$(sk, pk) \xleftarrow{\$} \mathsf{Gen}(\mathtt{ID})$**:** Given an identity $\mathtt{ID} \in \mathcal{IDS}$, the probabilistic key generation algorithm Gen returns a secret/public key pair (sk, pk).

$k \leftarrow \mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2)$**:** Given an identity $\mathtt{ID}_1 \in \mathcal{IDS}$ and its corresponding public key pk_1 along with another identity $\mathtt{ID}_2 \in \mathcal{IDS}$ and its corresponding secret key sk_2, the deterministic shared key establishment algorithm SdK returns a shared-key $k \in \mathcal{SKS}$, or a failure symbol $\perp$. We assume that SdK always returns $\perp$ if $\mathtt{ID}_1 = \mathtt{ID}_2$.

Correctness. Informally, *honest correctness* states that shared keys derived by two honest parties should be the same with overwhelming probability. Although our subsequent definition of correctness implies honest correctness, we state both definitions here for completeness.

Definition 4.2 (Honest Correctness). A non-interactive key exchange NIKE := (Stp, Gen, SdK) has *correctness error* δ (or is said to be δ-*correct*), if for all $par \in \mathsf{Stp}(1^\lambda)$ and $\mathtt{ID}_1, \mathtt{ID}_2 \in \mathcal{IDS}$ it holds that,

$$\Pr\left[\mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2) \neq \mathsf{SdK}(\mathtt{ID}_2, pk_2, \mathtt{ID}_1, sk_1) \;\middle|\; \begin{array}{l}(sk_1, pk_1) \xleftarrow{\$} \mathsf{Gen}(\mathtt{ID}_1)\\ (sk_2, pk_2) \xleftarrow{\$} \mathsf{Gen}(\mathtt{ID}_2)\end{array}\right] \leq \delta,$$

where the probability is taken over the random choices of Stp and Gen.

In this work we define a stronger notion, *semi-malicious correctness* that captures the property that two maliciously chosen key pairs (that are in the support of the key-generation algorithm) will not cause the key exchange to fail. Since this property clearly implies honest correctness, throughout the rest of this work we only focus on semi-malicious correctness. We formalise *semi-malicious correctness* for NIKE relative to a random oracle H via the game $\mathbf{SM\text{-}COR}_{\mathsf{NIKE}}$ depicted in Figure 2 and define the advantage of an adversary A in $\mathbf{SM\text{-}COR}_{\mathsf{NIKE}}$ as

$$\mathrm{Adv}^{\mathbf{SM\text{-}COR}}_{\mathsf{NIKE},par}(\mathsf{A}) := \Pr[\mathbf{SM\text{-}COR}^{\mathsf{A}}_{\mathsf{NIKE}} \Rightarrow 1].$$

Definition 4.3 (Semi-malicious Correctness). Let NIKE := (Stp, Gen, SdK) be a non-interactive key exchange. In the quantum random oracle model, we say that NIKE is $\delta(Q_{\mathsf{H}}, D)$-**SM-COR** if for all $\mathtt{ID}_1, \mathtt{ID}_2 \in \mathcal{IDS}$ and for all (possibly unbounded) adversaries A of depth at most D, making at most Q_{H} queries (possibly in superposition) to the random oracle H, we have $\mathrm{Adv}^{\mathbf{SM\text{-}COR}}_{\mathsf{NIKE},par}(\mathsf{A}) \leq \delta(Q_{\mathsf{H}}, D)$. [1]

[1]Note that in the standard model our correctness definition can be considered a special case where the number of random oracle queries is zero and hence $\delta(Q_{\mathsf{H}}, D)$ is a constant.

Game $\mathbf{SM\text{-}COR}_{\mathsf{NIKE}}$

```
01 $par \leftarrow \mathsf{Stp}(1^\lambda)$
02 $\left.\begin{array}{l}\mathrm{supp}(\mathsf{Gen}(\mathtt{ID}_1)) \ni (sk_1, pk_1)\\ \mathrm{supp}(\mathsf{Gen}(\mathtt{ID}_2)) \ni (sk_2, pk_2)\end{array}\right\} \xleftarrow{\$} \mathsf{A}^{|\mathsf{H}\rangle}(par)$
03 return $[\![\mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2) \neq \mathsf{SdK}(\mathtt{ID}_2, pk_2, \mathtt{ID}_1, sk_1)]\!]$
```

Figure 2: Correctness game $\mathbf{SM\text{-}COR}_{\mathsf{NIKE}}$ for a non-interactive key exchange NIKE defined relative to a random oracle H with adversary A.

Passive Security. We formalise the notion of key indistinguishability with *passive security* for a non-interactive key exchange NIKE, with respect to system parameters $par \in \mathsf{Stp}(1^\lambda)$ via the game $\mathbf{PasSec}^b_{\mathsf{NIKE},par}$ depicted in Figure 3. In $\mathbf{PasSec}^b_{\mathsf{NIKE},par}$, the adversary A provides two identities $\mathtt{ID}_1$ and $\mathtt{ID}_2$ for which the public and secret keys are derived honestly. Given both public keys, A has to distinguish the shared key from a random key. We define the advantage of adversary A in $\mathbf{PasSec}^b_{\mathsf{NIKE},par}$ as

$$\mathrm{Adv}^{\mathbf{PasSec}}_{\mathsf{NIKE},par}(\mathsf{A}) := \left|\begin{array}{l}\Pr[\mathbf{PasSec}^{0,\mathsf{A}}_{\mathsf{NIKE},par} \Rightarrow 1]\\ -\Pr[\mathbf{PasSec}^{1,\mathsf{A}}_{\mathsf{NIKE},par} \Rightarrow 1]\end{array}\right|.$$

Definition 4.4 (Passive Security). Let NIKE := (Stp, Gen, SdK) be a non-interactive key exchange. We say that NIKE is $(\epsilon, Q_{\mathsf{H}})$-**PasSec** relative to $par \in \mathsf{Stp}(1^\lambda)$ if for all $\mathtt{ID}_1, \mathtt{ID}_2 \in \mathcal{IDS}$ and for all PPT adversaries A, making at most Q_{H} queries (possibly in superposition) to the random oracle H, we have $\mathrm{Adv}^{\mathbf{PasSec}}_{\mathsf{NIKE},par}(\mathsf{A}) \leq \epsilon(Q_{\mathsf{H}})$.

Game $\mathbf{PasSec}^b_{\mathsf{NIKE},par}$

```
01 $(sk_1, pk_1) \xleftarrow{\$} \mathsf{Gen}(\mathtt{ID}_1)$
02 $(sk_2, pk_2) \xleftarrow{\$} \mathsf{Gen}(\mathtt{ID}_2)$
03 $k_0 := \mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2)$
04 $k_1 \xleftarrow{\$} \mathcal{SKS}$
05 $b' \leftarrow \mathsf{A}^{|\mathsf{H}\rangle}(pk_1, pk_2, k_b)$
06 return $[\![b = b']\!]$
```

Figure 3: Passive security game $\mathbf{PasSec}^b_{\mathsf{NIKE},par}$ for a non-interactive key exchange NIKE defined relative to a random oracle H with adversary A.

Active Security. We formalise the notion of key indistinguishability with *active security* for a non-interactive key exchange NIKE, with respect to system parameters $par \in \mathsf{Stp}(1^\lambda)$ via the game $\mathbf{ActSec}^b_{\mathsf{NIKE},par}$ depicted in Figure 4. Observe that the **ActSec** notion defined here corresponds to *CKS-light* which is polynomially equivalent to *CKS* and *m-CKS-heavy* in the work of [41]. The original *CKS* notion was defined in [28]. Unsurprisingly our definition of active security implies the former notion of passive security. The game starts by selecting a bit b uniformly at random after which the adversary A is given access to four oracles. A's queries may be made adaptively and are arbitrary in number. The RegHonUsr and RegCorUsr oracles let A register honest and corrupted user public keys, respectively. A may make multiple queries to RegCorUsr, in which case only the most recent $(corrupt, \mathsf{ID}, \perp, pk)$ entry is kept. The RevCorQue oracle provides A with a shared key between a pair of registered identities, subject only to the restriction that at least one of the two identities was registered as honest. Depending on the bit b, the TestQue oracle returns either a random key or a shared key between two identities registered as honest. Finally, the adversary outputs a guess bit b' and wins the game if and only if $b = b'$. We define the advantage of adversary A in $\mathbf{ActSec}^{b}_{\mathsf{NIKE},par}$ as

$$\mathsf{Adv}^{\mathbf{ActSec}}_{\mathsf{NIKE},par}(\mathsf{A}) := \left| \begin{array}{l} \Pr[\mathbf{ActSec}^{0,\mathsf{A}}_{\mathsf{NIKE},par} \Rightarrow 1] \\ -\Pr[\mathbf{ActSec}^{1,\mathsf{A}}_{\mathsf{NIKE},par} \Rightarrow 1] \end{array} \right|.$$

Definition 4.5 (Active Security [28]). Let $\mathsf{NIKE} := (\mathsf{Stp}, \mathsf{Gen}, \mathsf{SdK})$ be a non-interactive key exchange. We say that NIKE is $(\epsilon, Q_{\mathsf{H}}, Q_{\mathsf{RHU}}, Q_{\mathsf{RCU}}, Q_{\mathsf{RCQ}}, Q_{\mathsf{TQ}})$-**ActSec** relative to $par \in \mathsf{Stp}(1^\lambda)$ if for all PPT adversaries A making at most; Q_{H} queries (possibly in superposition) to the random oracle H, Q_{RHU} queries to RegHonUsr, Q_{RCU} queries to RegCorUsr, Q_{RCQ} queries to RevCorQue, and Q_{TQ} queries to TestQue, we have $\mathsf{Adv}^{\mathbf{ActSec}}_{\mathsf{NIKE},par}(\mathsf{A}) \leq \epsilon$.

5 CONSTRUCTION

We present our NIKE construction in two steps by introducing a scheme that only satisfies passive security followed by a generic transformation that turns it into a scheme with active security.

5.1 Passive Setting

In this section we present our construction of a non-interactive key exchange with semi-malicious correctness that satisfies key indistinguishability for honestly registered public keys (passive security) in the random-oracle model. The scheme is depicted in Figure 5.

Correctness. In order to achieve better bounds in our proof of security, we show that our scheme satisfies both honest correctness as well as the stronger notion of semi-malicious correctness of Definition 4.2 and Definition 4.3, respectively. Although Theorem 5.1 implies Lemma 1, we will use the latter and state its proof in Appendix B for sake of completeness.

Lemma 1 (Honest Correctness). For all (possibly unbounded) adversaries A the non-interactive key exchange $\mathsf{NIKE} := (\mathsf{Stp}, \mathsf{Gen}, \mathsf{SdK})$ construction depicted in Figure 5 has *honest correctness error*

$$\delta \leq \frac{4\beta^2 d^2 N}{q}$$

as per Definition 4.2.

We show that the scheme satisfies the stronger notion of semi-malicious correctness in the quantum random-oracle model.

Theorem 5.1 (**SM-COR** of NIKE). For all (possibly unbounded) adversaries A of depth D making at most Q_{H} queries (possibly in superposition) to the random oracle H, the non-interactive key exchange $\mathsf{NIKE} := (\mathsf{Stp}, \mathsf{Gen}, \mathsf{SdK})$ construction depicted in Figure 5 has *semi-malicious correctness error*

$$\delta(Q_{\mathsf{H}}, D) \leq 16 \cdot (D+2) \cdot (Q_{\mathsf{H}}+1) \cdot \frac{\beta^2 d^2 N}{q}$$

as per Definition 4.3, where β is a bound on the maximum absolute value of the support of χ.

Proof. We are going to prove that the adversary cannot cause an error in the key-derivation, i.e., a mismatch between the derived keys, even if he is allowed to choose both secret keys from the support of the key generation algorithm. This trivially implies semi-malicious correctness. Let (sk_1, pk_1) and (sk_2, pk_2) be the pairs returned by the adversary. Without loss of generality we can consider $sk_1 = sk_L$ and $pk_2 = pk_R$, i.e., only "one side" of the key. A key mismatch occurs whenever

$$\mathsf{Rec}\left(pk_L^\top sk_R + \boldsymbol{r}\right) \neq \mathsf{Rec}\left(sk_L^\top pk_R + \boldsymbol{r}\right)$$
$$\mathsf{Rec}\left(\left(\vec{\boldsymbol{s}}_L^\top \boldsymbol{A} + \vec{\boldsymbol{e}}_L^\top\right) \vec{\boldsymbol{s}}_R + \boldsymbol{r}\right) \neq \mathsf{Rec}\left(\vec{\boldsymbol{s}}_L^\top\left(\boldsymbol{A}\vec{\boldsymbol{s}}_R + \vec{\boldsymbol{e}}_R\right) + \boldsymbol{r}\right)$$
$$\mathsf{Rec}\left(\underbrace{\vec{\boldsymbol{s}}_L^\top \boldsymbol{A}\vec{\boldsymbol{s}}_R + \boldsymbol{r}}_{\boldsymbol{k}^\star \in \mathcal{R}_q} + \vec{\boldsymbol{e}}_L^\top \vec{\boldsymbol{s}}_R\right) \neq \mathsf{Rec}\left(\underbrace{\vec{\boldsymbol{s}}_L^\top \boldsymbol{A}\vec{\boldsymbol{s}}_R + \boldsymbol{r}}_{\boldsymbol{k}^\star \in \mathcal{R}_q} + \vec{\boldsymbol{s}}_L^\top \vec{\boldsymbol{e}}_R\right),$$

where $\boldsymbol{r}$ is the output of the random oracle on both public keys and $\vec{\boldsymbol{e}}_L$ and $\vec{\boldsymbol{e}}_R$ are sampled from the noise distribution χ^N. By definition of the Rec function, this means that the term $\vec{\boldsymbol{e}}_L^\top \vec{\boldsymbol{s}}_R$ (or, equivalently, the term $\vec{\boldsymbol{s}}_L^\top \vec{\boldsymbol{e}}_R$) is causing a rounding error on one of the coefficients of $\boldsymbol{k}^\star$. We now bound the size of the largest coefficient of $\vec{\boldsymbol{e}}_L^\top \vec{\boldsymbol{s}}_R$ as

$$\begin{aligned} \left\|\vec{\boldsymbol{e}}_L^\top \vec{\boldsymbol{s}}_R\right\|_\infty &= \left\|\sum_{i=1}^{N} \boldsymbol{e}_{L,i} \boldsymbol{s}_{R,i}\right\|_\infty \\ &\leq \sum_{i=1}^{N} \left\|\boldsymbol{e}_{L,i} \boldsymbol{s}_{R,i}\right\|_\infty \\ &\leq \beta^2 dN, \end{aligned}$$

where the first inequality follows from the triangle inequality. The norm of $\vec{\boldsymbol{s}}_L^\top \vec{\boldsymbol{e}}_R$ can be bounded similarly. It follows that, in order for a key-derivation error to occur, at least one coefficient of $\boldsymbol{k}^\star$ must be in the following interval

$$S^\star = \left[\frac{q}{4} \pm \beta^2 dN\right] \cup \left[\frac{3q}{4} \pm \beta^2 dN\right].$$

Next we define a function F that, on input two public keys and two identities samples a uniform $\boldsymbol{r}$, it returns 1 if a key mismatch occurs, i.e.,

$$\mathsf{Rec}\left(pk_L^\top sk_R + \boldsymbol{r}\right) \neq \mathsf{Rec}\left(sk_L^\top pk_R + \boldsymbol{r}\right)$$

and 0 otherwise. The function checks this by (inefficiently) recovering the secret keys and comparing the results of the Rec functions (see equation above). Note that, since $\boldsymbol{A}$ is invertible, the secret key is uniquely determined by the public key, and therefore

```
Game $\mathbf{ActSec}^b_{\mathsf{NIKE},par}$
$\mathcal{D} := \bot$
$\mathcal{K} := \bot$
$b' \leftarrow \mathsf{A}^{|\mathsf{H}\rangle, \mathtt{RegHonUsr}(\cdot), \mathtt{RegCorUsr}(\cdot,\cdot), \mathtt{RevCorQue}(\cdot,\cdot), \mathtt{TestQue}(\cdot,\cdot)}$
return $[\![b = b']\!]$

Oracle $\mathtt{RegHonUsr}(\mathtt{ID} \in \mathcal{IDS})$          / Twice in CKS-light
if $(corrupt, \mathtt{ID}, \bot, \cdot) \in \mathcal{D}$ :
    return $\bot$
$(sk, pk) \xleftarrow{\$} \mathsf{Gen}(\mathtt{ID})$
$\mathcal{D} \cup \{(honest, \mathtt{ID}, sk, pk)\}$
return $pk$

Oracle $\mathtt{RegCorUsr}(\mathtt{ID} \in \mathcal{IDS}, pk)$
if $(corrupt, \mathtt{ID}, \bot, \cdot) \in \mathcal{D}$ :
    $(corrupt, \mathtt{ID}, \bot, \cdot) := (corrupt, \mathtt{ID}, \bot, pk)$
else :
    $\mathcal{D} \cup \{(corrupt, \mathtt{ID}, \bot, pk)\}$

Oracle $\mathtt{RevCorQue}(\mathtt{ID}_1, \mathtt{ID}_2)$
if $(honest, \mathtt{ID}_1, \cdot, \cdot) \in \mathcal{D} \wedge (corrupt, \mathtt{ID}_2, \cdot, \cdot) \in \mathcal{D}$ :
    return $\mathsf{SdK}(\mathtt{ID}_2, pk_2, \mathtt{ID}_1, sk_1)$
elseif $(corrupt, \mathtt{ID}_1, \cdot, \cdot) \in \mathcal{D} \wedge (honest, \mathtt{ID}_2, \cdot, \cdot) \in \mathcal{D}$ :
    return $\mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2)$

Oracle $\mathtt{TestQue}(\mathtt{ID}_1, \mathtt{ID}_2)$          / Once in CKS-light
if $\mathtt{ID}_1 = \mathtt{ID}_2$ :
    return $\bot$
if $(honest, \mathtt{ID}_1, \cdot, \cdot) \in \mathcal{D} \wedge (honest, \mathtt{ID}_2, \cdot, \cdot) \in \mathcal{D}$ :
    if $b = 0$ :
        $k := \mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2)$
    if $b = 1$ :
        if $(\mathtt{ID}_1, \mathtt{ID}_2, k) \in \mathcal{K} \vee (\mathtt{ID}_2, \mathtt{ID}_1, k) \in \mathcal{K}$ :
            return $k$
        $k \xleftarrow{\$} \mathcal{SKS}$
        $\mathcal{K} \cup \{(\mathtt{ID}_1, \mathtt{ID}_2, k)\}$
    return $k$
return $\bot$
```

Figure 4: Game defining $\mathbf{ActSec}^b_{\mathsf{NIKE},par}$ for a non-interactive key exchange NIKE with adversary A.

```
$\mathsf{Stp}(1^\lambda)$
$\mathcal{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$
$\boldsymbol{A} \xleftarrow{\$} \mathrm{GL}(N, \mathcal{R}_q)$
$par := (q, d, \mathcal{R}_q, N, \boldsymbol{A})$
return $par$

$\mathsf{Gen}(\mathtt{ID})$
$\vec{s}_L, \vec{s}_R \leftarrow \mathsf{Cbd}(\cdot)$     / Samples $\vec{s} \in \mathcal{R}_q^N$ from $\chi^N$
$\vec{e}_L, \vec{e}_R \leftarrow \mathsf{Cbd}(\cdot)$     / Samples $\vec{e} \in \mathcal{R}_q^N$ from $\chi^N$
$sk_L := \vec{s}_L^\top \in \mathcal{R}_q^{1\times N}$
$sk_R := \vec{s}_R \in \mathcal{R}_q^N$
$pk_L := \vec{s}_L^\top \boldsymbol{A} + \vec{e}_L^\top \in \mathcal{R}_q^{1\times N}$
$pk_R := \boldsymbol{A}\vec{s}_R + \vec{e}_R \in \mathcal{R}_q^N$
return $(sk_{\mathtt{ID}} := (sk_L, sk_R), pk_{\mathtt{ID}} := (pk_L, pk_R))$

$\mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2)$
if $\mathtt{ID}_1 \leq \mathtt{ID}_2$ :
    $\boldsymbol{r} := \mathsf{H}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, pk_2) \in \mathcal{R}_q$
    parse $pk_1 \rightarrow (pk_L, \bot) =: \vec{u}_L^\top \in \mathcal{R}_q^{1\times N}$
    parse $sk_2 \rightarrow (\bot, sk_R) =: \vec{s}_R \in \mathcal{R}_q^N$
    $\boldsymbol{k}' := \vec{u}_L^\top \vec{s}_R + \boldsymbol{r} \in \mathcal{R}_q$
else :
    $\boldsymbol{r} := \mathsf{H}(\mathtt{ID}_2, pk_2, \mathtt{ID}_1, pk_1) \in \mathcal{R}_q$
    parse $pk_1 \rightarrow (\bot, pk_R) =: \vec{u}_R \in \mathcal{R}_q^N$
    parse $sk_2 \rightarrow (sk_L, \bot) =: \vec{s}_R^\top \in \mathcal{R}_q^{1\times N}$
    $\boldsymbol{k}' := \vec{s}_R^\top \vec{u}_R + \boldsymbol{r} \in \mathcal{R}_q$
$k := \mathsf{Rec}(\boldsymbol{k}') \in \{0,1\}^d$
return $k$

$\mathsf{Rec}(\boldsymbol{k})$
for $i \in \{0, \ldots, d-1\}$ :
    $\mathrm{k_i} := \mathsf{Rnd}(k_i) \in \{0,1\}$
return $k \in \{0,1\}^d$

$\mathsf{Rnd}(k_i)$
if $\frac{q}{4} \leq k_i \leq \frac{3q}{4}$ :
    return 1
else :
    return 0

$\mathsf{Cbd}(\cdot)$
for $i \in \{1, \ldots, N\}$ :
    for $j \in \{0, \ldots, d-1\}$ :
        $a, b \xleftarrow{\$} \{0,1\}$
        $f_j := a - b$
    $f_i := \sum_{j=0}^{d-1} f_j X^j$
return $\vec{f} := (f_1, \ldots, f_N)$
```

Figure 5: Construction of passively secure non-interactive key exchange $\mathsf{NIKE} := (\mathsf{Stp}, \mathsf{Gen}, \mathsf{SdK})$ with functions $\mathsf{Rec} : \mathcal{R}_q \to \{0,1\}^d$, $\mathsf{Rnd} : \mathbb{Z}_q \to \{0,1\}$ and $\mathsf{Cbd} : \emptyset \to \mathcal{R}_q^N$, and random oracle $\mathsf{H} : \mathcal{IDS} \times \left(\mathcal{R}_q^{1\times N} \times \mathcal{R}_q^N\right) \times \mathcal{IDS} \times \left(\mathcal{R}_q^{1\times N} \times \mathcal{R}_q^N\right) \to \mathcal{R}_q$. Here $\mathrm{GL}(N, \mathcal{R}_q)$ denotes the set of invertible matrices over $\mathcal{R}_q$.

this (inefficient) function is well defined on all inputs. Furthermore, note that the element

$$\boldsymbol{k}^\star = sk_L^\top \boldsymbol{A} sk_R + \boldsymbol{r}$$

is uniformly distributed in $\mathcal{R}_q$, since $\boldsymbol{r} \xleftarrow{\$} \mathcal{R}_q$. It follows that for any given input x:

$$\Pr[F(x) = 1] \leq \frac{4\beta^2 d^2 N}{q}.$$

Finally, observe that by definition a key mismatch happens if and only if the function F outputs 1 and consequently the adversary is able to find such accepting input. By Lemma 3.1, this happens with probability at most $16 \cdot (D+2) \cdot (Q_{\mathsf{H}}+1) \cdot \beta^2 d^2 N/q$ for an adversary of depth D, making at most Q_{H} quantum query to the random oracle. ∎

On the Need for Random Oracles. An astute reader may wonder whether the usage of the random oracle is needed at all to prove the above notion of correctness, since there does not appear to be an immediate attack even if we omit the random oracle completely from the scheme. It is plausible to conjecture that semi-malicious correctness holds even without the random oracle. Informally, semi-malicious correctness boils down to showing that, for a given public key $pk \in \mathcal{R}_q^N$, it is hard to find an $\boldsymbol{s} \in \mathcal{R}_q^N$ such that for no coefficient of the product $\boldsymbol{s}^\top pk$ lies in the interval $S^\star$. Thus, the a bound in these settings would require one to estimate the hardness of this version of the (inhomogenous) 1-dimensional short-integer-solution (SIS) problem. By relying on the random-oracle heuristic, we are able to bypass this problem and obtain a construction in the QROM that is: (i) unconditionally correct in *any ring* and (ii) whose security is based on the well-established M-LWE problem. We leave the precise study of the hardness of this 1-dimensional variant of the SIS problem as ground for future work.

Passive Security. Assuming the hardness of M-LWE, Definition 3.2, we show that the scheme satisfies *passive security*, Definition 4.4, in the QROM.

Theorem 5.2 (Passive Security). For any PPT adversary A against NIKE := (Stp, Gen, SdK), depicted in Figure 7, making at most Q_{H} queries (possibly in superposition) to H, there exist PPT adversaries $\mathsf{B}_1, \mathsf{B}_2$ such that

$$\mathsf{Adv}^{\mathbf{PasSec}}_{\mathsf{NIKE},par}(\mathsf{A}) \leq 6 \cdot \mathsf{Adv}^{\mathbf{M\text{-}LWE}}_{q,N,N,\chi}(\mathsf{B}_1) + 2 \cdot \mathsf{Adv}^{\mathbf{M\text{-}LWE}}_{q,N,N+1,\chi}(\mathsf{B}_2) + \frac{4\beta d}{q}.$$

Proof of Theorem 5.2. Let A be an adversary against NIKE in the **PasSec** game. Consider the sequence of games in Figure 6.

Game G_0. This is the original $\mathbf{PasSec}^b_{\mathsf{NIKE},par}$ game so by definition

$$\mathsf{Adv}^{\mathbf{PasSec}}_{\mathsf{NIKE},par}(\mathsf{A}) \leq \left|\Pr\left[\mathsf{G}_0^{\mathsf{A}} \Rightarrow 1\right] - \frac{1}{2}\right|.$$

Game G_1. In this game the half of pk_1 that is used in the key-derivation is replaced with a uniform key. Without loss of generality we can consider either half of the key. Indistinguishability follows from a reduction against the M-LWE problem, conditioned on the matrix $\boldsymbol{A}$ being invertible. Since this happens with probability at least $1/2$, we have that

$$\left|\Pr\left[\mathsf{G}_0^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_1^{\mathsf{A}} \Rightarrow 1\right]\right| \leq 2 \cdot \mathsf{Adv}^{\mathbf{M\text{-}LWE}}_{q,N,N,\chi}(\mathsf{B}_1).$$

```
Game $\mathbf{PasSec}^b_{\mathsf{NIKE},par}$
$(sk_1, pk_1) \xleftarrow{\$} \mathsf{Gen}(\mathtt{ID}_1)$
$pk_1 \xleftarrow{\$} \mathcal{R}_q^{1\times N} \times \mathcal{R}_q^N$                    / $\mathsf{G}_1$
$(sk_1, pk_1) \xleftarrow{\$} \mathsf{Gen}(\mathtt{ID}_1)$                   / $\mathsf{G}_4$
$(sk_2, pk_2) \xleftarrow{\$} \mathsf{Gen}(\mathtt{ID}_2)$
$pk_2 \xleftarrow{\$} \mathcal{R}_q^{1\times N} \times \mathcal{R}_q^N$                    / $\mathsf{G}_3$
$(sk_2, pk_2) \xleftarrow{\$} \mathsf{Gen}(\mathtt{ID}_2)$                   / $\mathsf{G}_4$
if $\mathtt{ID}_1 \leq \mathtt{ID}_2$ :
    $\boldsymbol{r} := \mathsf{H}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, pk_2) \in \mathcal{R}_q$
    parse $pk_1 \longrightarrow (pk_L, \perp) =: \vec{\boldsymbol{u}}_L^\top \in \mathcal{R}_q^{1\times N}$
    parse $sk_2 \longrightarrow (\perp, sk_R) =: \vec{\boldsymbol{s}}_R \in \mathcal{R}_q^N$
    $\boldsymbol{k}' := \vec{\boldsymbol{u}}_L^\top \vec{\boldsymbol{s}}_R + \boldsymbol{r} \in \mathcal{R}_q$
else :
    $\boldsymbol{r} := \mathsf{H}(\mathtt{ID}_2, pk_2, \mathtt{ID}_1, pk_1) \in \mathcal{R}_q$
    parse $pk_1 \longrightarrow (\perp, pk_R) =: \vec{\boldsymbol{u}}_R \in \mathcal{R}_q^N$
    parse $sk_2 \longrightarrow (sk_L, \perp) =: \vec{\boldsymbol{s}}_R^\top \in \mathcal{R}_q^{1\times N}$
    $\boldsymbol{k}' := \vec{\boldsymbol{s}}_R^\top \vec{\boldsymbol{u}}_R + \boldsymbol{r} \in \mathcal{R}_q$
$k_0 := \mathsf{Rec}(\boldsymbol{k}') \in \{0,1\}^d$                      / $\mathsf{G}_0$
$\boldsymbol{e} \xleftarrow{\$} \chi$                                       / $\mathsf{G}_2$
$k_0 := \mathsf{Rec}(\boldsymbol{k}' + \boldsymbol{e}) \in \{0,1\}^d$                  / $\mathsf{G}_2$
$\boldsymbol{u} \xleftarrow{\$} \mathcal{R}_q$                                      / $\mathsf{G}_3$
$k_0 := \mathsf{Rec}(\boldsymbol{u}) \in \{0,1\}^d$                       / $\mathsf{G}_3$
$k_1 \xleftarrow{\$} \mathcal{SKS}$
$b' \leftarrow \mathsf{A}^{|\mathsf{H}\rangle}(pk_1, pk_2, k_b)$
return $[\![b = b']\!]$
```

Figure 6: Games G_0, G_1, G_2, G_3, G_4 for the proof of PasSec of NIKE in Figure 5.

Game G_2. In this hybrid we modify the way we compute the shared key. Consider $\boldsymbol{k}'$ as computed in the SdK algorithm, we define the shared key as $\mathsf{Rec}(\boldsymbol{k}' + \boldsymbol{e})$ where $\boldsymbol{e} \xleftarrow{\$} \chi$ is a freshly sampled ring element from the noise distribution. Note that the adversary can only detect a change in this hybrid if

$$\mathsf{Rec}(\boldsymbol{k}' + \boldsymbol{e}) \neq \mathsf{Rec}(\boldsymbol{k}).$$

Since $\boldsymbol{k}'$ is uniformly sampled from $\mathcal{R}_q$, the probability that any coefficient is rounded to a different term is at most $4\beta d/q$, which is also an upper bound on the distinguishing advantage of the adversary. Thus we get

$$\left|\Pr\left[\mathsf{G}_1^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_2^{\mathsf{A}} \Rightarrow 1\right]\right| \leq \frac{4\beta d}{q}.$$

Game G_3. In this game the half of pk_2 used in the key-derivation is replaced with a uniform key, along with $\boldsymbol{k}' + \boldsymbol{e}$ that is replaced with a uniform ring element $\boldsymbol{u}$. By another invocation of the M-LWE assumption, again conditioning on $\boldsymbol{A}$ being invertible, we have that

$$\left|\Pr\left[\mathsf{G}_2^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_3^{\mathsf{A}} \Rightarrow 1\right]\right| \leq 2 \cdot \mathsf{Adv}^{\mathbf{M\text{-}LWE}}_{q,N,N+1,\chi}(\mathsf{B}_2).$$

Game G_4. In this game we revert the changes made to pk_1 and pk_2, and appealing again to Definition 3.2 we get

$$\left|\Pr\left[G_3^A \Rightarrow 1\right] - \Pr\left[G_4^A \Rightarrow 1\right]\right| \leq 4 \cdot \mathsf{Adv}^{\textbf{M-LWE}}_{q,N,N,\chi}(\mathsf{B}_1).$$

Observe that k_0 and k_1 are identically distributed and the adversary can only guess b'. Hence,

$$\Pr\left[G_4^A \Rightarrow 1\right] = \frac{1}{2}.$$

Collecting all probabilities yields the bound stated in Theorem 5.2. ■

5.2 Active Setting

Here we show how a non-interactive key exchange with passive security can be generically transformed to one with active security. The transformation, depicted in Figure 7, requires a simulation-sound NIZK with a straight-line extractor. The proof is deferred to Appendix B.

Theorem 5.3 (**PasSec** and **SM-COR** of $\mathsf{NIKE}' \overset{\text{QROM}}{\underset{\text{ZKPoK}}{\Rightarrow}}$ **ActSec** of NIKE). Let $\mathsf{H} : \{0,1\}^* \rightarrow \mathcal{R}_q$ be a random oracle and $\mathsf{NIKE}' := (\mathsf{Stp}', \mathsf{Gen}', \mathsf{SdK}')$ a passively secure non-interactive key exchange with semi-malicious correctness defined relative to $par' \in \mathsf{Stp}'(1^\lambda)$. Further, let $\mathsf{ZKPoK} := (\mathsf{ZK.Prv}, \mathsf{ZK.Ver})$ be a simulation-sound online extractabile zero-knowledge proof of knowledge for the NP relation $R = (pk_{\mathrm{ID}}, sk_{\mathrm{ID}})$. Then, for any **ActSec** adversary A against $\mathsf{NIKE} := (\mathsf{Stp}, \mathsf{Gen}, \mathsf{SdK})$, depicted in Figure 7, there exist PPT adversaries $\mathsf{B}_1, \mathsf{B}_2, \mathsf{B}_{3.i}, \mathsf{B}'_{3.i}, \mathsf{B}_4$ such that

$$\begin{aligned}\mathsf{Adv}^{\textbf{ActSec}}_{\mathsf{NIKE},par}(\mathsf{A}) \leq & Q_{\mathsf{RCU}} \cdot \mathsf{Adv}^{\textbf{SSND}}_{\mathsf{ZKPoK}}(\mathsf{B}_1) + 2 \cdot \mathsf{Adv}^{\textbf{SM-COR}}_{\mathsf{NIKE}',par'}(\mathsf{B}_2)\\ &+ Q_{\mathsf{TQ}} \cdot Q^2_{\mathsf{RHU}} \cdot \mathsf{Adv}^{\textbf{PasSec}}_{\mathsf{NIKE}',par'}(\mathsf{B}_{3.i})\\ &+ 2 \cdot Q_{\mathsf{TQ}} \cdot \mathsf{Adv}^{\textbf{SM-COR}}_{\mathsf{NIKE}',par'}(\mathsf{B}'_{3.i})\\ &+ 2 \cdot Q_{\mathsf{RHU}} \cdot \mathsf{Adv}^{\textbf{ZK}}_{\mathsf{ZKPoK}}(\mathsf{B}_4),\end{aligned}$$

where Q_{RCU} and Q_{RHU}, are the number of queries made by A to `RegCorUsr` and `RegHonUsr`, respectively, and Q_{TQ} denotes the number of queries made by B_i to `TestQue` for $i \in \{0, \ldots, Q_{\mathsf{TQ}} - 1\}$.

5.3 Practical considerations

Halving the Key Size. Observe that the "left" and "right" components pk_L and pk_R of the public key of the NIKE as specified in Figure 5 are necessary because we work in the non-commutative M-LWE setting. An easy way to halve the size of the public key would be to set $N = 1$, i.e., to work in the R-LWE setting; this also eliminates the need for the case distinction in SdK. We argue that for essentially all relevant applications of a NIKE, we can halve the public-key size even *without* moving to the R-LWE setting. All that is required is that protocol participants (and their associated NIKE keys) have different *roles*, typically called initiator and responder or client and server, and that these roles are clear from protocol context. This is certainly the case for the application examples sketched in Section 1.1: The OPTLS handshake, like the TLS handshake, clearly distinguishes the roles of client and server, so does the handshake in (post-quantum) WireGuard. Also in X3DH the critical static-semistatic key exchange has clear roles that can be used to distinguish between the "left" and "right" participant instead of transmitting both halves of the key and using comparison of IDs. Note that this setting of a NIKE using keys with different roles is very similar to the ℓ_A and ℓ_B keys of SIDH [50, § 3.2], when it was still considered as a replacement for DH, i.e., before it was shown to not be actively secure in [44] and completely broken in [29].

Based on these considerations, we stick to the M-LWE setting for the construction of Swoosh; in our performance evaluation in Section 6 we report the size of only one public-key component.

Security of the NIZK. We highlight that our proof of active security, Theorem 5.3, requires the strong property of simulation-sound online-extractability. Although constructions satisfying such a strong notion exist [77], they tend to be less efficient than alternatives satisfying weaker notions of security. For instance, a proof of knowledge of an M-LWE secret satisfying simulation soundness, but without *online*-extractability, using state of the art techniques [59] and appropriate parameters is around 70 KB in size.

It appears likely that the need for the stronger notion is an artefact of the proof, and we conjecture that our construction remains secure even if we use NIZKs that are simulation-sound and extractable, although not online-extractable (such as the protocol in [59]). We are not the first to make this additional assumption, in favour of a more efficient scheme and similar heuristics have already appeared in the literature, e.g., in [31]. While we cannot exclude that contrived examples of NIZKs could make our compiler fail, we believe that all natural candidates of NIZKs would lead to secure schemes.

Tangentially, we also mention that for some applications, the performance of the NIZK does not affect the efficiency of the shared-key computation, since it can be verified once and for all for a given public key: In any scenario where the public keys are distributed by some PKI, the NIZK proof can be simply verified by the PKI upon the registration of the key, and then immediately discarded. The users would then trust the PKI to have verified the NIZK on their behalf. Note that this does not introduce any extra trust assumption, since the PKI is anyway trusted to provide the correct public key. In these scenarios, the efficiency of the NIZK only marginally impacts the overall system performance, and thus justifies ignoring the costs of the NIZK for shared-key computation.

5.4 Parameter selection

Selecting parameters for the scheme influences several aspects, most notably the correctness error and the hardness of M-LWE. In order to evaluate the security of our scheme we use the *Lattice-Estimator* [2, 4, 68], to estimate the memory and CPU operations required to perform various lattice attacks, including dual attacks, uSVP, the Coded-BKW attack, and solving using Gröbner bases with the Arora-Ge attack. The estimator has been used to estimate the concrete security for all LWE and NTRU based candidates of the NIST competition [3], and is regularly updated to include the latest developments in lattice cryptanalysis[2]. However, we also take into account practical

[2]An up-to-date list of implemented works can be found https://lattice-estimator.readthedocs.io/en/latest/references.html.

```
Stp(1^λ)
01 $par \xleftarrow{\$} \mathsf{Stp}'(1^\lambda)$
02 return $par$

Gen(ID)
03 $(sk'_{\mathtt{ID}}, pk'_{\mathtt{ID}}) \xleftarrow{\$} \mathsf{Gen}'(\mathtt{ID})$
04 $\pi \xleftarrow{\$} \mathsf{ZK.Prv}(pk'_{\mathtt{ID}}, sk'_{\mathtt{ID}})$
05 $sk_{\mathtt{ID}} \coloneqq sk'_{\mathtt{ID}}$
06 $pk_{\mathtt{ID}} \coloneqq (pk'_{\mathtt{ID}}, \pi)$
07 return $(sk_{\mathtt{ID}}, pk_{\mathtt{ID}})$

SdK(ID_1, pk_1, ID_2, sk_2)
08 parse $pk_1 \rightarrow (pk'_1, \pi)$
09 if $\mathsf{ZK.Ver}(pk'_1, \pi) = 0$ : return $\perp$
10 $k' \coloneqq \mathsf{SdK}'(\mathtt{ID}_1, pk'_1, \mathtt{ID}_2, sk_2)$
11 return $k'$
```

Figure 7: Compiler for transforming a passively secure non-interactive key exchange $\mathsf{NIKE}' \coloneqq (\mathsf{Stp}', \mathsf{Gen}', \mathsf{SdK}')$ with semi-malicious correctness into an actively secure non-interactive key exchange $\mathsf{NIKE} \coloneqq (\mathsf{Stp}, \mathsf{Gen}, \mathsf{SdK})$.

considerations for the implementation when selecting our parameters, such as the use of ternary secrets and noise sampled from a centred binomial distribution. For our scheme with parameters $n = 8192$, $q = 2^{214} - 255$ and $\mathcal{X}$ a ternary distribution, we estimate the hardness of the M-LWE problem underlying Swoosh at 120 bits[3].

The other way to attack Swoosh is, for an active attacker, to try to produce failures. We consider a quantum attacker with a bounded query depth of $D = 2^{64}$ (i.e., what NIST considers to be "the approximate number of gates that current classical computing architectures can perform serially in a decade" [63, Sec. 4.A]) and a bound on the number of queries of 2^{120} (i.e., matching the hardness of the underlying lattice problem). Applying Theorem 5.1 yields a success probability (correctness error), after this amount of computation, of

$$16 \cdot \left(2^{64} + 2\right) \cdot \left(2^{120} + 1\right) \cdot \frac{256^2 \cdot 32}{2^{214}} < \frac{1}{2^4} = \delta(Q_{\mathsf{H}}, D),$$

i.e., considerably smaller than 1/2. Note that this analysis is conservative as it ignores the circuit depth for the Grover oracle that an attacker would need to implement.

Generating an Invertible Matrix. In order to justify our conservative estimate that at least 50% of all matrices in $\mathcal{R}_q^{N\times N}$ are invertible, we used Sage to generate 2000 random matrices and checked if they are invertible. They were all invertible. We additionally verified that the concrete matrix used by our implementation (see Section 6) is invertible.

Parameter	Description	Value
β	upper bound on $\lVert\vec{s}\rVert_\infty = \lVert\vec{e}\rVert_\infty$	1
q	prime modulus	$2^{214} - 255$
d	dim of $\mathcal{R}_q \coloneqq \mathbb{Z}_q[X]/(X^d + 1)$	256
l	# factors $X^d + 1$ splits into mod q	128
N	height of the $\boldsymbol{A}$ matrix	32
n	lattice dimension	8192
χ	noise distribution	$p(-1) = 25\%$ $p(0) = 50\%$ $p(1) = 25\%$

Table 1: Parameter selection for non-interactive key exchange NIKE.

[3]These numbers can be reproduced with the estimator — the version used in this work is at commit 96875622c6b0e6f98a91ddeecaaa17b66dbc5a87.

6 IMPLEMENTATION AND PERFORMANCE EVALUATION

In order to demonstrate the practicality of Swoosh in terms of performance, we implement the core part of the scheme, Passive-Swoosh, present benchmarks of this implementation, and compare to other KEMs and (pre- and post-quantum) NIKEs. We caution the reader that all implementation details and numbers we present in this section are for Passive-Swoosh only. To obtain a full picture of the performance of Swoosh, the implementation will need to be augmented with a future implementation of the NIZKP from [59]. As outlined in Section 5.3, the performance impact of adding the NIZKP in terms of both size and computational effort depends on the concrete application scenario and may be negligible if key-generation performance is not critical and if NIZKP verification can be outsourced to the PKI.

6.1 Implementation

As a NIKE, Swoosh is composed of two major functions, the key generation procedure and the shared-key computation, the performance of which dictates the practicality of Swoosh.

In the case of the key generation, the matrix $\boldsymbol{A}$ is fixed and assumed to be in the NTT domain, so performance is dictated by the sampling of the secret and error vectors, as well as the computation of the public key which involves two NTT transformations, and a matrix multiplication followed by a polynomial addition. As for the shared key computation, its performance is mainly dictated by the random offset computation, which requires the use of cSHAKE [51] and the polynomial base multiplication required to calculate $\boldsymbol{k}'$ (see Fig. 5). Similar to other schemes, the shared-key derivation also performs rounding of the shared key, however its execution time is negligible. At a high level, the architecture of our implementation is divided into two distinct parts: low-level field arithmetic over $\mathbb{F}_q$ that is implemented using the Jasmin language [6, 7], and polynomial arithmetic in $\mathcal{R}_q$ as well as the scheme itself, both of which are implemented in Rust.

The structure largely mimics the abstract specification in Figure 5. The main difference is that, like other lattice-based schemes [5, 72], we encode and transmit public keys in NTT domain. This massively reduces the number of cycles required for shared-key computation. In addition, as discussed in Section 5.3, we assume that the role of each party is well defined and thus only compute one half of the key. We implement this by passing a Boolean flag as an argument to

Scheme (variant)	Assumption	Non-interactive	Post-quantum	Sizes (in bytes)	
				Ciphertext	Public Key
CRYSTALS-Kyber [72] (`Kyber-512`)	M-LWE	✗	✓	768	800
Classic McEliece [1] (`mceliece348864`)	Binary Goppa Codes	✗	✓	96	261120
X25519 [14]	DLOG	✓	✗	—	32
CTIDH [11] (`CTIDH-1024`)	Supersingular Isogenies	✓	✓	—	128
Passive-Swoosh (this work)	M-LWE	✓	✓	—	221184

Table 2: Public-key sizes for select NIKEs and public-key and ciphertext sizes of select post-quantum KEMs

key generation and shared-key derivation to indicate which party is calling the respective function. Finally, we implement the noise sampling in a slightly different way than one might expect; we will discuss this later in this section.

Zooming in on the low-level field arithmetic, the operations on integers modulo $2^{214} - 255$ require multiple-precision integers since native scalar registers (64 bits in AMD64) are not large enough to store a single field element. This arithmetic is implemented through libjbn[4], a Jasmin library that exposes big-integer arithmetic.

Polynomial Arithmetic. On top of this layer, operations in polynomial rings are implemented using Rust, in addition to other functions such as reconciliation, matrix and noise generation. Similar to other lattice-based schemes, one of the more critical (and easier) operations to optimise (from a performance perspective) is polynomial multiplication. The naive algorithm for multiplying two polynomials in $\mathcal{R}_q$, sometimes called Schoolbook multiplication, involves multiplying all pairs of coefficients, calculating their sum and reducing modulo $X^d + 1$. However, the complexity of this approach is quadratic in the number of coefficients and thus quite costly.

The *Number Theoretic Transform* (NTT) provides a more efficient approach for polynomial multiplication with quasi-logarithmic time complexity $O(d \log(d))$ instead of $O(d^2)$. For a detailed discussion on the NTT refer to [74].

As is the case for other implementations [5, 72], we implement an in-place NTT which requires bit-reversal operations in the forward and inverse transforms but uses less memory. Another optimisation is to make the NTT a part of our scheme, which means the matrix $\boldsymbol{A}$ is sampled in the NTT domain, and the secret and public keys are stored in the NTT domain. This results in the NTT only being used three times, once for the shared key derivation and twice in the key generation to convert the secret and error vectors, which are sampled in the normal domain to the NTT domain before computing the public key. A common trick to speed-up the NTT transformation when using Montgomery reduction [61], as is the case for libjbn, is the pre-computed constants in Montgomery form $\zeta \cdot R \pmod q$.

Noise sampling and matrix generation. Both the matrix generation and noise sampling procedures use a seed, either set as a system parameter for $\boldsymbol{A}$ or as a secret input to a PRG in the case of $\vec{s}$ and $\vec{e}$, to produce a stream bytes from which the distributions are sampled. In the case of matrix generation this is achieved via rejection sampling on the stream of bytes produced by an extendable output function (XOF). The noise sampling procedure, used for generating the secret key and the error vector, samples these vectors from a centred binomial distribution using the output of a PRF with a random seed. As with other schemes where multiplication is optimised using the NTT, the choice of (symmetric) primitive that underlies these functions tends to be a deciding factor for the performance. We chose cSHAKE [51] based on Keccak [38] as the underlying primitive for the XOF and AES256-CTR for the PRF used in noise sampling.

Similar to the NewHope scheme [5], for efficiency reasons the secret and error vectors are sampled from a centred binomial distribution rather than a discrete Gaussian distribution. Using ternary noise means that each coefficient can be generated from only 2 bits and thus, the generation of a polynomial in $\mathcal{R}_q$ only requires $(32 \cdot 256 \cdot 2)/8 = 2048$ (pseudo-random) bytes. Intuitively, our CBD definition in Figure 5 when a and b are sourced from a PRG, maps 00_b and 11_b to $0 \mod q$ with 50% probability, 10_b to $1 \mod q$ and 01_b to $-1 \mod q$ with 25% probability each. Our implementation differs from the specification by applying signed reduction modulo 3 to each two bit block and converting it to a congruent value in $\mathbb{F}_q$, as opposed to using big integer field arithmetic to map bits a and b to an element in $\mathbb{F}_q$. Although this approach produces a different mapping (11_b to $-1 \mod q$, 00_b and 10_b to $0 \mod q$ and 01_b to $1 \mod q$), the distribution of the outputs is identical. Due to the size of our field elements, this approach results in a considerable speed up in the noise sampling.

The random offset used in our scheme is generated by performing rejection sampling on the output of cSHAKE-256 [51].

6.2 Performance Evaluation

In this section we evaluate the performance of our scheme and compare it to others. We also provide a comparison of key sizes and the properties of each scheme such as post-quantum security, and whether they are non-interactive.

The benchmark results for Passive-Swoosh were obtained on an Intel Core i7-6500U (Skylake) running on a single core with Hyper-threading and TurboBoost disabled. The Rust compiler version used for the benchmarks was 1.62.1[5] and the Jasmin compiler version was 2022.09.0. We report the median cycle counds of 10000 runs. In Table 3 we list the results and compare to the cycle counts of CTIDH-1024 as reported in [11, Sec. 8] and of lib25519 [62], on Intel Skylake CPUs.

As expected, the pre-quantum X25519 [14] scheme is orders of magnitude faster than Passive-Swoosh for key generation. However, in many applications of NIKEs, keys are re-used many

[4]See https://github.com/formosa-crypto/libjbn.

[5]The following build configuration options/values were used: `opt-level=3` and `target-cpu="native"`.

times and what is more critical is the performance of shared-key computation. Here the gap to pre-quantum X25519 is considerably smaller and Passive-Swoosh outperforms the only real post-quantum competitor CTIDH by a factor of 48.

Operation	X25519	CTIDH-1024	Passive-Swoosh
NTT	—	—	217 430
NTT^{-1}	—	—	262 992
Noise generation	—	—	89 776
Key generation	28 187	469 520 000	146 920 890
Shared key	87 942	511 190 000	10 612 666

Table 3: Cycle counts on Intel Skylake.

However, as shown in Table 2, CTIDH, Kyber, and X25519 have a public-key size several orders of magnitude smaller than Passive-Swoosh. In this aspect, only Classic McEliece has a public key size comparable to that of Passive-Swoosh, even when taking into account the expected size of the proof of knowledge (see Section 5.3).

7 CONCLUSIONS

In this work, we constructed a NIKE based on the M-LWE problem, with a proof in the QROM. Our scheme is based on the standard blueprint, but with an additional twist to guarantee provable security for arbitrary rings. Our optimised implementation shows that our scheme offers reasonable computational performance and key sizes that should be acceptable for most applications. We view our work as the first evidence contradicting the folklore belief that lattice-based NIKE is too inefficient to be used in practice. As future work, we plan an implementation of the full Swoosh scheme, i.e., including the NIZK proof. We also plan to explore applications of our scheme to more complex protocols and to formally verify the correctness of (parts of) our implementation.

REFERENCES

[1] Martin R. Albrecht, Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, Tanja Lange, Varun Maram, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Kenneth G. Paterson, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Cen Jung Tjhai, Martin Tomlinson, and Wen Wang. 2022. *Classic McEliece.* Technical Report. National Institute of Standards and Technology. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-4-submissions.

[2] Martin R. Albrecht, Benjamin R. Curtis, Amit Deo, Alex Davidson, Rachel Player, Eamonn W. Postlethwaite, Fernando Virdia, and Thomas Wunderer. 2018. Estimate All the LWE, NTRU Schemes!. In *SCN 18: 11th International Conference on Security in Communication Networks (Lecture Notes in Computer Science, Vol. 11035)*, Dario Catalano and Roberto De Prisco (Eds.). Springer, Heidelberg, Germany, Amalfi, Italy, 351–367. https://doi.org/10.1007/978-3-319-98113-0_19

[3] Martin R. Albrecht, Benjamin R. Curtis, Amit Deo, Alex Davidson, Rachel Player, Eamonn W. Postlethwaite, Fernando Virdia, and Thomas Wunderer. 2018. Estimate all the LWE, NTRU schemes! Cryptology ePrint Archive, Report 2018/331. https://eprint.iacr.org/2018/331.

[4] Martin R. Albrecht, Rachel Player, and Sam Scott. 2015. On The Concrete Hardness Of Learning With Errors. Cryptology ePrint Archive, Report 2015/046. https://eprint.iacr.org/2015/046.

[5] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. 2016. Post-quantum Key Exchange - A New Hope. In *USENIX Security 2016: 25th USENIX Security Symposium*, Thorsten Holz and Stefan Savage (Eds.). USENIX Association, Austin, TX, USA, 327–343.

[6] José Bacelar Almeida, Manuel Barbosa, Gilles Barthe, Arthur Blot, Benjamin Grégoire, Vincent Laporte, Tiago Oliveira, Hugo Pacheco, Benedikt Schmidt, and Pierre-Yves Strub. 2017. Jasmin: High-Assurance and High-Speed Cryptography. In *ACM CCS 2017: 24th Conference on Computer and Communications Security*, Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu (Eds.). ACM Press, Dallas, TX, USA, 1807–1823. https://doi.org/10.1145/3133956.3134078

[7] José Bacelar Almeida, Manuel Barbosa, Gilles Barthe, Benjamin Grégoire, Adrien Koutsos, Vincent Laporte, Tiago Oliveira, and Pierre-Yves Strub. 2020. The Last Mile: High-Assurance and High-Speed Cryptographic Implementations. In *2020 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, San Francisco, CA, USA, 965–982. https://doi.org/10.1109/SP40000.2020.00028

[8] Andris Ambainis, Mike Hamburg, and Dominique Unruh. 2019. Quantum Security Proofs Using Semi-classical Oracles. In *Advances in Cryptology – CRYPTO 2019, Part II (Lecture Notes in Computer Science, Vol. 11693)*, Alexandra Boldyreva and Daniele Micciancio (Eds.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 269–295. https://doi.org/10.1007/978-3-030-26951-7_10

[9] Yawning Angel, Benjamin Dowling, Andreas Hülsing, Peter Schwabe, and Florian Weber. 2022. Post Quantum Noise. , 97–109 pages. http://cryptojedi.org/papers/#pqnoise.

[10] Reza Azarderakhsh, David Jao, and Christopher Leonardi. 2017. Post-Quantum Static-Static Key Agreement Using Multiple Protocol Instances. In *SAC 2017: 24th Annual International Workshop on Selected Areas in Cryptography (Lecture Notes in Computer Science, Vol. 10719)*, Carlisle Adams and Jan Camenisch (Eds.). Springer, Heidelberg, Germany, Ottawa, ON, Canada, 45–63. https://doi.org/10.1007/978-3-319-72565-9_3

[11] Gustavo Banegas, Daniel J. Bernstein, Fabio Campos, Tung Chou, Tanja Lange, Michael Meyer, Benjamin Smith, and Jana Sotáková. 2021. CTIDH: faster constant-time CSIDH. *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2021, 4 (2021), 351–387. https://doi.org/10.46586/tches.v2021.i4.351-387 https://tches.iacr.org/index.php/TCHES/article/view/9069.

[12] Mihir Bellare and Phillip Rogaway. 1993. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM CCS 93: 1st Conference on Computer and Communications Security*, Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby (Eds.). ACM Press, Fairfax, Virginia, USA, 62–73. https://doi.org/10.1145/168588.168596

[13] Mihir Bellare and Phillip Rogaway. 2006. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In *Advances in Cryptology – EUROCRYPT 2006 (Lecture Notes in Computer Science, Vol. 4004)*, Serge Vaudenay (Ed.). Springer, Heidelberg, Germany, St. Petersburg, Russia, 409–426. https://doi.org/10.1007/11761679_25

[14] Daniel J. Bernstein. 2006. Curve25519: New Diffie-Hellman Speed Records. In *PKC 2006: 9th International Conference on Theory and Practice of Public Key Cryptography (Lecture Notes in Computer Science, Vol. 3958)*, Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin (Eds.). Springer, Heidelberg, Germany, New York, NY, USA, 207–228. https://doi.org/10.1007/11745853_14

[15] Daniel J. Bernstein, Billy Bob Brumley, Ming-Shing Chen, and Nicola Tuveri. 2022. OpenSSLNTRU: Faster post-quantum TLS key exchange. In *31st USENIX Security Symposium (USENIX Security 22)*. USENIX Association, Boston, MA, 845–862. https://www.usenix.org/conference/usenixsecurity22/presentation/bernstein

[16] Daniel J. Bernstein, Tanja Lange, Chloe Martindale, and Lorenz Panny. 2019. Quantum Circuits for the CSIDH: Optimizing Quantum Evaluation of Isogenies. In *Advances in Cryptology – EUROCRYPT 2019, Part II (Lecture Notes in Computer Science, Vol. 11477)*, Yuval Ishai and Vincent Rijmen (Eds.). Springer, Heidelberg, Germany, Darmstadt, Germany, 409–441. https://doi.org/10.1007/978-3-030-17656-3_15

[17] Manuel Blum, Paul Feldman, and Silvio Micali. 1988. Non-Interactive Zero-Knowledge and Its Applications (Extended Abstract). In *20th Annual ACM Symposium on Theory of Computing*. ACM Press, Chicago, IL, USA, 103–112. https://doi.org/10.1145/62212.62222

[18] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. 2011. Random Oracles in a Quantum World. In *Advances in Cryptology – ASIACRYPT 2011 (Lecture Notes in Computer Science, Vol. 7073)*, Dong Hoon Lee and Xiaoyun Wang (Eds.). Springer, Heidelberg, Germany, Seoul, South Korea, 41–69. https://doi.org/10.1007/978-3-642-25385-0_3

[19] Dan Boneh and Mark Zhandry. 2014. Multiparty Key Exchange, Efficient Traitor Tracing, and More from Indistinguishability Obfuscation. In *Advances in Cryptology – CRYPTO 2014, Part I (Lecture Notes in Computer Science, Vol. 8616)*, Juan A. Garay and Rosario Gennaro (Eds.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 480–499. https://doi.org/10.1007/978-3-662-44371-2_27

[20] Xavier Bonnetain and André Schrottenloher. 2020. Quantum Security Analysis of CSIDH. In *Advances in Cryptology – EUROCRYPT 2020, Part II (Lecture Notes in Computer Science, Vol. 12106)*, Anne Canteaut and Yuval Ishai (Eds.). Springer, Heidelberg, Germany, Zagreb, Croatia, 493–522. https://doi.org/10.1007/978-3-030-45724-2_17

[21] Joppe W. Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. 2016. Frodo: Take off the Ring! Practical, Quantum-Secure Key Exchange from LWE. In *ACM CCS 2016: 23rd Conference on Computer and Communications Security*, Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi (Eds.). ACM Press, Vienna, Austria, 1006–1018. https://doi.org/10.1145/2976749.2978425

[22] Joppe W. Bos, Craig Costello, Michael Naehrig, and Douglas Stebila. 2015. Post-Quantum Key Exchange for the TLS Protocol from the Ring Learning with Errors

Problem. In *2015 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, San Jose, CA, USA, 553–570. https://doi.org/10.1109/SP.2015.40

[23] Colin Boyd, Yvonne Cliff, Juan González Nieto, and Kenneth G. Paterson. 2008. Efficient One-Round Key Exchange in the Standard Model. In *ACISP 08: 13th Australasian Conference on Information Security and Privacy (Lecture Notes in Computer Science, Vol. 5107)*, Yi Mu, Willy Susilo, and Jennifer Seberry (Eds.). Springer, Heidelberg, Germany, Wollongong, Australia, 69–83.

[24] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. 2013. Classical hardness of learning with errors. In *45th Annual ACM Symposium on Theory of Computing*, Dan Boneh, Tim Roughgarden, and Joan Feigenbaum (Eds.). ACM Press, Palo Alto, CA, USA, 575–584. https://doi.org/10.1145/2488608.2488680

[25] Jacqueline Brendel, Rune Fiedler, Felix Günther, Christian Janson, and Douglas Stebila. 2022. Post-quantum Asynchronous Deniable Key Exchange and the Signal Handshake. In *Public-Key Cryptography – PKC 2022*, Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe (Eds.). Springer International Publishing, Cham, 3–34.

[26] Jacqueline Brendel, Marc Fischlin, Felix Günther, Christian Janson, and Douglas Stebila. 2020. Towards Post-Quantum Security for Signal's X3DH Handshake. In *SAC 2020: 27th Annual International Workshop on Selected Areas in Cryptography (Lecture Notes in Computer Science, Vol. 12804)*, Orr Dunkelman, Michael J. Jacobson Jr., and Colin O'Flynn (Eds.). Springer, Heidelberg, Germany, Halifax, NS, Canada (Virtual Event), 404–430. https://doi.org/10.1007/978-3-030-81652-0_16

[27] Leon Groot Bruinderink, Andreas Hülsing, Tanja Lange, and Yuval Yarom. 2016. Flush, Gauss, and Reload - A Cache Attack on the BLISS Lattice-Based Signature Scheme. In *Cryptographic Hardware and Embedded Systems – CHES 2016 (Lecture Notes in Computer Science, Vol. 9813)*, Benedikt Gierlichs and Axel Y. Poschmann (Eds.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 323–345. https://doi.org/10.1007/978-3-662-53140-2_16

[28] David Cash, Eike Kiltz, and Victor Shoup. 2008. The Twin Diffie-Hellman Problem and Applications. In *Advances in Cryptology – EUROCRYPT 2008 (Lecture Notes in Computer Science, Vol. 4965)*, Nigel P. Smart (Ed.). Springer, Heidelberg, Germany, Istanbul, Turkey, 127–145. https://doi.org/10.1007/978-3-540-78967-3_8

[29] Wouter Castryck and Thomas Decru. 2022. An efficient key recovery attack on SIDH (preliminary version). Cryptology ePrint Archive, Report 2022/975. https://eprint.iacr.org/2022/975.

[30] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. 2018. CSIDH: An Efficient Post-Quantum Commutative Group Action. In *Advances in Cryptology – ASIACRYPT 2018, Part III (Lecture Notes in Computer Science, Vol. 11274)*, Thomas Peyrin and Steven Galbraith (Eds.). Springer, Heidelberg, Germany, Brisbane, Queensland, Australia, 395–427. https://doi.org/10.1007/978-3-030-03332-3_15

[31] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. 2017. Post-Quantum Zero-Knowledge and Signatures from Symmetric-Key Primitives. In *ACM CCS 2017: 24th Conference on Computer and Communications Security*, Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu (Eds.). ACM Press, Dallas, TX, USA, 1825–1842. https://doi.org/10.1145/3133956.3133997

[32] Craig Costello, Patrick Longa, and Michael Naehrig. 2016. Efficient Algorithms for Supersingular Isogeny Diffie-Hellman. In *Advances in Cryptology – CRYPTO 2016, Part I (Lecture Notes in Computer Science, Vol. 9814)*, Matthew Robshaw and Jonathan Katz (Eds.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 572–601. https://doi.org/10.1007/978-3-662-53018-4_21

[33] Bor de Kock. 2018. *A non-interactive key exchange based on ring-learning with errors.* Master's thesis. Master's thesis, Eindhoven University of Technology.

[34] Whitfield Diffie and Martin E. Hellman. 1976. New Directions in Cryptography. *IEEE Transactions on Information Theory* 22, 6 (1976), 644–654.

[35] Jintai Ding, Xiang Xie, and Xiaodong Lin. 2012. A Simple Provably Secure Key Exchange Scheme Based on the Learning with Errors Problem. Cryptology ePrint Archive, Report 2012/688. https://eprint.iacr.org/2012/688.

[36] Samuel Dobson and Steven D. Galbraith. 2022. Post-Quantum Signal Key Agreement from SIDH. In *Post-Quantum Cryptography*, Jung Hee Cheon and Thomas Johansson (Eds.). Springer International Publishing, Cham, 422–450.

[37] Julien Duman, Dominik Hartmann, Eike Kiltz, Sabrina Kunzweiler, Jonas Lehmann, and Doreen Riepel. 2022. Group Action Key Encapsulation and Non-Interactive Key Exchange in the QROM. Cryptology ePrint Archive, Paper 2022/1230. https://eprint.iacr.org/2022/1230 https://eprint.iacr.org/2022/1230.

[38] Morris J. Dworkin. 2015. *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions.* Technical Report. National Institute of Standards and Technology. https://doi.org/10.6028/nist.fips.202

[39] Thomas Espitau, Pierre-Alain Fouque, Benoît Gérard, and Mehdi Tibouchi. 2017. Side-Channel Attacks on BLISS Lattice-Based Signatures: Exploiting Branch Tracing against strongSwan and Electromagnetic Emanations in Microcontrollers. In *ACM CCS 2017: 24th Conference on Computer and Communications Security*, Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu (Eds.). ACM Press, Dallas, TX, USA, 1857–1874. https://doi.org/10.1145/3133956.3134028

[40] Amos Fiat and Adi Shamir. 1987. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Advances in Cryptology – CRYPTO'86 (Lecture Notes in Computer Science, Vol. 263)*, Andrew M. Odlyzko (Ed.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 186–194. https://doi.org/10.1007/3-540-47721-7_12

[41] Eduarda S. V. Freire, Dennis Hofheinz, Eike Kiltz, and Kenneth G. Paterson. 2013. Non-Interactive Key Exchange. In *PKC 2013: 16th International Conference on Theory and Practice of Public Key Cryptography (Lecture Notes in Computer Science, Vol. 7778)*, Kaoru Kurosawa and Goichiro Hanaoka (Eds.). Springer, Heidelberg, Germany, Nara, Japan, 254–271. https://doi.org/10.1007/978-3-642-36362-7_17

[42] Eiichiro Fujisaki and Tatsuaki Okamoto. 1999. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In *Advances in Cryptology – CRYPTO'99 (Lecture Notes in Computer Science, Vol. 1666)*, Michael J. Wiener (Ed.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 537–554. https://doi.org/10.1007/3-540-48405-1_34

[43] Eiichiro Fujisaki and Tatsuaki Okamoto. 2013. Secure Integration of Asymmetric and Symmetric Encryption Schemes. *Journal of Cryptology* 26, 1 (Jan. 2013), 80–101. https://doi.org/10.1007/s00145-011-9114-1

[44] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. 2016. On the Security of Supersingular Isogeny Cryptosystems. In *Advances in Cryptology – ASIACRYPT 2016, Part I (Lecture Notes in Computer Science, Vol. 10031)*, Jung Hee Cheon and Tsuyoshi Takagi (Eds.). Springer, Heidelberg, Germany, Hanoi, Vietnam, 63–91. https://doi.org/10.1007/978-3-662-53887-6_3

[45] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. 1985. The Knowledge Complexity of Interactive Proof-Systems (Extended Abstract). In *17th Annual ACM Symposium on Theory of Computing*. ACM Press, Providence, RI, USA, 291–304. https://doi.org/10.1145/22145.22178

[46] Siyao Guo, Pritish Kamath, Alon Rosen, and Katerina Sotiraki. 2020. Limits on the Efficiency of (Ring) LWE Based Non-interactive Key Exchange. In *PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part I (Lecture Notes in Computer Science, Vol. 12110)*, Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas (Eds.). Springer, Heidelberg, Germany, Edinburgh, UK, 374–395. https://doi.org/10.1007/978-3-030-45374-9_13

[47] Keitaro Hashimoto, Shuichi Katsumata, Kris Kwiatkowski, and Thomas Prest. 2021. An Efficient and Generic Construction for Signal's Handshake (X3DH): Post-Quantum, State Leakage Secure, and Deniable. In *PKC 2021: 24th International Conference on Theory and Practice of Public Key Cryptography, Part II (Lecture Notes in Computer Science, Vol. 12711)*, Juan Garay (Ed.). Springer, Heidelberg, Germany, Virtual Event, 410–440. https://doi.org/10.1007/978-3-030-75248-4_15

[48] Andreas Hülsing, Kai-Chun Ning, Peter Schwabe, Florian Weber, and Philip R. Zimmermann. 2021. Post-quantum WireGuard. In *2021 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, San Francisco, CA, USA, 304–321. https://doi.org/10.1109/SP40001.2021.00030

[49] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, and David Urbanik. 2017. *SIKE.* Technical Report. National Institute of Standards and Technology. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-1-submissions.

[50] David Jao and Luca De Feo. 2011. Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies. In *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, Bo-Yin Yang (Ed.). Springer, Heidelberg, Germany, Tapei, Taiwan, 19–34. https://doi.org/10.1007/978-3-642-25405-5_2

[51] John Kelsey, Shu jen Change, and Ray Perlner. 2016. *SHA-3 derived functions: cSHAKE, KMAC, TupleHash and ParallelHash.* Technical Report. National Institute of Standards and Technology. https://doi.org/10.6028/nist.sp.800-185

[52] Hugo Krawczyk and Hoeteck Wee. 2016. The OPTLS Protocol and TLS 1.3. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, Saarbruecken, Germany, 81–96. https://doi.org/10.1109/eurosp.2016.18

[53] Kris Kwiatkowski and Luke Valenta. 2019. The TLS Post-Quantum Experiment. Post on the Cloudflare blog. https://blog.cloudflare.com/the-tls-post-quantum-experiment/.

[54] Adam Langley. 2016. CECPQ1 results. Blog post. https://www.imperialviolet.org/2016/11/28/cecpq1.html.

[55] Adam Langley. 2018. CECPQ2. Blog post. https://www.imperialviolet.org/2018/12/12/cecpq2.html.

[56] Adeline Langlois and Damien Stehlé. 2015. Worst-Case to Average-Case Reductions for Module Lattices. *Designs, Codes and Cryptography* 75, 3 (2015), 565–599.

[57] Vadim Lyubashevsky. 2017. Converting NewHope/LWE key exchange to a Diffe-Hellman-like algorithm. Crypto Stack Exchange. https://crypto.stackexchange.com/questions/48146/converting-newhope-lwe-key-exchange-to-a-diffe-hellman-like-algorithm [Online:] https://crypto.stackexchange.com/questions/48146/converting-newhope-lwe-key-exchange-to-a-diffe-hellman-like-algorithm.

[58] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. 2022. *CRYSTALS-DILITHIUM.* Technical Report. National Institute of Standards and Technology. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

[59] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. 2022. Lattice-Based Zero-Knowledge Proofs and Applications: Shorter, Simpler, and More General. In *Advances in Cryptology – CRYPTO 2022, Part II (Lecture Notes in Computer Science, Vol. 13508)*, Yevgeniy Dodis and Thomas Shrimpton (Eds.). Springer, Heidelberg, Germany, Santa Barbara, CA, USA, 71–101. https://doi.org/10.1007/978-3-031-15979-4_3

[60] Moxie Marlinspike and Trevor Perrin. 2016. The X3DH Key Agreement Protocol (Revision 1). Part of the Signal Protocol Documentation. https://signal.org/docs/specifications/x3dh/x3dh.pdf.

[61] Peter L. Montgomery. 1985. Modular Multiplication without Trial Division. *Math. Comp.* 44, 170 (1985), 519–521.

[62] Kaushik Nath and Daniel J. Bernstein. 2022. lib25519. https://lib25519.cr.yp.to/.

[63] NIST. 2016. Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf.

[64] Christian Paquin, Douglas Stebila, and Goutam Tamvada. 2020. Benchmarking Post-quantum Cryptography in TLS. In *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, Jintai Ding and Jean-Pierre Tillich (Eds.). Springer, Heidelberg, Germany, Paris, France, 72–91. https://doi.org/10.1007/978-3-030-44223-1_5

[65] Chris Peikert. 2020. He Gives C-Sieves on the CSIDH. In *Advances in Cryptology – EUROCRYPT 2020, Part II (Lecture Notes in Computer Science, Vol. 12106)*, Anne Canteaut and Yuval Ishai (Eds.). Springer, Heidelberg, Germany, Zagreb, Croatia, 463–492. https://doi.org/10.1007/978-3-030-45724-2_16

[66] Trevor Perrin. 2018. Noise Protocol Framework. https://noiseprotocol.org/noise.pdf (Revision 34 vom 2018-07-11).

[67] Peter Pessl, Leon Groot Bruinderink, and Yuval Yarom. 2017. To BLISS-B or not to be: Attacking strongSwan's Implementation of Post-Quantum Signatures. In *ACM CCS 2017: 24th Conference on Computer and Communications Security*, Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu (Eds.). ACM Press, Dallas, TX, USA, 1843–1855. https://doi.org/10.1145/3133956.3134023

[68] Rachel Player. 2018. *Parameter selection in lattice-based cryptography.* Ph. D. Dissertation. Royal Holloway, University of London.

[69] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. 2022. *FALCON.* Technical Report. National Institute of Standards and Technology. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

[70] Oded Regev. 2005. On lattices, learning with errors, random linear codes, and cryptography. In *37th Annual ACM Symposium on Theory of Computing*, Harold N. Gabow and Ronald Fagin (Eds.). ACM Press, Baltimore, MA, USA, 84–93. https://doi.org/10.1145/1060590.1060603

[71] Eric Rescorla. 2018. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446. https://doi.org/10.17487/RFC8446

[72] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, Damien Stehlé, and Jintai Ding. 2022. *CRYSTALS-KYBER.* Technical Report. National Institute of Standards and Technology. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

[73] Peter Schwabe, Douglas Stebila, and Thom Wiggers. 2020. Post-Quantum TLS Without Handshake Signatures. In *ACM CCS 2020: 27th Conference on Computer and Communications Security*, Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna (Eds.). ACM Press, Virtual Event, USA, 1461–1480. https://doi.org/10.1145/3372297.3423350

[74] Gregor Seiler. 2018. Faster AVX2 optimized NTT multiplication for Ring-LWE lattice cryptography. Cryptology ePrint Archive, Report 2018/039. https://eprint.iacr.org/2018/039.

[75] Peter W. Shor. 1994. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In *35th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, Santa Fe, NM, USA, 124–134. https://doi.org/10.1109/SFCS.1994.365700

[76] Sara Stadler, Vitor Sakaguti, Harjot Kaur, and Anna Lena Fehlhaber. 2021. Hybrid Signal protocol for post-quantum email encryption. Cryptology ePrint Archive, Report 2021/875. https://eprint.iacr.org/2021/875.

[77] Dominique Unruh. 2015. Non-Interactive Zero-Knowledge Proofs in the Quantum Random Oracle Model. In *Advances in Cryptology – EUROCRYPT 2015, Part II (Lecture Notes in Computer Science, Vol. 9057)*, Elisabeth Oswald and Marc Fischlin (Eds.). Springer, Heidelberg, Germany, Sofia, Bulgaria, 755–784. https://doi.org/10.1007/978-3-662-46803-6_25

[78] Bas Westerbaan and Cefan Daniel Rubin. 2019. Defending against future threats: Cloudflare goes post-quantum. Post on the Cloudflare blog. https://blog.cloudflare.com/post-quantum-for-all/.

A PROOFS FOR SECTION 4 (DEFINITIONS)

A.1 Non-Interactive Zero-Knowledge Proofs

Zero-Knowledge proofs [45] allow a verifier to convince a prover of the validity of a statement without revealing anything beyond that. In the random oracle model [12] zero-knowledge proofs can be made non-interactive [17] by applying the Fiat-Shamir transformation [40].

Definition A.1 (Zero-Knowledge Proof of Knowledge). A zero-knowledge proof of knowledge ZKPoK for an NP language $\mathcal{L}$ [6] is defined as a tuple $\mathsf{ZKPoK} := (\mathsf{ZK.Prv}, \mathsf{ZK.Ver})$ of the following oracle algorithms.

$\pi \xleftarrow{\$} \mathsf{ZK.Prv}^{\mathsf{H}}(x, w)$: Given a statement x and a witness w, the probabilistic prover algorithm ZK.Prv returns a proof π.

$1/0 \leftarrow \mathsf{ZK.Ver}^{\mathsf{H}}(x, \pi)$: Given a statement x and a proof π, the deterministic verifier algorithm returns either 1 for accept or 0 for reject.

Similar to the work of [77] we assume a distribution RODist on functions, modelling the distribution of our random oracle. That is, given a random oracle $\mathsf{H} : \{0,1\}^* \rightarrow \{0,1\}^n$, RODist would be the uniform distribution on $\{0,1\}^* \rightarrow \{0,1\}^n$.

ZKPoK Security Notions. Besides *completeness*, which captures that valid proofs are accepted by the verifier, a zero-knowledge proof of knowledge should fulfil two additional properties; *soundness* ensures a cheating prover cannot convince the verifier of a false proof, and *zero-knowledge* conveys that the verifier learns nothing from its interaction with the prover beyond the fact that he knows a valid witness to the proof. We make this more precise with the following definitions and note that we require the strong notion of *simulation soundness with a straight-line extractor* [77], sometimes referred to as "online extractability" in the literature.

Definition A.2 (Completeness). Completeness for a zero-knowledge proof of knowledge ZKPoK of an NP language $\mathcal{L}$ is defined via the game $\mathbf{CMPLT}_{\mathsf{ZKPoK}}$ depicted in Figure 8. For an adversary A, we define A's advantage in $\mathbf{CMPLT}_{\mathsf{ZKPoK}}$ as

$$\mathsf{Adv}^{\mathbf{CMPLT}}_{\mathsf{ZKPoK}}(\mathsf{A}) := \Pr[\mathbf{CMPLT}^{\mathsf{A}}_{\mathsf{ZKPoK}} \Rightarrow 1],$$

and say that ZKPoK is $(\epsilon, Q_{\mathsf{H}})$-**CMPLT** if for all quantum-polynomial-time adversaries A, making at most Q_{H} queries (possibly in superposition) to the random oracle H, we have $\mathsf{Adv}^{\mathbf{CMPLT}}_{\mathsf{ZKPoK}}(\mathsf{A}) \leq \epsilon(Q_{\mathsf{H}})$.

[6]The language $\mathcal{L}$ is defined as the set of all yes-instances of the relation R, i.e. $\mathcal{L} = \{x : \exists\ w \text{ s.t. } R(x, w) = 1\}$.

```
Game CMPLT_ZKPoK
H ←$ RODist
(x, w) ←$ A^{|H⟩}
π ←$ ZK.Prv^H(x, w)
return ⟦ZK.Ver^H(x, π) = 0 ∧ R(x, w) = 1⟧
```

Figure 8: Game defining $\mathbf{CMPLT}_{\mathsf{ZKPoK}}$ for a zero-knowledge proof of knowledge ZKPoK with adversary A.

For the following notions we additionally require a simulator $\mathsf{ZK.Sim} := (\mathsf{ZK.Sim}_1, \mathsf{ZK.Sim}_2)$ that is split into two classical algorithms $\mathsf{ZK.Sim}_1$ and $\mathsf{ZK.Sim}_2$, where:

$\mathsf{H} \xleftarrow{\$} \mathsf{ZK.Sim}_1$: The probabilistic simulator algorithm $\mathsf{ZK.Sim}_1$ returns a circuit H which represents the initial simulated random oracle.

$\pi \xleftarrow{\$} \mathsf{ZK.Sim}_2(x)$: Given a statement x the stateful simulator algorithm $\mathsf{ZK.Sim}_2$ returns a proof π. Additionally, $\mathsf{ZK.Sim}_2$ is given access to the description of H and may replace it with a different description (i.e. it can program the random oracle).

Definition A.3 (Zero-Knowledge [45]). Zero-knowledge for a zero-knowledge proof of knowledge ZKPoK of an NP language $\mathcal{L}$ is defined via the game $\mathbf{ZK}^b_{\mathsf{ZKPoK}}$, depicted in Figure 9, where $\mathbf{ZK}^b_{\mathsf{ZKPoK}}$ is parametrised by a bit b. For an adversary A, we define A's advantage in $\mathbf{ZK}^b_{\mathsf{ZKPoK}}$ as

$$\mathsf{Adv}^{\mathbf{ZK}}_{\mathsf{ZKPoK}}(\mathsf{A}) := \left|\Pr[\mathbf{ZK}^{0,\mathsf{A}}_{\mathsf{ZKPoK}} \Rightarrow 1] - \Pr[\mathbf{ZK}^{1,\mathsf{A}}_{\mathsf{ZKPoK}} \Rightarrow 1]\right|,$$

and say that ZKPoK is (ϕ, Q_H)-**ZK**, if there exists a PPT simulator $\mathsf{ZK.Sim} := (\mathsf{ZK.Sim}_1, \mathsf{ZK.Sim}_2)$, such that for all quantum-polynomial-time adversaries A, making at most Q_H queries (possibly in superposition) to the random oracle H, we have $\mathsf{Adv}^{\mathbf{ZK}}_{\mathsf{ZKPoK}}(\mathsf{A}) \le \phi(Q_\mathsf{H})$.

```
Game ZK^0_ZKPoK
H ←$ RODist
b' ←$ A^{|H⟩, ZK.Prv(·,·)}
return ⟦b' = 0⟧

Game ZK^1_ZKPoK
H ←$ ZK.Sim_1
b' ←$ A^{|H⟩, ZK.Sim'_2(·,·)}
return ⟦b' = 1⟧

Procedure ZK.Sim'_2(x, w)
if R(x, w) = 0 :
    return ⊥
else :
    return ZK.Sim_2(x)
```

Figure 9: Games defining $\mathbf{ZK}^b_{\mathsf{ZKPoK}}$ for a zero-knowledge proof of knowledge ZKPoK with adversary A and simulator $\mathsf{ZK.Sim} := (\mathsf{ZK.Sim}_1, \mathsf{ZK.Sim}_2)$. The purpose of $\mathsf{ZK.Sim}'_2(\cdot,\cdot)$ is merely to serve as an interface for the adversary who expects a prover taking two arguments x and w.

Definition A.4 (Simulation-Sound Online-Extractability [77]). Simulation-sound online-extractability [7] for a zero-knowledge proof of knowledge ZKPoK of an NP language $\mathcal{L}$ is defined via the game $\mathbf{SSND}_{\mathsf{ZKPoK}}$, depicted in Figure 10. For an adversary A, we define A's advantage in $\mathbf{SSND}_{\mathsf{ZKPoK}}$ as

$$\mathsf{Adv}^{\mathbf{SSND}}_{\mathsf{ZKPoK}}(\mathsf{A}) := \Pr[\mathbf{SSND}^{\mathsf{A}}_{\mathsf{ZKPoK}} \Rightarrow 1],$$

and say that ZKPoK is (ψ, Q_H)-**SSND** relative to a simulator $\mathsf{ZK.Sim} := (\mathsf{ZK.Sim}_1, \mathsf{ZK.Sim}_2)$, if there exists a PPT extractor ZK.Ext such that for all quantum-polynomial-time adversaries A, making at most Q_H queries to the random oracle H, we have $\mathsf{Adv}^{\mathbf{SSND}}_{\mathsf{ZKPoK}}(\mathsf{A}) \le \psi(Q_\mathsf{H})$.

```
Game SSND_ZKPoK
H ←$ ZK.Sim_1
(x, π) ←$ A^{|H⟩, ZK.Sim_2(·)}
w ←$ ZK.Ext(H, x, π)
return ⟦ZK.Ver^H(x, π) = 1 ∧ R(x, w) = 0 ∧ (x, π) ∉ π̃⟧
```

Figure 10: Games defining $\mathbf{SSND}_{\mathsf{ZKPoK}}$ for a zero-knowledge proof of knowledge ZKPoK with adversary A, simulator $\mathsf{ZK.Sim} := (\mathsf{ZK.Sim}_1, \mathsf{ZK.Sim}_2)$ and extractor ZK.Ext. Here, $\tilde{\boldsymbol{\pi}}$ denotes the set of all proofs returned by $\mathsf{ZK.Sim}_2(\cdot)$ (together with the corresponding statements).

B PROOFS FOR SECTION 5 (CONSTRUCTION)

Lemma 1 (Honest Correctness). *For all (possibly unbounded) adversaries A the non-interactive key exchange NIKE := (Stp, Gen, SdK) construction depicted in Figure 5 has honest correctness error*

$$\delta \le \frac{4\beta^2 d^2 N}{q}$$

as per Definition 4.2.

Proof. The proof strategy is similar to the proof of Theorem 5.1, except that we can bound the probability of any coefficient of $\boldsymbol{k}^\star$ being in the interval

$$S^\star = \left[\frac{q}{4} \pm \beta^2 dN\right] \cup \left[\frac{3q}{4} \pm \beta^2 dN\right]$$

by $\frac{4\beta^2 d^2 N}{q}$, with a union bound over all coefficients. ■

Proof of Theorem 5.3. Let A be an adversary against NIKE in the **ActSec** game. Consider the sequence of games in Figure 11, where Q_{TQ} is the number of queries to `TestQue`.

Game G_0. This is the original $\mathbf{ActSec}^0_{\mathsf{NIKE},par}$ game, where the bit b is fixed to 0, hence

$$\Pr\left[G_0^\mathsf{A} \Rightarrow 1\right] = \Pr\left[\mathbf{ActSec}^{0,\mathsf{A}}_{\mathsf{NIKE},par} \Rightarrow 1\right].$$

[7]Online-extractability is sometimes referred to as straight line extractability in the literature.

Game G_1. In this game we modify the RegCorUsr oracle so that the secret key $\widetilde{sk'_{\mathsf{ID}}}$ is extracted from the proof π of a public key pk on Line 18, and stored with the identity ID. This requires the strong notion of simulation-sound online-extractability, from Definition A.4. If the extraction fails, then by default $\widetilde{sk'_{\mathsf{ID}}} = \perp$. I.e. the secret key is not stored, as in the original game. Therefore,

$$\left|\Pr\left[G_0^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[G_1^{\mathsf{A}} \Rightarrow 1\right]\right| \leq Q_{\mathsf{RCU}} \cdot \mathsf{Adv}^{\mathbf{SSND}}_{\mathsf{ZKPoK}}(\mathsf{B}_1).$$

Game G_2. In this game we introduce a new condition for aborting: If at any point in the simulation the adversary asks a query to the TestQue or to the RevCorQue oracles on two public keys that cause a key mismatch, then abort the simulation. Note that this condition is efficiently testable, as the game knows all the secret keys. We can bound the probability of this event happening with a reduction to the semi-malicious correctness property, Definition 4.3, of NIKE′ by

$$\left|\Pr\left[G_1^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[G_2^{\mathsf{A}} \Rightarrow 1\right]\right| \leq \mathsf{Adv}^{\mathbf{SM\text{-}COR}}_{\mathsf{NIKE}',par'}(\mathsf{B}_2).$$

Game G_3. In this game we modify the RevCorQue oracle on Line 29 and Line 34 to use the secret key that was extracted when the corresponding public key was registered as a corrupt key. Since the derived key is always the same for both secret keys, we get

$$\left|\Pr\left[G_2^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[G_3^{\mathsf{A}} \Rightarrow 1\right]\right| = 0.$$

Game $G_{4.0}$. In this game we modify the RegHonUsr oracle and replace the zero-knowledge proof of knowledge on Line 11 with a simulated proof. By the zero-knowledge property, Definition A.3, we get

$$\left|\Pr\left[G_3^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[G_{4.0}^{\mathsf{A}} \Rightarrow 1\right]\right| \leq Q_{\mathsf{RHU}} \cdot \mathsf{Adv}^{\mathbf{ZK}}_{\mathsf{ZKPoK}}(\mathsf{B}_4).$$

Game $G_{4.i}$. This is identical to the previous game, except that the i^{th} query to TestQue is answered with the bit b fixed to 1. We now state the following Lemma.

Claim (Reduction). There exists a pair of adversaries $\mathsf{B}_{3.i}$ and $\mathsf{B}'_{3.i}$ such that

$$\left|\Pr\left[G_{4.i}^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[G_{4.i+1}^{\mathsf{A}} \Rightarrow 1\right]\right| \leq \epsilon, \tag{1}$$

where

$$\epsilon = Q_{\mathsf{RHU}}^2 \cdot \mathsf{Adv}^{\mathbf{PasSec}}_{\mathsf{NIKE}',par'}(\mathsf{B}_{3.i}) + 2 \cdot \mathsf{Adv}^{\mathbf{SM\text{-}COR}}_{\mathsf{NIKE}',par'}(\mathsf{B}'_{3.i}).$$

Proof. To prove the Claim, we modify the game to guess two identities $\mathsf{ID}^\star$ and $\mathsf{ID}^{\star\star}$ that were queried to the RegHonUsr oracle. As a first modification, we no longer use the secret keys corresponding to $\mathsf{ID}^\star$ and $\mathsf{ID}^{\star\star}$ to answer any oracle queries, except for the query involving both $\mathsf{ID}^\star$ and $\mathsf{ID}^{\star\star}$. Since key mismatches cannot happen, the resulting game is in fact identical to the previous one.

Next, we further modify the game to no longer check for key mismatches in that involve the public keys associated with $\mathsf{ID}^\star$ and $\mathsf{ID}^{\star\star}$, this change is indistinguishable by another invocation of the semi-malicious correctness.

Next, switch the bit for the i^{th} query. To show that this change is indistinguishable, we construct a reduction $\mathsf{B}_{3.i}$ (for $i \in \{0, \ldots, Q_{\mathsf{TQ}} - 1\}$) against **PasSec** of NIKE′. As a first step, the reduction sets the public keys given by the challenger to be the keys associated with $\mathsf{ID}^\star$ and $\mathsf{ID}^{\star\star}$. RegHonUsr, RegCorUsr, and RevCorQue remain unchanged, and the only difference is in the case where the TestQue oracle is queried on $\mathsf{ID}^\star$ and $\mathsf{ID}^{\star\star}$. For the latter queries, we consider two cases:

- The query involving both $\mathsf{ID}^\star$ and $\mathsf{ID}^{\star\star}$ is the i^{th} query. In this case we simply answer with the key k_b provided by the reduction.
- The query involving both $\mathsf{ID}^\star$ and $\mathsf{ID}^{\star\star}$ is *not* the i^{th} query. In this case we abort the execution.

Note that, if the reduction correctly guesses $\mathsf{ID}^\star$ and $\mathsf{ID}^{\star\star}$ as being the identities queried in the i^{th} query of the TestQue oracle, then the second case does not happen and the reduction perfectly reproduces the view of the adversary with the bit $b = 0$ (in case k_b is the real key) or with the bit $b = 1$ (in case k_b is random). Since the reduction guesses correctly with probability at least $1/Q_{\mathsf{RHU}}^2$, the bound follows.

As the final change to the experiment, we undo the first and second modifications done above, namely, we check again for key mismatches for all keys and we no longer randomly sample two identities. Indistinguishability follows by a similar argument. Overall, this yields Equation (1).

■

Game $G_{4.Q_{\mathsf{TQ}}-1}$. In this game the bit b is fixed to 1. Applying a standard hybrid argument, yields

$$\left|\Pr\left[G_{4.Q_{\mathsf{TQ}}-1}^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[G_{4.0}^{\mathsf{A}} \Rightarrow 1\right]\right| \leq Q_{\mathsf{TQ}} \cdot \epsilon.$$

Game G_5. In this game we undo the changes made Games G_2 and G_3. Appealing to the semi-malicious correctness, we obtain

$$\left|\Pr\left[G_5^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[G_{4.Q_{\mathsf{TQ}}-1}^{\mathsf{A}} \Rightarrow 1\right]\right| \leq \mathsf{Adv}^{\mathbf{SM\text{-}COR}}_{\mathsf{NIKE}',par'}(\mathsf{B}_2).$$

Game G_6. In this game we undo the changes made to the RegHonUsr oracle in Game G_4 and replace the simulated proof with a real proof on Line 12. Appealing to Definition A.3 again, we get

$$\left|\Pr\left[G_6^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[G_5^{\mathsf{A}} \Rightarrow 1\right]\right| \leq Q_{\mathsf{RHU}} \cdot \mathsf{Adv}^{\mathbf{ZK}}_{\mathsf{ZKPoK}}(\mathsf{B}_4).$$

Game G_7. In this game we undo the changes made Game G_1. Using a similar argument, we obtain

$$\left|\Pr\left[G_7^{\mathsf{A}} \Rightarrow 1\right] - \Pr\left[G_6^{\mathsf{A}} \Rightarrow 1\right]\right| \leq Q_{\mathsf{RCU}} \cdot \mathsf{Adv}^{\mathbf{SSND}}_{\mathsf{ZKPoK}}(\mathsf{B}_1).$$

Observe that this game is the original $\mathbf{ActSec}^1_{\mathsf{NIKE},par}$ game. Hence,

$$\Pr\left[G_7^{\mathsf{A}} \Rightarrow 1\right] = \Pr\left[\mathbf{ActSec}^{1,\mathsf{A}}_{\mathsf{NIKE},par} \Rightarrow 1\right].$$

Collecting all probabilities yields the bound stated in Theorem 5.3.

■

Game $\mathsf{ActSec}^{b}_{\mathsf{NIKE},par}$

```
$\mathcal{D} := \bot$
$\mathcal{K} := \bot$
$b := 0$                                                        / $\mathsf{G}_0$
$b := 1$                                                        / $\mathsf{G}_{4.Q_{\mathtt{TQ}}}$
$b' \leftarrow \mathsf{A}^{\mathtt{RegHonUsr}(\cdot),\ \mathtt{RegCorUsr}(\cdot,\cdot),\ \mathtt{RevCorQue}(\cdot,\cdot),\ \mathtt{TestQue}(\cdot,\cdot)}$
return $[\![ b = b' ]\!]$
```

Oracle $\mathtt{RegHonUsr}(\mathtt{ID} \in \mathcal{IDS})$

```
if $(corrupt, \mathtt{ID}, \bot, \cdot) \in \mathcal{D}$ :
    return $\bot$
$(sk'_{\mathtt{ID}}, pk'_{\mathtt{ID}}) \xleftarrow{\$} \mathsf{Gen}(\mathtt{ID})$
$\pi \xleftarrow{\$} \mathsf{ZK.Prv}(pk'_{\mathtt{ID}}, sk'_{\mathtt{ID}})$          / $\mathsf{G}_0$
$\pi \xleftarrow{\$} \mathsf{ZK.Sim}_2(pk'_{\mathtt{ID}})$                         / $\mathsf{G}_{4.0}$
$\pi \xleftarrow{\$} \mathsf{ZK.Prv}(pk'_{\mathtt{ID}}, sk'_{\mathtt{ID}})$          / $\mathsf{G}_6$
$sk_{\mathtt{ID}} := sk'_{\mathtt{ID}}$
$pk_{\mathtt{ID}} := (pk'_{\mathtt{ID}}, \pi)$
$\mathcal{D} \cup \{(honest, \mathtt{ID}, sk_{\mathtt{ID}}, pk_{\mathtt{ID}})\}$
return $pk_{\mathtt{ID}}$
```

Oracle $\mathtt{RegCorUsr}(\mathtt{ID} \in \mathcal{IDS}, pk)$

```
parse $pk \longrightarrow (pk'_{\mathtt{ID}}, \pi)$
$\widetilde{sk'_{\mathtt{ID}}} \xleftarrow{\$} \mathsf{ZK.Ext}(\mathsf{H}, pk'_{\mathtt{ID}}, \pi)$        / $\mathsf{G}_1$-$\mathsf{G}_5$
if $(corrupt, \mathtt{ID}, \cdot, \cdot) \in \mathcal{D}$ :
    $(corrupt, \mathtt{ID}, \bot, \cdot) := (corrupt, \mathtt{ID}, \bot, pk)$
    $(corrupt, \mathtt{ID}, \cdot, \cdot) := (corrupt, \mathtt{ID}, \widetilde{sk'_{\mathtt{ID}}}, pk)$     / $\mathsf{G}_1$–$\mathsf{G}_5$
else :
    $\mathcal{D} \cup \{(corrupt, \mathtt{ID}, \bot, pk)\}$
    $\mathcal{D} \cup \left\{\left(corrupt, \mathtt{ID}, \widetilde{sk'_{\mathtt{ID}}}, pk\right)\right\}$          / $\mathsf{G}_1$–$\mathsf{G}_5$
```

Oracle $\mathtt{RevCorQue}(\mathtt{ID}_1, \mathtt{ID}_2)$

```
if $(honest, \mathtt{ID}_1, \cdot, \cdot) \in \mathcal{D} \wedge (corrupt, \mathtt{ID}_2, \cdot, \cdot) \in \mathcal{D}$ :
    if $\mathsf{SdK}(\mathtt{ID}_2, pk_2, \mathtt{ID}_1, sk_1) \neq \mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2)$ :   / $\mathsf{G}_2$-$\mathsf{G}_{4.Q_{\mathtt{TQ}}-1}$
        abort                                                   / $\mathsf{G}_2$-$\mathsf{G}_{4.Q_{\mathtt{TQ}}-1}$
    return $\mathsf{SdK}(\mathtt{ID}_2, pk_2, \mathtt{ID}_1, sk_1)$
    return $\mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, \widetilde{sk_2})$                           / $\mathsf{G}_3$
elseif $(corrupt, \mathtt{ID}_1, \cdot, \cdot) \in \mathcal{D} \wedge (honest, \mathtt{ID}_2, \cdot, \cdot) \in \mathcal{D}$ :
    if $\mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2) \neq \mathsf{SdK}(\mathtt{ID}_2, pk_2, \mathtt{ID}_1, sk_1)$ :   / $\mathsf{G}_2$-$\mathsf{G}_{4.Q_{\mathtt{TQ}}-1}$
        abort                                                   / $\mathsf{G}_2$-$\mathsf{G}_{4.Q_{\mathtt{TQ}}-1}$
    return $\mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2)$
    return $\mathsf{SdK}(\mathtt{ID}_2, pk_2, \mathtt{ID}_1, \widetilde{sk_1})$                           / $\mathsf{G}_3$
```

Oracle $\mathtt{TestQue}(\mathtt{ID}_1, \mathtt{ID}_2)$

```
if $\mathtt{ID}_1 = \mathtt{ID}_2$ :
    return $\bot$
if $(honest, \mathtt{ID}_1, \cdot, \cdot) \in \mathcal{D} \wedge (honest, \mathtt{ID}_2, \cdot, \cdot) \in \mathcal{D}$ :
    $b := 1$                                                    / $\mathsf{G}_{4.i}$
    if $b = 0$ :
        if $\mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2) \neq \mathsf{SdK}(\mathtt{ID}_2, pk_2, \mathtt{ID}_1, sk_1)$ : / $\mathsf{G}_2$-$\mathsf{G}_{4.Q_{\mathtt{TQ}}-1}$
            abort                                               / $\mathsf{G}_2$-$\mathsf{G}_{4.Q_{\mathtt{TQ}}-1}$
        $k := \mathsf{SdK}(\mathtt{ID}_1, pk_1, \mathtt{ID}_2, sk_2)$
    if $b = 1$ :
        if $(\mathtt{ID}_1, \mathtt{ID}_2, k) \in \mathcal{K} \vee (\mathtt{ID}_2, \mathtt{ID}_1, k) \in \mathcal{K}$ :
            return $k$
        $k \xleftarrow{\$} \mathcal{SKS}$
        $\mathcal{K} \cup \{(\mathtt{ID}_1, \mathtt{ID}_2, k)\}$
    return $k$
return $\bot$
```

Figure 11: Games G_0, G_1, G_3, $\mathsf{G}_{4.i}$ (for $i \in \{0 \leq i \leq Q_{\mathtt{TQ}} - 1\}$), G_6 for the proof of ActSec of NIKE in Figure 7.