On the Multi-User Security of LWE-based NIKE

Roman Langrehr 

ETH Zurich
roman.langrehr@inf.ethz.ch

Abstract. Non-interactive key exchange (NIKE) schemes like the Diffie-Hellman key exchange are a widespread building block in several cryptographic protocols. Since the Diffie-Hellman key exchange is not post-quantum secure, it is important to investigate post-quantum alternatives. We analyze the security of the LWE-based NIKE by Ding et al. (ePrint 2012) and Peikert (PQCrypt 2014) in a multi-user setting where the same public key is used to generate shared keys with multiple other users. The Diffie-Hellman key exchange achieves this security notion. The mentioned LWE-based NIKE scheme comes with an inherent correctness error (Guo et al., PKC 2020), and this has significant implications for the multi-user security, necessitating a closer examination.

Single-user security generically implies multi-user security when all users generate their keys honestly for NIKE schemes with negligible correctness error. However, the LWE-based NIKE requires a super-polynomial modulus to achieve a negligible correctness error, which makes the scheme less efficient. We show that

- generically, single-user security does not imply multi-user security when the correctness error is non-negligible, but despite this
- the LWE-based NIKE with polynomial modulus is multi-user secure for honest users when the number of users is fixed in advance. This result takes advantage of the leakage-resilience properties of LWE.

We then turn to a stronger model of multi-user security that allows adversarially generated public keys. For this model, we consider a variant of the LWE-based NIKE where each public key is equipped with a NIZKPoK of the secret key. Adding NIZKPoKs is a standard technique for this stronger model and Hesse et al. (Crypto 2018) showed that this is sufficient to achieve security in the stronger multi-user security model for perfectly correct NIKEs (which the LWE-based NIKE is not). We show that

- for certain parameters that include all parameters with polynomial modulus, the LWE-based NIKE can be efficiently attacked with adversarially generated public keys, despite the use of NIZKPoKs, but
- for suitable parameters (that require a super-polynomial modulus), this security notion is achieved by the LWE-based NIKE with NIZKPoKs.

This stronger security notion has been previously achieved for LWE-based NIKE only in the QROM, while all our results are in the standard model.

1 Introduction

Non-Interactive Key Exchange. Non-interactive key exchange (NIKE) schemes allow every pair of users to compute a common shared key, that is hidden to everybody else, using a public-key infrastructure. The first NIKE scheme was presented in the seminal work by Diffie and Hellman [20]. There, public keys are group elements g^{x_i} of a suitable prime-order group with a fixed generator g and the corresponding secret key is the discrete log x_i. The shared key for two users with public keys g^{x_i} and g^{x_j} is $K_{i,j} = g^{x_i \cdot x_j}$ and can be computed using x_i as $K_{i,j} = (g^{x_j})^{x_i}$ and similarly with x_j as $K_{i,j} = (g^{x_i})^{x_j}$. For an outsider who only sees the public keys g^{x_i} and g^{x_j}, the shared key $K_{i,j}$ is indistinguishable from a uniformly random group element by the Decisional Diffie-Hellman assumption.

The notion of a NIKE was only formalized much later by Cash, Kiltz, and Shoup [16] and further analyzed by Freire, Hofheinz, Kiltz, and Paterson [25]. NIKE schemes are a useful building block, especially when minimizing communication costs, for example, for wireless channels [15] or interactive key exchange [10]. Another interesting application of NIKE is deniable authentication [22], however this requires the NIKE to be perfectly correct and thus cannot be instantiated with the LWE-based NIKE considered in this work.

Although the Diffie-Hellman NIKE is very simple and efficient, it unfortunately is not secure against quantum computers. With the recent breaking [17, 38, 46] of SIDH [32], there are now, to the best of our knowledge, essentially two NIKE schemes that promise to achieve post-quantum security. The first is CSIDH [18], which, like SIDH, is based on isogenies, but is not affected by the attacks on SIDH. The other is the LWE-based NIKE introduced by Ding, Xie, and Lin [21] and Peikert [42]. This work is about the latter scheme.

NIKE from LWE. In the LWE-based NIKE from [21, 42] there are two different types of users, which we will call left and right users. A shared key can be computed only between a left and a right user. This can be easily converted to a full-fledged NIKE by generating for every user a left and a right key and deciding, based on some fixed rule, which user uses his left key and which user uses his right key for the shared key computation.

The users share a common random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$. Now each user samples a short vector $\mathbf{s}_i \in \mathbb{Z}_q^n$ (any short distribution with sufficient min-entropy will do), which will be its secret key, and an error vector $\mathbf{e}_i \in \mathbb{Z}^n$ whose entries are sampled according to a discrete Gaussian distribution. For a left user, the public key is $\mathsf{pk}_{L,i} = \mathbf{s}_i^\top \mathbf{A} + \mathbf{e}_i^\top$. For a right user, the public key is $\mathsf{pk}_{R,i} = \mathbf{A}\mathbf{s}_i + \mathbf{e}_i$. Now, a left user i and a right user j can both compute approximations of $\mathbf{s}_i^\top \mathbf{A} \mathbf{s}_j$: The left user can do so by sampling $e' \in \mathbb{Z}$ according to a discrete Gaussian distribution and computing

$$\mathbf{s}_i^\top \mathsf{pk}_{R,j} + e' = \mathbf{s}_i^\top \mathbf{A} \mathbf{s}_j + \underbrace{\mathbf{s}_i^\top \mathbf{e}_j + e'}_{\text{short}}.$$

Similarly, the right user can sample $e'' \in \mathbb{Z}$ according to a discrete Gaussian distribution and compute

$$\mathsf{pk}_{L,i}\mathbf{s}_j + e'' = \mathbf{s}_i^\top \mathbf{A}\mathbf{s}_j + \underbrace{\mathbf{e}_i^\top \mathbf{s}_j + e''}_{\text{short}}.$$

The additional error terms e' and e'' are necessary to prove security.[1]

By rounding these preliminary shared keys to, e.g., one bit, both users can obtain the same shared key with high probability. When a super-polynomial modulus-to-noise ratio is used, they will get the same bit with overwhelming probability, but with a polynomial modulus-to-noise ratio, there will be a non-negligible probability for not obtaining the same bit. Guo, Kamath, Rosen, and Sotiraki [28] showed that this is unavoidable for a large class of LWE-based NIKE schemes, even when multiple LWE samples are used per user.

Since LWE with polynomial modulus-to-noise ratio is more efficient due to the smaller modulus and is a qualitatively weaker assumption than LWE with super-polynomial modulus-to-noise ratio (it has better reductions to worst-case lattice problems [45, 43, 14]), it is still desirable to use a polynomial modulus-to-noise ratio when a small but non-negligible error probability is tolerable. This is the case, for example, in scenarios where interaction is possible but costly, such as [15, 10].

Most of this work focuses on the polynomial modulus-to-noise ratio.

Security models for NIKE. To study the security of a NIKE scheme, [16] introduced several security notions. We recall three of them which we use in this work.

The first notion is light security. Here, the adversary gets two public keys (here, a left public key and a right public key) and has to distinguish the shared key corresponding to these two public keys from a uniformly random key. This security notion captures scenarios where each public/secret key pair is used only for one shared key computation. However, for scalability and ease of use, it is often desirable to use the same key pair to compute shared keys with many other users.

The second notion we consider is adaptive HKR (honest key registration) security. It captures such scenarios, at least when all users generate their keys honestly. It is defined with a security game where the adversary can make adaptively the following operations via oracles:

- register a user and obtain their public key
- extract a user's secret key

[1] Using the learning with rounding assumption, the error terms e' and e'' could be avoided to prove "light security" (defined later). However, for the more advanced security notions considered in this work, rounding alone is not sufficient. Also, for super-polynomial modulus-to-noise ratio, the rounding can be omitted for light and adaptive HKR security because it has only negligible probability of changing the shared key.

- reveal the shared key of a pair of users (the adversary can specify which user contributes the secret key)
- (once) get challenged by either obtaining a real shared key (as in a reveal query) or a random shared key

The goal of the adversary is to guess whether he gets real or random shared keys in the challenge without making any query that makes this trivial (extracting the secret key of one challenge user or using the reveal oracle on the pair of challenge users).

The third and strongest security notion is adaptive DKR (dishonest key registration) security, where the adversary can do the same operations as in the adaptive HKR security game but additionally it can register users with adversarially chosen public keys. These users can be used to obtain shared keys with honest users (e.g., a challenge user) in the reveal oracle. The honest user's secret key is used here to compute the shared key.

In [25] it was shown that light security implies adaptive security, but this reduction requires that the NIKE scheme has a negligible correctness error. The reduction works by guessing both challenge users and using the public keys from the light security game as their public keys. Since it does not know the secret keys of the challenge users, the reduction answers reveal queries with one challenge user always with the secret key of the other, non-challenge user (here we rely on the negligible correctness error). The challenge query is answered with the purported shared key from the light security game. So far, this security model has been used directly only in the context of tight security (since the above reduction is not tight) [4, 29, 30], but we think it is also important to consider this notion for NIKE schemes with non-negligible correctness error, like the LWE-based construction with polynomial modulus-to-noise ratio.

In [29] it was furthermore shown that any adaptive HKR secure NIKE scheme with perfect correctness can be made adaptive DKR secure by equipping public keys with a non-interactive zero-knowledge proof of knowledge (NIZKPoK) of the secret key. Due to the requirement of perfect correctness, their transformation is not immediately applicable to the LWE-based NIKE schemes (no matter the modulus-to-noise ratio).

The original works on LWE-based NIKE only proved light security [21, 42]. A very recent work [26] showed that a variant of the LWE-based NIKE with super-polynomial modulus is adaptively DKR secure in the (quantum) random oracle model. Informally speaking, they achieved this by showing that the NIZKPoK approach of [29] can be applied even to NIKE schemes achieving only statistical correctness when we can guarantee that the correctness error is negligible even for adversarially generated public/secret key pairs (possibly depending on the other public key). They achieve this by adding a random offset to the unrounded shared keys that is obtained by querying the random oracle on the public keys and identities of the two users. This approach seems not to carry over to the standard model and cannot be applied to a polynomial modulus-to-noise ratio because this would require a polynomial bound on the number of (Q)ROM queries.

1.1 Our results and open questions.

This work analyzes the security notions achieved by the LWE-based NIKE of [21, 42], depending on the parameters.

Adaptive HKR security with polynomial modulus-to-noise ratio. A natural approach to prove adaptive HKR security for the LWE-based NIKE with polynomial modulus-to-noise ratio would be to generalize the generic result of [25] to NIKE with non-negligible correctness error. We show that this is unfortunately impossible. Namely, in Appendix B we give a separation between light and adaptive HKR security by constructing a NIKE scheme with correctness error $p(\lambda) \in [0, 1]$, for any efficiently computable function p, which achieves light security, but has an attacker against adaptive HKR security with advantage $\approx p(\lambda)$. This attacker only needs to register one additional user (apart from the two users for the challenge).

The main result of this work is that, fortunately, for suitable parameters (that still have a polynomial modulus-to-noise ratio), the LWE-based NIKE scheme can achieve adaptive HKR security. However, the parameters we need depend on the number of users N that the adversary registers in the security game. Thus, we only achieve adaptive HKR security for an (arbitrary) a priori bounded number of users. We call this N-bounded adaptive HKR security. An interesting open problem is whether the NIKE can also achieve adaptive HKR security for an unbounded number of users. We show some barriers that need to be overcome to prove this.

Adaptive DKR security. We then turn to adaptive DKR security. In this setting, we consider the NIKE where all public keys are equipped with a NIZKPoK of the secret key. We first show that when the noise that is added to the unrounded shared keys is of polynomial size, the scheme is not DKR secure by providing an attacker. The authors of [26] conjectured that their scheme is secure without the use of a (Q)ROM. However, the provided attack can break their scheme in time $\mathcal{O}(\sqrt{q/B_e})$ (in particular, the attack is independent of the LWE dimension) if the QROM is omitted and in polynomial time if the rounding function is additionally slightly changed. Especially since the particularities of the rounding function are irrelevant in their proof, the attack also shows barriers to prove DKR security without the QROM for the original rounding function.

We then show that when the noise added to the unrounded shared keys is super-polynomially larger than the size of the public keys and the size of the LWE secret, the scheme achieves adaptive DKR secure via the transformation of [29]. This requires a super-polynomial modulus-to-noise ratio. An interesting open problem is whether there is a different way to construct a NIKE from LWE with polynomial modulus-to-noise ratio that achieves adaptive DKR security.

Table 1 summarizes the results of this work.

Table 1. The security notions achieved by LWE-based NIKE. The highlighted results are from this work. In the adaptive DKR security column, poly. and super-poly. refer to the size of noise added to the unrounded shared keys. ✓ means that this security notion is achieved and ✗ means that there is an attack.

modulus-to-noise ratio	light security	adaptive HKR security	adaptive DKR security (with NIZKPoK)
polynomial	✓	✓ (bounded) ? (unbounded)	✗
super-polynomial	✓	✓	✗ poly. ✓ super-poly. ✓ with (Q)ROM [26]

1.2 Technical overview

We briefly recall the security proof of light security for the LWE-based NIKE. Let us assume without loss of generality that the challenge shared key is computed with the left user's secret key $\mathbf{s}_L^\star$. We need to show that $\mathbf{s}_L^\star \cdot \mathsf{pk}_R^\star$ looks random to an adversary that knows the public key of the left user $(\mathbf{s}_L^\star)^\top \mathbf{A} + (\mathbf{e}_L^\star)^\top$ and the right user's public key $\mathsf{pk}_R^\star = \mathbf{A}\mathbf{s}_R^\star + \mathbf{e}_R^\star$. The proof proceeds with a two-step hybrid argument. In the first step, $\mathsf{pk}_R^\star$ is replaced by a uniformly random value chosen from $\mathbb{Z}_q^n$, which is justified by the LWE assumption (with n samples). In the next and final step, we switch the left user's public key and the challenge shared key to uniformly random. To do this, we need $n+1$ samples $\mathbf{b} = (\mathbf{s}_L^\star)^\top(\mathbf{A}|\mathbf{v}) + (\mathbf{e}_L^\star)^\top$. The reduction uses $\mathbf{v}$ as the public key for the right user and the leftmost n entries of $\mathbf{b}$ as the public key for the left user. The rightmost entry $\mathbf{b}$ is then the shared key between these users. Since $\mathbf{b}$ is computationally indistinguishable from a uniformly random vector, light security follows.

Bounded HKR security for LWE-NIKE. To prove adaptive HKR security, we first have to make one minor change to the NIKE scheme. Namely, we need to make the shared key generation algorithm deterministic, to avoid that the adversary can learn multiple shared keys for the same pair of users with the same user contributing its secret key. We do this by adding a key for a pseudorandom function (PRF) to each user's secret key. In the shared key algorithm, the randomness is now replaced by the output of the PRF evaluated on the other user's identity (or public key).

The reduction then begins by guessing which of the users registered by the adversary will be used for the challenge query, just as in the generic reduction of [25]. Without loss of generality we assume again that the challenge shared key is computed with the left challenge user's secret key. The difficulty here is that the secret keys of the challenge users are also used in reveal shared key queries to reveal the shared key with other users. In [25] these queries are answered using the other, non-challenge user's secret key. Here, however, the reduction can not

directly use the other user's secret key to compute this shared key because it will differ with noticeable probability.

Trick 1: Using leakage. The first trick we use to be able to answer these reveal queries is to use leakage about the challenge user's secret key and noise to correct for the other user's secret key being used. Concretely, when the right challenge user with public key $\mathsf{pk}_R^\star = \mathbf{A}\mathbf{s}_R^\star + \mathbf{e}_R^\star$ computes the shared key with another left user with public key $\mathsf{pk}_L = \mathbf{s}_L^\top \mathbf{A} + \mathbf{e}_L^\top$ the right user gets as unrounded shared key

$$\mathbf{s}_L^\top \mathbf{A}\mathbf{s}_R^\star + \mathbf{e}_L^\top \mathbf{s}_R^\star + e'$$

while the left user gets

$$\mathbf{s}_L^\top \mathbf{A}\mathbf{s}_R^\star + \mathbf{s}_L^\top \mathbf{e}_R^\star + e''.$$

where e and e'' is a noise that is identically distributed in both cases. When the reduction gets the leakage $\mathbf{e}_L^\top \mathbf{s}_R^\star$ and $\mathbf{s}_L^\top \mathbf{e}_R^\star$, it can use the leakage to compute the shared key from the right user's perspective given the shared key from the left user's perspective, which the reduction can compute without the right user's secret key.

In [14, 11] it is shown that LWE (with short secrets) is hard even given arbitrary leakage of the secret $\mathbf{s}$, as long as enough min-entropy remains in $\mathbf{s}$. We use this result to justify the leakage of $\mathbf{e}_L^\top \mathbf{s}_R^\star$.

Leakage about the error was first considered in [14] under the name extended LWE. There, the leakage is an inner product of the error with a uniformly random binary vector. This result is applicable in our setting (when the secret distribution is a binary distribution), but unfortunately it is not clear how to generalize this result to multiple hints. Thus, we can allow only one user apart from the challenge users. A later work considered a version with multiple hints [1], where you get the inner product of the LWE error with a matrix $\mathbf{Z}$, but that matrix has a special structure that makes their result not applicable to our problem. Instead, we use a recent result [23] showing that LWE holds even when several *noisy* hints about the LWE secret are provided. Concretely, they show that the LWE holds even given $\mathbf{Z}\mathbf{e} + \mathbf{e}'$ for a short matrix $\mathbf{Z}$, the LWE error vector $\mathbf{e}$ and some new noise $\mathbf{e}'$ sampled according to a discrete Gaussian distribution. This works well with our LWE-based NIKE because we already have a noise term added to the shared keys that can be used for this result.

This allows us to show security for any number of users N, as long as this number is fixed in advance (before selecting the concrete parameters for the scheme). The drawback of this approach is that we have to increase the noise size and therefore the correctness error (if the modulus stays fixed) to support a larger number of users. Concretely, the noise in the shared key grows with $\mathcal{O}(\sqrt{N})$ and there is an inherent barrier to do better than this: Note that our reduction so far has not taken advantage of the rounding (the rounding is done only to achieve correctness). Thus, the reduction would still work when the adversary gets access to the unrounded shared keys in the reveal shared key queries. In this experiment, the adversary can query a shared key between a non-challenge user

and a challenge user once with the challenge user's secret key and once with the other user's secret key and thus learn their difference

$$\mathbf{e}_L^\top \mathbf{s}_R^\star - \mathbf{s}_L^\top \mathbf{e}_R^\star + e' - e''.$$

The adversary can do this for many users to obtain a system of noisy linear equations of the secret $(\mathbf{s}_R^\star | \mathbf{e}_R^\star)$. Importantly, unlike in LWE, in these equations all components are so small that, with overwhelming probability, these equations hold over the integers $\mathbb{Z}$ and not only in $\mathbb{Z}_q$. Such systems of equations can be solved efficiently, given only logarithmically more equations than information-theoretically necessary to determine the secret [7]. This lower bound matches our result asymptotically (up to the logarithmic factor).

Trick 2: Make use of the rounding. We next present how we can bypass the above barrier by using the rounding function in the security proof. The high-level idea for this is very simple: When an unrounded shared key is far enough away from a rounding boundary, the reduction can ensure that the rounded shared key is the same, regardless of which user's secret key is used for the computation. In this way, the reduction does not need the leakage terms in many cases. See Figure 1 for an illustration.

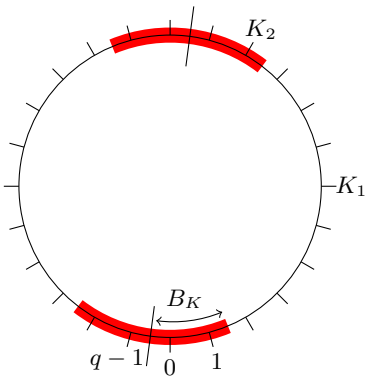

Fig. 1. Illustration of Trick 2. Here, the unrounded shared keys are rounded to one bit. The rounding boundaries are at 0 and $(q-1)/2$. B_K denotes the maximum value by which the unrounded shared keys of one pair of users can differ. When an unrounded shared key is outside the red area, such as K_1, it does not matter which user's secret key is used to compute the shared key, because the rounded result will be the same. However, if the shared key lies in the red zone, such as K_2, it can make a difference which secret key is used.

Realizing this idea is delicate because it is crucial that the leakage about the LWE secret is independent of the LWE matrix $\mathbf{A}$. Even with just one bit of

leakage about $\mathbf{s}$ which depends on $\mathbf{A}$, we could break LWE.[2] However, whether an unrounded shared key is close to a rounding boundary or not clearly depends on $\mathbf{A}$.

We exploit that the leakage about the error vectors, in contrast to the LWE secret, can depend on $\mathbf{A}$. Concretely, the result of [23] shows that for suitable parameters

$$(\mathbf{e}_1, \mathbf{Z}\mathbf{e}_1 + \mathbf{e}_2) \approx_s (\mathbf{e}' + \mathbf{f}_1, \mathbf{f}_2),$$

where $\mathbf{Z}$ is any short matrix, $\mathbf{e}'$ is a noise term with a distribution suitable for the LWE assumption, $\mathbf{e}_1$ is distributed as the noise we add to the public keys, and $\mathbf{e}_2$ is distributed as the noise we add to the unrounded shared keys. The vectors $\mathbf{f}_1$ and $\mathbf{f}_2$ are correlated, but independent of $\mathbf{e}'$. They can be efficiently sampled given $\mathbf{Z}$ (and the parameters for the noise terms). This can be used to reduce the hardness of LWE with error leakage to the hardness of plain LWE as follows: Given an LWE instance $(\mathbf{A}, \mathbf{b})$ where $\mathbf{b}$ is uniformly random or $\mathbf{b} = \mathbf{s}^\top\mathbf{A} + \mathbf{e}^\top$ where $\mathbf{e}$ has the same distribution as $\mathbf{e}'$ from the statement above. The reduction can now compute a matrix $\mathbf{Z}$ (possibly dependent on $\mathbf{A}$ and $\mathbf{b}$) and use it to sample $(\mathbf{f}_1, \mathbf{f}_2)$. The new LWE challenge is $(\mathbf{A}, \mathbf{b} + \mathbf{f}_1^\top)$ with leakage $\mathbf{f}_2$. Note that $\mathbf{b} + \mathbf{f}_1^\top$ is uniformly random if $\mathbf{b}$ was uniformly random, or an LWE sample (with higher noise) if $\mathbf{b}$ was an LWE sample.

We next explain how we use this to switch the right challenge key from real to random with less leakage. The switch for the left challenge key and shared key works analogously. For this reduction, we use a parameter k that determines the leakage sizes. If k is at least super-logarithmic in the security parameter, the number of users N can be increased by increasing the modulus linearly.

The reduction starts by sampling k error vectors $\mathbf{e}_{L,1}, \ldots, \mathbf{e}_{L,k}$ for public keys. These are used to define the leakage function for the LWE secret. This is then sent to the leakage resilient LWE challenger to get the LWE instance $(\mathbf{A}, \mathbf{b})$ with leakage about the secret $\mathbf{s}_R^\star$ (namely $\mathbf{e}_{L,i}^\top\mathbf{s}_R^\star$ for all $i \in [k]$). The public key for the right challenge user will be $\mathbf{b} + \mathbf{f}_1^\top$, but the reduction does not know $\mathbf{f}_1$ at this point, because it would have to specify the error leakage matrix $\mathbf{Z}$ before. However, $\mathbf{f}_1$ will be short, and we can use $\mathbf{b}$ as an approximation of the right challenge user's public key. The reduction then samples for all left non-challenge users the secret key $\mathbf{s}_L$ and computes $\mathbf{s}_L^\top\mathbf{b}$. If this is close to a rounding barrier, the reduction has to use the leakage to simulate a reveal query between this user and the challenge user. In this case, the reduction uses one of the pre-sampled error vectors $\mathbf{e}_{L,i}$ for this user and adds the secret $\mathbf{s}_L$ of this user (as a row vector) to the error leakage matrix $\mathbf{Z}$. If this case occurs more than k times, the reduction aborts. If $\mathbf{s}_L^\top\mathbf{b}$ is far from a rounding barrier, the reduction samples a fresh error vector $\mathbf{e}$ for this user. When this has been done for all users, the matrix $\mathbf{Z}$ is complete and the reduction can now sample $(\mathbf{f}_1, \mathbf{f}_2)$. It then uses $\mathbf{b} + \mathbf{f}_1$ as a public key for the right challenge user. The error leakage $\mathbf{f}_2$, together with the leakage

[2] The leakage bit can be the most significant bit of the first entry of $\mathbf{s}^\top\mathbf{A}$. This is very likely to agree with the most significant bit of the first entry of $\mathbf{s}^\top\mathbf{A} + \mathbf{e}^\top$ since $\mathbf{e}$ is short, but only agrees with probability $1/2$ with the most significant bit of a uniformly random vector.

about the LWE secret obtained before, can be used as in Trick 1 to simulate unrounded shared keys that are supposed to be computed with the right challenge user's secret key by using the left user's secret key instead and correcting the error that occurred by this change for a part of the left non-challenge users. For the other non-challenge users we have the guarantee that the error that occurred by this change disappears by the rounding operation.

Since the security proof here relies crucially on the rounding operation, it is important that only the output of the rounding function is used in subsequent protocols. Therefore, the key reconciliation mechanism of [21, 42], a protocol that can correct the LWE-NIKE correctness error simply by interactively sending one bit, cannot be used here directly. But we show that this protocol can, in fact, be used by using the two least significant bits of the rounding result. That is, by rounding to three bits, we can get a one-bit shared key along with the necessary auxiliary information for the interactive reconciliation procedure.

Limitations for unbound adaptive HKR security. At the end of the description of Trick 1, we showed that an adversary can learn noisy linear equations about a challenge user's secret key by taking the differences of unrounded shared keys computed with the challenge user's secret key and the other user's secret key. This attack can still be applied when the adversary gets only the rounded shared keys, but now the adversary will learn a certain interval where the inner product $(\mathbf{s}_R^\star|\mathbf{e}_R^\star)\mathbf{a}_i$ lies for several short vectors $\mathbf{a}_i$. This can be turned into an integer linear program (ILP) where $(\mathbf{s}_R^\star|\mathbf{e}_R^\star)$ is one solution. In all our results, we use parameters such that the solution space of this ILP is still exponentially large, and thus the ILP is not useful. However, if the number of users is unbounded, the adversary can obtain enough inequalities for the ILP to (likely) have a unique solution. Since ILP is an NP-complete problem, this does not immediately yield an efficient attack. But it seems that this makes proving security difficult and requires new techniques.

DKR insecurity of LWE-NIKE with NIZKPoK. We then analyze the security of the LWE NIKE in the DKR security model. We show how to turn the above attack idea with the ILP into an efficient attack with dishonest key registration queries. With dishonest key registrations, the adversary can essentially control the vectors $\mathbf{a}_i$. In particular, the adversary can make all $\mathbf{a}_i$ unit vectors, so that this $2n$ dimensional ILP can be solved by solving $2n$ 1-dimensional ILPs, which can be done efficiently.

The difficulty herewith is to ensure that the ILP has one unique solution. Essentially, we show that this attack works as long as the noise added to the shared keys is of polynomial size, capturing all (useful) parameter settings for polynomial modulus-to-noise ratio and some parameters for super-polynomial modulus-to-noise ratio.

DKR security by smudging. We show that despite the above attack, the LWE NIKE is DKR secure when the noise added to the unrounded shared keys

is super-polynomially larger than the size of the public keys and the size of the LWE secret, because it "smudges" the problematic terms $\mathbf{e}_L^\top \mathbf{s}_R^\star$ and $\mathbf{s}_L^\top \mathbf{e}_R^\star$. Thus, the distribution of the shared key is now statistically independent of which user's secret key was used, even if one key pair is adversarially generated. This is sufficient to reduce adaptive DKR security to adaptive HKR security by equipping public keys with NIZKPoKs, as in [29].

Ring LWE. All our results generalize to the ring and module LWE setting. Therefore, we present our results in the technical part with module LWE, which contains unstructured LWE (LWE in $\mathbb{Z}_q$) and ring LWE as special cases.

1.3 Roadmap

In Section 2 we recall the basic definitions and results of previous work. In Section 2.4, we also included a brief survey of papers on the leakage resilience of unstructured, ring, and module LWE. The separation of light and adaptive HKR security for NIKE with non-negligible correctness error is postponed to Appendix B. In Section 3 we prove bounded adaptive HKR security for the LWE-based NIKE with polynomial modulus-to-noise ratio. In Appendix A.1 we describe an attacker against DKR security (in the presence of NIZKPoKs) when the noise added to the unrounded shared keys is of polynomial size. In Appendix A.1 we prove DKR security with NIZKPoKs when this noise is of super-polynomial size.

2 Preliminaries

We use $\mathbb{N}_0$ for the set of natural numbers with zero and $\mathbb{N}_+$ for the set of natural numbers without zero. For strings a, b we use $a||b$ to denote the concatenation of a and b.

We use $x \xleftarrow{\$} S$ to denote the process of sampling an element x from a set S uniformly at random. For a probability distribution $\mathcal{D}$, we write $x \leftarrow \mathcal{D}$ to denote that the random variable x is distributed according to $\mathcal{D}$. For a (probabilistic) algorithm $\mathcal{A}$ we write $x \leftarrow \mathcal{A}(b)$ to denote the random variable x outputted by $\mathcal{A}$ on input b. When we want to make the random coins used for sampling explicit, we write $x \xleftarrow{\$}_r S$, $x \leftarrow_r \mathcal{D}$, or $x \leftarrow_r \mathcal{A}(b)$ where r is a string of sufficiently many (uniformly random) bits.

Definition 1 (Statistical distance). *The statistical distance between two random variables X and Y is defined as*

$$\mathsf{SD}(X, Y) := \frac{1}{2} \sum_x |\Pr[X = x] - \Pr[Y = x]|.$$

2.1 Linear algebra

Every real matrix $\mathbf{M} \in \mathbb{R}^{n \times m}$ can be written as $\mathbf{M} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$ where $\mathbf{U} \in \mathbb{R}^{n \times n}$, $\mathbf{V} \in \mathbb{R}^{m \times m}$ is a orthogonal matrix and $\mathbf{D} \in \mathbb{R}^{n \times m}$ is an upper diagonal matrix (singular value decomposition). The entries of $\mathbf{D}$ are called the singular values of $\mathbf{M}$ and we denote the smallest singular value by $\sigma_{\mathsf{min}}(\mathbf{M})$ and the largest singular value by $\sigma_{\mathsf{max}}(\mathbf{M})$. The largest singular value $\sigma_{\mathsf{max}}(\mathbf{M})$ is equal to the Euclidean spectral norm $\|\mathbf{M}\|_2 := \max_{\|\mathbf{x}\|_2=1} \|\mathbf{A}\mathbf{x}\|_2$.

If $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ is a symmetric positive-definite matrix, its singular value decomposition is of the form $\mathbf{\Sigma} = \mathbf{U}\mathbf{D}\mathbf{U}^\top$ where $\mathbf{U} \in \mathbb{R}^{n \times n}$ is an orthogonal matrix and $\mathbf{D} \in \mathbb{R}^{n \times n}$ is a diagonal matrix. Let $\sqrt{\mathbf{D}} \in \mathbb{R}^{n \times n}$ be the matrix obtained by applying the square root function component-wise to all entries of $\mathbf{D}$. With this, we define $\sqrt{\mathbf{\Sigma}} = \mathbf{U}\sqrt{\mathbf{D}}\mathbf{U}^\top$.

We will use the following bound for the largest singular value of a short matrix.

Lemma 1. *Let $\mathbf{M} \in [-B, B]^{n \times m}$ for $B > 0$. Then $\sigma_{\mathsf{max}}(\mathbf{M}) \leq B\sqrt{n}$.*

Proof.

$$\sigma_{\mathsf{max}}(\mathbf{M}) = \|\mathbf{M}\|_2 = \max_{\|\mathbf{x}\|_2=1} \|\mathbf{A}\mathbf{x}\|_2 \leq B\sqrt{n}$$

The last step uses the fact that $\mathbf{A}\mathbf{x}$ is an n-dimensional vector with entries in $[-B, B]$. □

2.2 Discrete Gaussian distribution

Let $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ be a symmetric positive-definite matrix. Then the *Gaussian function* on $\mathbb{R}^n$ is defined as $\rho_{\sqrt{\mathbf{\Sigma}}}(\mathbf{x}) := \exp(-\pi \mathbf{x}^\top \mathbf{\Sigma}^{-1} \mathbf{x})$. The function extends to sets in the usual way. That is, for any countable set $A \subseteq \mathbb{R}^n$, $\rho_{\sqrt{\mathbf{\Sigma}}}(A) := \sum_{\mathbf{x} \in A} \rho_{\sqrt{\mathbf{\Sigma}}}(\mathbf{x})$.

Moreover, for every countable set $A \subseteq \mathbb{R}^n$ and any $\mathbf{x} \in A$, the *discrete Gaussian function* is defined by $\rho_{A,\sqrt{\mathbf{\Sigma}}}(\mathbf{x}) := \frac{\rho_{\sqrt{\mathbf{\Sigma}}}(\mathbf{x})}{\rho_{\sqrt{\mathbf{\Sigma}}}(A)}$ and we denote the corresponding *discrete Gaussian distribution* as $\mathcal{D}_{A,\sqrt{\mathbf{\Sigma}}}$.

If $\mathbf{\Sigma} = \sigma^2 \cdot \mathbf{I}_n$, where $\mathbf{I}_n$ is the $n \times n$ identity matrix, we denote the Gaussian function by ρ_σ, the discrete Gaussian function by $\rho_{A,\sigma}$ and the discrete Gaussian distribution by $\mathcal{D}_{A,\sigma}$ for short.

2.3 Lattices

A lattice $\mathbf{\Lambda} \subseteq \mathbb{R}^n$ is a set of all integer linear combinations of a set of k linear independent vectors of $\mathbb{R}^n$, the basis of $\mathbf{\Lambda}$. We call k the rank of the lattice. The dual of a lattice $\mathbf{\Lambda}$ is $\mathbf{\Lambda}^* := \{\mathbf{w} \in \mathbb{R}^n \mid \forall \mathbf{v} \in \mathbf{\Lambda} : \langle \mathbf{v}, \mathbf{w} \rangle \in \mathbb{Z}\}$

In this work, we will be dealing with lattices of the form $\mathcal{R}^n$ where $\mathcal{R}$ is a ring that becomes a lattice through a fixed embedding (injective ring homomorphism[3])

[3] Here, $\mathbb{R}^d$ is a ring with component-wise addition and multiplication.

$\psi : \mathcal{R} \to \mathbb{R}^d$. We extend ψ componentwise to vectors and matrices over $\mathcal{R}$. In this work, we will always implicitly assume that rings come with such an embedding. Typically, $\mathcal{R}$ will be $\mathbb{Z}[\zeta]$ where ζ is an element of order ℓ (i.e., an ℓ-th root of unity), the ring of algebraic integers of the cyclotomic number field $\mathbb{Q}(\zeta)$. The minimal polynomial of ζ has degree $d := \phi(\ell)$ (where ϕ is Euler's totient function) and we will also call this the degree of the ring.

For this ring, we will use the canonical embedding that is defined as follows: Every number field $\mathbb{Q}(\zeta)$ of degree d has a d embedding in $\mathbb{C}$ denoted by $\sigma_i : \mathbb{Q}(\zeta) \to \mathbb{C}$, each of them sending ζ to one of the primitive d-th roots of $\mathbb{C}$. For a suitable ordering of these embeddings and suitable $s_1, s_2 \in \mathbb{N}_0$ satisfying $d = s_1 + 2s_2$ we get that the following is a ring-homorphism called the canonical embedding:

$$\begin{aligned} \sigma : \mathbb{Q}(\zeta) &\to H := \{(x_1, \ldots, x_d) \in \mathbb{R}^{s_1} \times \mathbb{C}^{s_2} \mid \forall i \in [s_2] : x_{s_1+i} = \overline{x_{s_1+s_2+i}}\} \\ x &\mapsto (\sigma_1(x), \ldots, \sigma_d(x)) \end{aligned}$$

There exists an inner product space isomorphism $\Theta : H \to \mathbb{R}^d$. With this we get the embedding $\psi := (\Theta \circ \sigma)\big|_{\mathbb{Z}[\zeta]}$.

The embedding also induces a norm on the ring elements $\mathbf{r} \in \mathcal{R}^n$: For $p \in \mathbb{N}_+ \cup \{\infty\}$ we define$\|\mathbf{r}\|_p := \|\psi(\mathbf{r})\|_p$.

To bound the norm of a product, we have to take into account the ring expansion factor.

Definition 2 (Ring expansion factor). *The ring expansion factor $\gamma_{\mathcal{R}}$ of a ring is* $\max_{a,b \in \mathcal{R}\setminus\{0\}} \frac{\|a \cdot b\|_2}{\|a\|_2 \cdot \|b\|_2}$.

When working with $\mathcal{R} = \mathbb{Z}$, the ring expansion factor is obviously $\gamma_{\mathbb{Z}} = 1$. For prime-power cyclotomics it is $\gamma_{\mathcal{R}} \leq 2d$ and for power-of-two cyclotomics it is $\gamma_{\mathcal{R}} \leq d$ [2].

For a symmetric positive-definite matrix $\mathbf{\Sigma} \in \mathbb{R}^{dn \times dn}$ we write $\mathcal{D}_{\mathcal{R}^n, \sqrt{\mathbf{\Sigma}}}$ for the distribution that samples $\mathbf{r} \in \mathcal{R}^n$ with probability $\rho_{\psi(\mathcal{R})^n, \sqrt{\mathbf{\Sigma}}}(\psi(\mathbf{r}))$.

For $q \in \mathbb{N}_+$ we write $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$.

We recall the definition of the smoothing parameter for lattices.

Definition 3 ([41]). *Let $\mathbf{\Lambda}$ be a lattice, and $\varepsilon > 0$. Then* $\eta_\varepsilon(\mathbf{\Lambda}) := \min\{s > 0 \mid \rho_{1/s}(\mathbf{\Lambda}^* \setminus \{\mathbf{0}\}) \leq \varepsilon\}$.

For an invertible matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ and a lattice $\mathbf{\Lambda} \subseteq \mathbb{R}^n$, we write $\mathbf{M} \geq \eta_\varepsilon(\mathbf{\Lambda})$ iff $1 \geq \eta_\varepsilon(\mathbf{\Lambda}\mathbf{M}^{-1})$.

We use the following tail-bound for (possibly non-spherical) discrete Gaussian distributions.

Lemma 2. *For any $k > 1$, $n \in \mathbb{N}_+$ and any symmetric positive definite matrix $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$*

$$\Pr_{\mathbf{z} \leftarrow \mathcal{D}_{\mathbb{Z}^n, \sqrt{\mathbf{\Sigma}}}}[\|\mathbf{z}\|_2 > k\sqrt{\sigma_{\mathsf{max}}(\mathbf{\Sigma})n/2\pi}] < k^n e^{n(1-k^2)/2}.$$

The proof follows the outline of the proof of [35, Lemma 4.4 (3)].

Proof. We apply [6, Lemma 1.5], that for all lattices $\mathbf{\Lambda} \in \mathbb{R}^n$ and any $c \geq 1/\sqrt{2\pi}$

$$\sum_{\mathbf{z}\in\mathbf{\Lambda}, \|\mathbf{z}\|_2 > c\sqrt{n}} \exp(-\pi\|\mathbf{z}\|_2^2) < \left(c\sqrt{2\pi e}e^{-\pi c^2}\right)^n \sum_{\mathbf{z}\in\mathbf{\Lambda}} e^{-\pi\|\mathbf{z}\|_2^2},$$

to the lattice $\mathbf{\Lambda} = \sqrt{\mathbf{\Sigma}}^{-1}\mathbb{Z}^n$ to get the following:

$$\sum_{\mathbf{z}\in\sqrt{\mathbf{\Sigma}}^{-1}\mathbb{Z}^n, \mathbf{z}^\top\mathbf{z} > c^2 n} \exp(-\pi\mathbf{z}^\top\mathbf{z}) < \left(c\sqrt{2\pi e}e^{-\pi c^2}\right)^n \sum_{\mathbf{z}\in\sqrt{\mathbf{\Sigma}}^{-1}\mathbb{Z}^n} e^{-\pi\mathbf{z}^\top\mathbf{z}}.$$

Now we substitute $\mathbf{x} := \sqrt{\mathbf{\Sigma}}\mathbf{z}$ to get

$$\sum_{\mathbf{x}\in\mathbb{Z}^n, \mathbf{x}^\top\mathbf{\Sigma}^{-1}\mathbf{x} > c^2 n} \exp(-\pi\mathbf{x}^\top\mathbf{\Sigma}^{-1}\mathbf{x}) < \left(c\sqrt{2\pi e}e^{-\pi c^2}\right)^n \sum_{\mathbf{x}\in\mathbb{Z}^n} e^{-\pi\mathbf{x}^\top\mathbf{\Sigma}^{-1}\mathbf{x}}. \quad (1)$$

Next we use $\mathbf{x}^\top\sigma_{\mathsf{max}}(\mathbf{\Sigma})^{-1}\mathbf{I}_n\mathbf{x} = \mathbf{x}^\top\sigma_{\mathsf{min}}(\mathbf{\Sigma}^{-1})\mathbf{I}_n\mathbf{x} \leq \mathbf{x}^\top\mathbf{\Sigma}^{-1}\mathbf{x}$ to get

$$\begin{aligned}\sum_{\mathbf{x}\in\mathbb{Z}^n, \mathbf{x}^\top\sigma_{\mathsf{max}}(\mathbf{\Sigma})^{-1}\mathbf{I}_n\mathbf{x} > c^2 n} \exp(-\pi\mathbf{x}^\top\mathbf{\Sigma}^{-1}\mathbf{x}) &\leq \sum_{\mathbf{x}\in\mathbb{Z}^n, \mathbf{x}^\top\mathbf{\Sigma}^{-1}\mathbf{x} > c^2 n} \exp(-\pi\mathbf{x}^\top\mathbf{\Sigma}^{-1}\mathbf{x}) \\ &\overset{(1)}{<} \left(c\sqrt{2\pi e}e^{-\pi c^2}\right)^n \sum_{\mathbf{x}\in\mathbb{Z}^n} e^{-\pi\mathbf{x}^\top\mathbf{\Sigma}^{-1}\mathbf{x}}\end{aligned}$$

which can be rearranged to

$$\sum_{\mathbf{x}\in\mathbb{Z}^n, \mathbf{x}^\top\mathbf{x} > c^2\sigma_{\mathsf{max}}(\mathbf{\Sigma})n} \rho_{\mathbb{Z}^n,\sqrt{\mathbf{\Sigma}}}(\mathbf{x}) < \left(c\sqrt{2\pi e}e^{-\pi c^2}\right)^n = (c\sqrt{2\pi})^n e^{(1/2-\pi c^2)n}.$$

The lemma follows by setting $c = k/\sqrt{2\pi}$. □

2.4 (Module) learning with errors

We will use the module learning with errors problem (M-LWE) in this work. This includes the interesting special cases of unstructured LWE and ring LWE (R-LWE).

Definition 4 (M-LWE assumption with leakage). *The $(\mathcal{R}, n, m, q, \mathcal{S}, \mathcal{E}, \mathcal{L})$-*$\mathsf{LWE}$ *assumption for a ring $\mathcal{R}$ of degree d, $n, m, q \in \mathbb{N}_+$, the secret distribution $\mathcal{S}$ on $\mathcal{R}_q^n$ and the error distribution $\mathcal{E}$ on $\mathcal{R}_q^m$ and an (efficiently decidable) set of allowed (efficiently computable) leakage functions $\mathcal{L}$ states that for every PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,*

$$\mathsf{Adv}^{\mathsf{lwe}}_{\mathcal{R},n,m,q,\mathcal{S},\mathcal{E},\mathcal{L}}(\mathcal{A}) := \Pr[\mathcal{A}_2(\mathbf{A}, \mathbf{s}^\top\mathbf{A} + \mathbf{e}^\top, \mathsf{st}, \ell) - \mathcal{A}_2(\mathbf{A}, \mathbf{z}, \mathsf{st}, \ell)]$$

is negligible in $d \cdot n$, where the probability is taken over $\mathbf{A} \leftarrow \mathcal{R}_q^{n\times m}$, $\mathbf{s} \leftarrow \mathcal{S}$, $\mathbf{e} \leftarrow \mathcal{E}$, $(\mathsf{st}, f) \leftarrow \mathcal{A}_1(1^\lambda)$, and the internal randomness of $\mathcal{A}_2$ for $\ell := f(\mathbf{s})$ if $f \in \mathcal{L}$ and $\ell := \varepsilon$ otherwise.

In this paper, we require that the error distribution is a discrete Gaussian distribution ($\mathcal{E} = \mathcal{D}_{\mathcal{R}^n,\mathbf{\Sigma}}$). To build a NIKE scheme, we also require that $\mathcal{S}$ outputs only short vectors, e.g., binary or discrete Gaussian vectors.

Unstructured LWE. The case of $d = 1$, that is, where, without loss of generality, $\mathcal{R} = \mathbb{Z}$, is called (unstructured) LWE and was introduced by [45].

For standard LWE, where $\mathcal{S}$ is the uniform distribution on $\mathbb{Z}_q^n$, Regev [45] showed that for n growing polynomially in the security parameter and $\mathcal{E} = \mathcal{D}_{\mathbb{Z}_q^n,\sigma}$ with $\sigma > \max\{2\sqrt{n}, q/2^{n^c}\}$ for a constant $c \in (0,1)$ the hardness of LWE can be reduced quantumly to hard worst-case lattice problems, namely approximating the decision version of the shortest vector problem (GapSVP) and the shortest independent vector problem (SIVP) within $\widetilde{\mathcal{O}}(n2^{n^c})$. Later works also gave classical reductions for $q \geq 2^{n/2}$ consisting of small primes [43] and polynomial moduli [14].

This gives strong indication that the LWE assumption holds for both polynomial and super-polynomial modulus(q)-to-noise(σ) ratios, while the former is preferable because it has qualitatively better reductions to lattice problems and arithmetic with polynomial moduli is more efficient than for super-polynomial moduli.

The work [3] first showed the hardness of LWE with a short secret distribution, namely a discrete Gaussian distribution, as used for the error, can be reduced to standard LWE. The work [27] then showed that the LWE with binary secret and leakage about the secret (as long as enough min-entropy remains in the secret) is implied by standard LWE in the super-polynomial modulus-to-noise ratio regime. The follow-ups [14, 40] showed that LWE with a polynomial modulus-to-noise ratio and a binary secret (and leakage in [14]) is implied by LWE with uniform secrets. Finally, the work [11] showed the hardness of LWE with "noise lossy" distributions, which include all short distributions with enough min-entropy, with a reduction to standard LWE. The result can be easily extended to capture also leakage about the secret as long as enough min-entropy remains in the secret. In particular, their result holds even with polynomial modulus-to-noise ratio. The reductions of both [27] and [11] are not dimension-preserving. The work [19] presents a framework for incorporating (noisy) linear leakage about the LWE secret (and error) into attacks on LWE.

Ring LWE. Ring LWE refers to the special case where $n = 1$. The problem was introduced in [37, 36] to improve the concrete efficiency of LWE-based schemes.

The work [12] analyzed the hardness of the search variant for ring LWE for entropic secret distributions, and again their result can easily be extended to the leakage setting. They show the hardness of search ring LWE for entropic secret distributions based on the Decisional Small Polynomial Ratio (DSPR) problem. Their result requires that the min-entropy of the secret distributions is at least $d\log(\gamma\mathsf{poly}(d)) + \omega(\log\lambda)$, where γ is the standard deviation of the Gaussian distribution used in the DSPR problem. This excludes the use of binary secrets but can be satisfied by somewhat short distributions that are sufficient for our purposes. Their result also requires a certain non-degeneracy property that is proven only for power-of-two cyclotomic rings, but they conjecture that it holds for the ring of algebraic integer over all number fields.

Unfortunately, our NIKE construction relies on the hardness of the decisional (ring) LWE problem, so this result cannot be directly applied here.[4] The result of [34], which we describe in the next paragraph, implies a hardness result for decisional entropic ring LWE.

Many hardness results for ring LWE naturally require the non-spherical discrete Gaussian error distributions [37, 36, 44, 12] and converting to spherical Gaussians comes at a price [44]. Thus in this work we present our results for possibly non-spherical discrete Gaussians.

Module LWE. Module LWE was introduced in [13, 33] to interpolate between ring and module LWE.

For module LWE, there are two hardness results for entropic secret distributions (which again can easily be restated as results about leakage resilience). The first result [34] comes in a flavor similar to [11]: They show hardness of module LWE for all short secret distributions with enough min-entropy, including binary distributions, via a reduction to module LWE that is not dimension-preserving.

The second result [8] comes in the flavor of [12]: They show hardness of the search variant (thus, this result is not directly applicable for our construction) of module LWE for secret distributions with enough min-entropy (again, the entropy requirement here excludes binary distributions but can be satisfied by somewhat short distributions) with a dimension-preserving reduction to the module NTRU.

2.5 Noisy hints

We recall (a special case of) a theorem of [23] showing that noisy hints about the LWE error can be simulated, for suitable parameters, without knowing the error (at the cost of increasing the error size).

Lemma 3 ([23, Theorem 2, simplified]). *Let $\mathcal{R}$ be a ring of degree d, m, $k \in \mathbb{N}_+$, $\mathbf{Z} \in \mathbb{R}^{k \times m}$ and $\varepsilon > 0$. Let $\mathbf{\Sigma}_0 \in \mathbb{R}^{dm \times dm}$, $\mathbf{T}_0 \in \mathbb{R}^{dk \times dk}$ and $s, t \geq 2\sqrt{2}$ such that*

$$\sqrt{\mathbf{\Sigma}_0} \geq \eta_\varepsilon(\mathcal{R}^m), \qquad \sqrt{\mathbf{T}_0} \geq \eta_\varepsilon(\mathcal{R}^k), \text{ and}$$
$$t^2 \sigma_{\mathsf{min}}(\mathbf{T}_0) \geq \frac{(s^2+1)(s^2+2)}{s^2} \sigma_{\mathsf{max}}(\mathbf{\Sigma}_0) \sigma_{\mathsf{max}}(\psi(\mathbf{Z}))^2.$$

Then, for $\mathbf{\Sigma} := (s^2+1)\mathbf{\Sigma}_0$, $\mathbf{T} := (t^2+1)\mathbf{T}_0$, and $\overline{\mathbf{\Sigma}} := s^2/4 \cdot \mathbf{\Sigma}_0$ there exists an efficiently sampable distribution $\mathcal{F}$ on $\mathcal{R}^m \times \mathcal{R}^k$ such that for $\mathbf{e}_1 \leftarrow \mathcal{D}_{\mathcal{R}^m, \sqrt{\mathbf{\Sigma}}}$, $\mathbf{e}_2 \leftarrow \mathcal{D}_{\mathcal{R}^k, \sqrt{\mathbf{T}}}$, $\mathbf{e} \leftarrow \mathcal{D}_{\mathcal{R}^m, \sqrt{\overline{\mathbf{\Sigma}}}}$, and $(\mathbf{f}_1, \mathbf{f}_2) \leftarrow \mathcal{F}$ we have

$$\mathsf{SD}((\mathbf{e}_1, \mathbf{Z}\mathbf{e}_1 + \mathbf{e}_2), (\mathbf{e} + \mathbf{f}_1, \mathbf{f}_2)) \leq 22\varepsilon$$

[4] There exist search-to-decision reductions for (ring/module) LWE, but those reductions do not preserve the secret distributions and thus cannot be applied in the leaky/entropic secret regime.

2.6 Non-interactive key exchange (NIKE)

For the purpose of this paper, we introduce the notion of an asymmetric NIKE. In contrast to a standard NIKE, this distinguishes between "left" and "right" users. A shared key can be computed only between a left and a right user. This is more convenient to work with in the LWE setting, where left and right keys are computed differently. An asymmetric NIKE implies a standard NIKE as introduced in [16] by generating for every user a left and a right key and deciding based on a canonical rule (e.g., based on the lexicographic order of their identities) which user takes the left role and which user takes the right role in the shared key computation. All security notions considered in this work carry over to their standard NIKE counterparts under this transformation.

Definition 5 (NIKE). *An asymmetric* NIKE *scheme with identity space $\mathcal{IDS}$ and shared key space $\mathcal{K}$ (with $|\mathcal{IDS}|, |\mathcal{K}| \geq 2$) consists of the following five PPT algorithms:*

- Setup *inputs the unary encoded security parameter 1^λ and samples public parameters* pp,
- $\mathsf{KeyGenL}$ *inputs the parameters* pp *and an identity $\mathsf{id}_L \in \mathcal{IDS}$ and samples a left key pair $(\mathsf{pk}_L, \mathsf{sk}_L) \in \mathcal{LPK} \times \mathcal{LSK}$,*
- $\mathsf{KeyGenR}$ *inputs the parameters* pp *and an identity $\mathsf{id}_R \in \mathcal{IDS}$ and samples a right key pair $(\mathsf{pk}_R, \mathsf{sk}_R) \in \mathcal{RPK} \times \mathcal{RSK}$,*
- $\mathsf{SharedKeyL}$ *inputs the parameters* pp*, an identity id_R with its corresponding right public key pk_R and another identity id_L with its corresponding left secret key sk_L and outputs a shared key K or the failure symbol $\perp$.*
- $\mathsf{SharedKeyR}$ *inputs the parameters* pp*, an identity id_L with its corresponding left public key pk_L and another identity id_R with its corresponding right secret key sk_R and outputs a shared key K or the failure symbol $\perp$.*

The standard definition of NIKE is obtained by requiring $\mathsf{KeyGenL} = \mathsf{KeyGenR}$ and $\mathsf{SharedKeyL} = \mathsf{SharedKeyR}$.

Definition 6 (Correctness). *We say that a NIKE* $\mathsf{NIKE} = (\mathsf{Setup}, \mathsf{KeyGenL}, \mathsf{KeyGenR}, \mathsf{SharedKeyL}, \mathsf{SharedKeyR})$ *for identity space $\mathcal{IDS}$ has correctness error $\varepsilon_{\mathsf{corr}}$, if for all $\lambda \in \mathbb{N}_+$ and all $\mathsf{id}_L, \mathsf{id}_R \in \mathcal{IDS}$ with $\mathsf{id}_L \neq \mathsf{id}_R$ for*

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$,
- $(\mathsf{pk}_L, \mathsf{sk}_L) \leftarrow \mathsf{KeyGenL}(\mathsf{pp}, \mathsf{id}_L)$,
- $(\mathsf{pk}_R, \mathsf{sk}_R) \leftarrow \mathsf{KeyGenR}(\mathsf{pp}, \mathsf{id}_R)$,
- $K_L \leftarrow \mathsf{SharedKeyL}(\mathsf{pp}, \mathsf{id}_R, \mathsf{pk}_R, \mathsf{id}_L, \mathsf{sk}_L)$*, and*
- $K_R \leftarrow \mathsf{SharedKeyR}(\mathsf{pp}, \mathsf{id}_L, \mathsf{pk}_L, \mathsf{id}_R, \mathsf{sk}_R)$

$$\Pr[K_L = K_R \neq \perp] \geq 1 - \varepsilon_{\mathsf{corr}}(\lambda),$$

where the probability is taken over the randomness used to sample pp*, $(\mathsf{pk}_L, \mathsf{sk}_L)$, $(\mathsf{pk}_R, \mathsf{sk}_R)$, K_L and K_R.*

A NIKE is statistically correct *if $\varepsilon_{\mathsf{corr}}$ is negligible and* perfectly correct *iff $\varepsilon_{\mathsf{corr}} = 0$.*

In this work, we consider three variants of security for NIKE. The weakest notion is light security. Here, the adversary can register only two users $\mathsf{id}_L^\star, \mathsf{id}_R^\star$ and use them for the challenge. This security notion captures scenarios where each public key is used only to compute a single shared key.

The next stronger notion is adaptive HKR (honest key registration) security, which allows an adversary to adaptively register an arbitrary number of users, corrupt users, and reveal shared keys between two users. For simplicity we still allow only one challenge query, but the more realistic notion with multiple challenge queries is implied by the single-challenge notion via a standard hybrid argument. This notion captures multi-user scenarios where a public key can be used to generate shared keys with many other users. However, it assumes that all users generate their public and secret key honestly.

The more realistic and stronger notion is adaptive DKR (dishonest key registration), which additionally allows the adversary to reveal shared keys between honest users and public keys of his choice.

Definition 7 (Security). *An asymmetric NIKE* NIKE *is lightly, (N–user) adaptively HKR, or DKR secure if for all PPT adversaries* $\mathcal{A}$

$$\mathsf{Adv}^{\mathcal{A},\mathsf{xxx}}_{\mathsf{NIKE},\mathcal{I}}(\lambda) := 2|\Pr[\mathsf{Exp}^{\mathsf{xxx}}_{\mathcal{A},\mathsf{NIKE}}(\lambda) \Rightarrow 1] - 1/2|$$

is negligible for $\mathsf{xxx} = \mathsf{light}$, $\mathsf{xxx} = \mathsf{N\text{-}user\text{-}adaptive\text{-}HKR}$, $\mathsf{xxx} = \mathsf{adaptive\text{-}HKR}$ *or* $\mathsf{xxx} = \mathsf{adaptive\text{-}DKR}$, *respectively. The games* $\mathsf{Exp}^{\mathsf{xxx}}_{\mathcal{A},\mathsf{NIKE}}$ *are defined in Figure 2 and Table 2.*

Table 2. The following table indicates how often the adversary is allowed to query each of the oracles in the light, (N–user) adaptive HKR, and adaptive DKR security game. The symbol "∞" stands for arbitrary many allowed queries. In all games, a $\mathcal{O}_{\mathsf{testL}}$ or a $\mathcal{O}_{\mathsf{testR}}$ query is allowed.

	$\mathcal{O}_{\mathsf{regHL}}$	$\mathcal{O}_{\mathsf{regHR}}$	$\mathcal{O}_{\mathsf{extrL}}$	$\mathcal{O}_{\mathsf{extrR}}$	$\mathcal{O}_{\mathsf{regDL}}$	$\mathcal{O}_{\mathsf{regDR}}$	$\mathcal{O}_{\mathsf{revL}}$	$\mathcal{O}_{\mathsf{revL}}$	$\mathcal{O}_{\mathsf{testL}}$	$\mathcal{O}_{\mathsf{testR}}$
light	1	1	0	0	0	0	0	0	1	
N–user adpt. HKR	N	N	∞	∞	0	0	∞	∞	1	
adaptive HKR	∞	∞	∞	∞	0	0	∞	∞	1	
adaptive DKR	∞	∞	∞	∞	∞	∞	∞	∞	1	

2.7 Pseudorandom function (PRF)

Definition 8 (PRF). *A* PRF *for domain* $\mathcal{D}$ *and range* $\mathcal{E}$ *consists of two PPT algorithms*

- $\mathsf{Gen}(1^\lambda)$ *inputs the unary encoded security parameter and outputs a PRF key* k *and*

$\underline{\mathsf{Exp}^{\mathsf{light/(N\text{-}user\text{-})adaptive\text{-}HKR/adaptive\text{-}DKR}}_{\mathcal{A},\mathsf{NIKE}}(\lambda)}$:

$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
$Q_{\mathsf{extrL}} := \emptyset;\ Q_{\mathsf{extrR}} := \emptyset$
$Q_{\mathsf{rev}} := \emptyset;\ Q_{\mathsf{test}} := \emptyset$
$\mathsf{lpks} : \mathcal{IDS} \dashrightarrow \mathcal{LPK}$
$\mathsf{rpks} : \mathcal{IDS} \dashrightarrow \mathcal{RPK}$
$\mathsf{lsks} : \mathcal{IDS} \dashrightarrow \mathcal{LSK}$
$\mathsf{rsks} : \mathcal{IDS} \dashrightarrow \mathcal{RSK}$
$b \xleftarrow{\$} \{0,1\}$
$b^\star \leftarrow \mathcal{A}(\mathsf{pp})$
if $Q_{\mathsf{rev}} \cap Q_{\mathsf{test}} = \emptyset \wedge \nexists(\mathsf{id}_1, \mathsf{id}_2) \in Q_{\mathsf{test}} : \mathsf{id}_1 \in Q_{\mathsf{extrL}} \vee \mathsf{id}_2 \in Q_{\mathsf{extrR}}$ **then**
  return $b \stackrel{?}{=} b^\star$
else
  return $b' \xleftarrow{\$} \{0,1\}$

$\underline{\mathcal{O}_{\mathsf{regHL}}(\mathsf{id}_L \in \mathcal{IDS})}$:
if $\mathsf{lpks}(\mathsf{id}_L) \neq \bot$ **then return** $\bot$
$(\mathsf{pk}_L, \mathsf{sk}_L) \leftarrow \mathsf{KeyGenL}(\mathsf{pp}, \mathsf{id}_L)$
$\mathsf{lpks}(\mathsf{id}_L) := \mathsf{pk}_L;\ \mathsf{lsks}(\mathsf{id}_L) := \mathsf{sk}_L$
return pk_L

$\underline{\mathcal{O}_{\mathsf{regHR}}(\mathsf{id}_R \in \mathcal{IDS})}$:
if $\mathsf{rpks}(\mathsf{id}_R) \neq \bot$ **then return** $\bot$
$(\mathsf{pk}_R, \mathsf{sk}_R) \leftarrow \mathsf{KeyGenR}(\mathsf{pp}, \mathsf{id}_R)$
$\mathsf{rpks}(\mathsf{id}_R) := \mathsf{pk}_R;\ \mathsf{rsks}(\mathsf{id}_R) := \mathsf{sk}_R$
return pk_R

$\underline{\mathcal{O}_{\mathsf{extrL}}(\mathsf{id}_L \in \mathcal{IDS})}$:
if $\mathsf{lsks}(\mathsf{id}_L) \neq \bot$ **then**
  $Q_{\mathsf{extrL}} := Q_{\mathsf{extrL}} \cup \{\mathsf{id}_L\}$
  return $\mathsf{lsks}(\mathsf{id}_L)$
else return $\bot$

$\underline{\mathcal{O}_{\mathsf{extrR}}(\mathsf{id}_R \in \mathcal{IDS})}$:
if $\mathsf{rsks}(\mathsf{id}_R) \neq \bot$ **then**
  $Q_{\mathsf{extrR}} := Q_{\mathsf{extrR}} \cup \{\mathsf{id}_R\}$
  return $\mathsf{rsks}(\mathsf{id}_R)$
else return $\bot$

$\underline{\mathcal{O}_{\mathsf{regDL}}(\mathsf{id}_L \in \mathcal{IDS}, \mathsf{pk}_L \in \mathcal{PK})}$:
if $\mathsf{lpks}(\mathsf{id}_L) \neq \bot$ **then return** $\bot$
$\mathsf{lpks}(\mathsf{id}_L) := \mathsf{pk}_L$
return *OK*

$\underline{\mathcal{O}_{\mathsf{regDR}}(\mathsf{id}_R \in \mathcal{IDS}, \mathsf{pk}_R \in \mathcal{PK})}$:
if $\mathsf{rpks}(\mathsf{id}_R) \neq \bot$ **then return** $\bot$
$\mathsf{rpks}(\mathsf{id}_R) := \mathsf{pk}_R$
return *OK*

$\underline{\mathcal{O}_{\mathsf{revL}}(\mathsf{id}_R \in \mathcal{IDS}, \mathsf{id}_L \in \mathcal{IDS})}$:
if $\mathsf{rpks}(\mathsf{id}_R) \neq \bot \wedge \mathsf{lsks}(\mathsf{id}_L) \neq \bot$ **then**
  $Q_{\mathsf{rev}} := Q_{\mathsf{rev}} \cup \{(\mathsf{id}_L, \mathsf{id}_R)\}$
  $\mathsf{pk}_R := \mathsf{rpks}(\mathsf{id}_R);\ \mathsf{sk}_L := \mathsf{lsks}(\mathsf{id}_L)$
  ret. $\mathsf{SharedKeyL}(\mathsf{pp}, \mathsf{id}_R, \mathsf{pk}_R, \mathsf{id}_L, \mathsf{sk}_L)$
else return $\bot$

$\underline{\mathcal{O}_{\mathsf{revR}}(\mathsf{id}_L \in \mathcal{IDS}, \mathsf{id}_R \in \mathcal{IDS})}$:
if $\mathsf{lpks}(\mathsf{id}_L) \neq \bot \wedge \mathsf{rsks}(\mathsf{id}_R) \neq \bot$ **then**
  $Q_{\mathsf{rev}} := Q_{\mathsf{rev}} \cup \{(\mathsf{id}_L, \mathsf{id}_R)\}$
  $\mathsf{pk}_L := \mathsf{lpks}(\mathsf{id}_L);\ \mathsf{sk}_R := \mathsf{rsks}(\mathsf{id}_R)$
  ret. $\mathsf{SharedKeyR}(\mathsf{pp}, \mathsf{id}_L, \mathsf{pk}_L, \mathsf{id}_R, \mathsf{sk}_R)$
else return $\bot$

$\underline{\mathcal{O}_{\mathsf{testL}}(\mathsf{id}_R^\star \in \mathcal{IDS}, \mathsf{id}_L^\star \in \mathcal{IDS})}$:
if $\mathsf{rsks}(\mathsf{id}_R^\star) \neq \bot \wedge \mathsf{lsks}(\mathsf{id}_L^\star) \neq \bot$ **then**
  $Q_{\mathsf{test}} := Q_{\mathsf{test}} \cup \{(\mathsf{id}_L^\star, \mathsf{id}_R^\star)\}$
  $\mathsf{pk}_R^\star := \mathsf{rpks}(\mathsf{id}_R^\star);\ \mathsf{sk}_L^\star := \mathsf{lsks}(\mathsf{id}_L^\star)$
  $K_0^\star \leftarrow \mathsf{SharedKeyL}(\mathsf{pp}, \mathsf{id}_R^\star, \mathsf{pk}_R^\star, \mathsf{id}_L^\star, \mathsf{sk}_L^\star)$
  $K_1^\star \xleftarrow{\$} \mathcal{K}$
  return $K_b^\star$
else return $\bot$

$\underline{\mathcal{O}_{\mathsf{testR}}(\mathsf{id}_L^\star \in \mathcal{IDS}, \mathsf{id}_R^\star \in \mathcal{IDS})}$:
if $\mathsf{lsks}(\mathsf{id}_L^\star) \neq \bot \wedge \mathsf{rsks}(\mathsf{id}_R^\star) \neq \bot$ **then**
  $Q_{\mathsf{test}} := Q_{\mathsf{test}} \cup \{(\mathsf{id}_L^\star, \mathsf{id}_R^\star)\}$
  $\mathsf{pk}_L^\star := \mathsf{lpks}(\mathsf{id}_L^\star);\ \mathsf{sk}_R^\star := \mathsf{rsks}(\mathsf{id}_R^\star)$
  $K_0^\star \leftarrow \mathsf{SharedKeyR}(\mathsf{pp}, \mathsf{id}_L^\star, \mathsf{pk}_L^\star, \mathsf{id}_R^\star, \mathsf{sk}_R^\star)$
  $K_1^\star \xleftarrow{\$} \mathcal{K}$
  return $K_b^\star$
else return $\bot$

Fig. 2. Experiment for light, N–user adaptive HKR, and adaptive DKR security of an asymmetric NIKE scheme $\mathsf{NIKE} = (\mathsf{Setup}, \mathsf{KeyGenL}, \mathsf{KeyGenR}, \mathsf{SharedKeyL}, \mathsf{SharedKeyR})$ with identity space $\mathcal{IDS}$. $\mathcal{LPK}$ and $\mathcal{RPK}$ denote the left and right public key spaces, and $\mathcal{LSK}$ and $\mathcal{RSK}$ denote the left and right secret key spaces, respectively. The adversary $\mathcal{A}$ has access to the oracles as indicated in Table 2. The partial maps lpks, rpks, lsks, and rsks are initially totally undefined.

– $\mathsf{PRF}(k, x)$ *inputs a a PRF key* k *and* $x \in \mathcal{D}$ *and outputs* $y \in \mathcal{E}$.

Definition 9 (Pseudorandomness of PRFs). *A PRF is* pseudorandom *if for every PPT adversary* $\mathcal{A}$

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathcal{A},\mathsf{PRF}}(\lambda) := |\Pr_{k \leftarrow \mathsf{Gen}(1^\lambda)}[\mathcal{A}^{\mathsf{PRF}(k,\cdot)}(1^\lambda) = 1] - \Pr_{\mathsf{RF} \xleftarrow{\$} \mathcal{E}^{\mathcal{D}}}[\mathcal{A}^{\mathsf{RF}(\cdot)}(1^\lambda) = 1]|.$$

Here, $\mathcal{E}^{\mathcal{D}}$ *denotes the set of all functions from* $\mathcal{D}$ *to* $\mathcal{E}$.

3 Security for multiple users

In this section, we present an LWE-based NIKE that achieves N–user adaptive HKR security for any polynomial number of users N. The price for a higher number of users is either an increase in the correctness error or an increase in the modulus (without increasing the absolute error size). Compared to the LWE-based NIKEs in previous works, the only difference in this construction is that we make the $\mathsf{SharedKey}$ algorithm deterministic by generating all the randomness used for the shared key generation with a PRF whose seed is stored in the user's secret key. This is necessary to avoid duplicated shared key queries for the same (ordered) pair of users. Alternatively, a user could keep a state of all already generated shared keys.

The NIKE scheme is shown in Figure 3. It requires a PRF $\mathsf{PRF} = (\mathsf{Gen}_{\mathsf{PRF}}, \mathsf{PRF})$ with domain $\mathcal{D} = \mathcal{IDS}$ and range $\{0,1\}^{\alpha+\rho}$ where α is the number of random bits required for sampling from discrete Gaussian $\mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$ and ρ determines the shared key space (see below). The LWE secret distribution $\mathcal{S}$ can be any short distribution on $\mathcal{R}^n$. Let $B_{\mathbf{s}} > 0$ be a bound on the size of $\mathcal{S}$, i.e., for all $\mathbf{s}$ in the range of $\mathcal{S}$ we have $\|\mathbf{s}\|_\infty < B_{\mathbf{s}}$. We also put the following mild requirement on the distribution $\mathcal{S}$: For all $k \in \mathbb{N}_+$ (growing polynomial in the security parameter), $\mathbf{s}, \mathbf{s}_1, \ldots, \mathbf{s}_k \leftarrow \mathcal{S}$, $\mathbf{A} \leftarrow \mathcal{R}_q^{n \times n}$, and $u_1, \ldots, u_k \leftarrow \mathcal{R}_q$ we have

$$\mathsf{SD}((\mathbf{s}^\top \mathbf{A}\mathbf{s}_1, \ldots, \mathbf{s}^\top \mathbf{A}\mathbf{s}_k), (u_1, \ldots, u_k)) \leq \varepsilon_{\mathsf{lohl}}(\lambda) \tag{2}$$

for a negligible function $\varepsilon_{\mathsf{lohl}}$. This is true for any distribution $\mathcal{S}$ with sufficient min-entropy if $\mathcal{R} = \mathbb{Z}$ by the left-over hash lemma [31] and for any cyclotomic ring $\mathcal{R}$ by a ring version of the left-over hash lemma [39],[9, Lemma 7].

In this NIKE, the shared key is obtained by rounding a ring element. When two users compute their shared key, they will get close (but rarely identical) unrounded shared keys. The purpose of the rounding procedure is to obtain a shared key that agrees with high probability, but in this work it will also play an important role in the security reduction. Let $d' \leq d$ be a parameter controlling the size of the shared key.[5] To define the rounding function (for $\beta \in \{0,1\}^\rho$), let

[5] With ring/module LWE it might be desirable to use a higher ring degree than the number of bits needed for the shared key.

$\underline{\mathsf{Setup}(1^\lambda)}$:
$\mathbf{A} \xleftarrow{\$} \mathcal{R}_q^{n\times n}$
return $\mathsf{pp} := \mathbf{A}$

$\underline{\mathsf{KeyGenL}(\mathsf{pp}, \mathsf{id}_L)}$:
$\mathbf{s}_L \leftarrow \mathcal{S};\ \mathbf{e}_L \leftarrow \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}$
$\mathsf{pk}_L := \mathbf{s}_L^\top \mathbf{A} + \mathbf{e}_L^\top$
$k_{\mathsf{PRF}} \leftarrow \mathsf{Gen}_{\mathsf{PRF}}(1^\lambda)$
$\mathsf{sk}_L := (\mathbf{s}_L, k_{\mathsf{PRF}})$
return $(\mathsf{pk}_L, \mathsf{sk}_L)$

$\underline{\mathsf{KeyGenR}(\mathsf{pp}, \mathsf{id}_R)}$:
$\mathbf{s}_R \leftarrow \mathcal{S};\ \mathbf{e}_R \leftarrow \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}$
$\mathsf{pk}_R := \mathbf{A}\mathbf{s}_R + \mathbf{e}_R$
$k_{\mathsf{PRF}} \leftarrow \mathsf{Gen}_{\mathsf{PRF}}(1^\lambda)$
$\mathsf{sk}_R := (\mathbf{s}_R, k_{\mathsf{PRF}})$
return $(\mathsf{pk}_R, \mathsf{sk}_R)$

$\underline{\mathsf{SharedKeyL}(\mathsf{pp}, \mathsf{id}_R, \mathsf{pk}_R, \mathsf{id}_L, \mathsf{sk}_L)}$:
parse $\mathsf{sk}_L =: (\mathbf{s}_L, k_{\mathsf{PRF}})$
$r||\beta := \mathsf{PRF}(k_{\mathsf{PRF}}, \mathsf{id}_R)$
$e' \leftarrow_r \mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$
return $\mathsf{Round}_\beta(\mathbf{s}_L^\top \mathsf{pk}_R + e')$

$\underline{\mathsf{SharedKeyR}(\mathsf{pp}, \mathsf{id}_L, \mathsf{pk}_L, \mathsf{id}_R, \mathsf{sk}_R)}$:
parse $\mathsf{sk}_R =: (\mathbf{s}_R, k_{\mathsf{PRF}})$
$r||\beta := \mathsf{PRF}(k_{\mathsf{PRF}}, \mathsf{id}_L)$
$e' \leftarrow_r \mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$
return $\mathsf{Round}_\beta(\mathsf{pk}_L \mathbf{s}_R + e')$

Fig. 3. The LWE based NIKE we consider in this section. It is identical to previous works except for the use of a PRF.

q' be the unique integer with $q' \in [q - 2^\rho + 1, q]$ and $2^q \mid q'$. Then we define

$$\begin{aligned}
\mathsf{Round}_\beta : \mathcal{R}_q &\to (\{0,1\}^\rho)^{d'} \\
r &\mapsto (\mathsf{round}_\beta(\mathbf{x}_1), \ldots, \mathsf{round}_\beta(\mathbf{x}_{d'})) \text{ where } \mathbf{x} = \psi(r) \\
\mathsf{round}_\beta : \{0, \ldots, q-1\} &\to \{0,1\}^\rho \\
x &\mapsto \begin{cases} x \text{ div } \frac{q'}{2^\rho} & \text{if } x < q' \\ \beta & \text{otherwise} \end{cases}
\end{aligned}$$

where the coefficients of $\mathcal{R}_q$ (under the fixed embedding ψ) are interpreted as integers in $\{0, \ldots, q\}$. This definition of the rounding function satisfies two properties that are important for the NIKE construction:

- For uniformly random input $r \in \mathcal{R}_q$ and uniformly random β, $\mathsf{Round}_\beta(r)$ is uniformly random in $(\{0,1\}^\rho)^d$.
- "Close" inputs lead as often as possible to the same output.

The latter property is analyzed more formally in the correctness theorem for this NIKE.

Theorem 1 (Correctness). *The NIKE scheme presented in Figure 3 has correctness error*

$$\begin{aligned}
\varepsilon_{\mathsf{corr}}(\lambda) \leq \frac{2^\rho(4nB_{\mathbf{s}}\sqrt{nd\sigma_{\mathsf{max}}(\mathbf{\Sigma})/\pi} + 4\sqrt{(n+1)d\sigma_{\mathsf{max}}(\mathbf{T})/2\pi} + 1)}{q} \\
+ 2\sqrt{2}^{nd} e^{-nd/2} + 2\sqrt{n+1}^{d} e^{-nd/2} + \varepsilon_{\mathsf{lohl}}(\lambda) + \mathsf{Adv}^{\mathsf{prf}}_{\mathcal{A},\mathsf{PRF}}(\lambda).
\end{aligned}$$

Proof. For this analysis we assume that the randomness used in the shared key generation algorithm is truly random and not generated with a PRF. This changes the correctness error at most by $\mathsf{Adv}^{\mathsf{prf}}_{\mathcal{A},\mathsf{PRF}}(\lambda)$.

Let $\mathbf{A} \xleftarrow{\$} \mathcal{R}_q^{n\times n}, \mathbf{s}_L, \mathbf{s}_R \leftarrow \mathcal{S}$; $\mathbf{e}_L, \mathbf{e}_R \leftarrow \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}$, and $e'_L, e'_R \leftarrow \mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$. When a user with left public key $\mathbf{s}_L^\top\mathbf{A} + \mathbf{e}_L^\top$ and a user with right public key $\mathsf{pk}_R := \mathbf{A}\mathbf{s}_R + \mathbf{e}_R$ compute the shared key between them, the unrounded keys are

$$r_L := \mathbf{s}_L^\top(\mathbf{A}\mathbf{s}_R + \mathbf{e}_R) + e'_L \quad \text{and} \quad r_R := (\mathbf{s}_L^\top\mathbf{A} + \mathbf{e}_L^\top)\mathbf{s}_R + e'_R$$

and thus the ∞-norm of their difference is

$$\begin{aligned}\|r_L - r_R\|_\infty &= \|\mathbf{s}_L^\top\mathbf{e}_R + e'_L - \mathbf{e}_L^\top\mathbf{s}_R - e'_R\|_\infty \\ &\le n \cdot \|\mathbf{s}_L\|_\infty \cdot \|\mathbf{e}_R\|_\infty + \|e'_L\|_\infty + n \cdot \|\mathbf{e}_L\|_\infty \cdot \|\mathbf{s}_R\|_\infty + \|e'_R\|_\infty. \\ &\le n \cdot \|\mathbf{s}_L\|_\infty \cdot \|\mathbf{e}_R\|_2 + \|e'_L\|_2 + n \cdot \|\mathbf{e}_L\|_2 \cdot \|\mathbf{s}_R\|_\infty + \|e'_R\|_2. \\ &\overset{(*)}{\le} 2nB_{\mathbf{s}}\sqrt{2nd\sigma_{\mathsf{max}}(\mathbf{\Sigma})/2\pi} + 2\sqrt{(n+1)d\sigma_{\mathsf{max}}(\mathbf{T})/2\pi} =: B_K.\end{aligned}$$

where the inequality marked with $(*)$ holds with probability at least $2\sqrt{2}^{nd}e^{-nd/2} + 2\sqrt{n+1}^{d}e^{-nd/2}$ (which is negligible in nd) by applying Lemma 2 with $k = \sqrt{2}$ to bound $\|\mathbf{e}_R\|_2$ and $\|\mathbf{e}_L\|_2$ and applying the same lemma with $k = \sqrt{n+1}$ to bound $\|e'_L\|_2$ and $\|e'_R\|_2$.

The round_β procedure splits the set $\{0,\dots,q-1\}$ into 2^ρ intervals and rounds two values to the same bits if they lie inside the same interval. A correctness error can only occur if the unrounded shared key of the left user is within distance B_K from an interval boundary or greater than or equal to q' (in which case the rounding function returns the random value β). Since $\mathbf{s}_L^\top\mathbf{A}\mathbf{s}_R$ and thus also the unrounded shared key is $\varepsilon_{\mathsf{lohl}}(\lambda)$ close to uniformly random by Equation (2), this can happen at most with probability $2^\rho(2B_K+1)/q + \varepsilon_{\mathsf{lohl}}(\lambda)$.

Combining these results proves the theorem. □

In the security analysis, we introduce another parameter k that controls the number of users N such that the NIKE is still N–user adaptive HKR secure. We can always have $N = k+1$ many users, but by increasing the modulus (i.e., decreasing the correctness error), we can allow more users. Increasing the parameter k also increases the leakage about the LWE secret (and thus ultimately nd) as well as the noise and thus the correctness error. For the N–user adaptive HKR secure of the NIKE we have the following requirements on the parameters:

- $\varepsilon > 0$ is negligible.
- The matrices $\mathbf{\Sigma} \in \mathbb{R}^{nd\times nd}$, $\mathbf{T} \in \mathbb{R}^{d\times d}$, and $\overline{\mathbf{\Sigma}} \in \mathbb{R}^{nd\times nd}$ meet the requirements of Lemma 3 for every hint matrix $\mathbf{Z} \in \mathcal{R}_q^{k\times n}$ with $\|\mathbf{Z}\|_\infty \le B_{\mathbf{s}}$. That is, there exist $s,t \in \mathbb{R}$ with
  - $s,t \ge 2\sqrt{2}$
  - $\sqrt{\mathbf{\Sigma}_0} \ge \eta_\varepsilon(\mathcal{R}^n)$ and $\sqrt{\mathbf{T}_0} \ge \eta_\varepsilon(\mathcal{R}^k)$
  - $t^2\sigma_{\mathsf{min}}(\mathbf{T}_0) \ge \frac{(s^2+1)(s^2+2)}{s^2}\sigma_{\mathsf{max}}(\mathbf{\Sigma}_0)B_{\mathbf{s}}^2 k$ (using Lemma 1)

– The $(\mathcal{R}, n, n+1, q, \mathcal{S}, \mathcal{D}_{\mathcal{R}_q^{n+1}, \sqrt{\widehat{\mathbf{\Sigma}}}}, \mathcal{L})$-$\mathsf{LWE}$ assumption holds for

$$\mathcal{L} = \left\{ \begin{array}{r} f : \mathcal{R}_q^n \to \mathcal{R}_q^k \\ \mathbf{s} \mapsto \mathbf{Y}\mathbf{s} \end{array} \middle| \; \mathbf{Y} \in \mathcal{R}_q^{k \times n} \right\} \quad \text{and} \quad \widehat{\mathbf{\Sigma}} = \begin{pmatrix} \overline{\mathbf{\Sigma}} & \mathbf{0} \\ \mathbf{0} & \mathbf{T} \end{pmatrix} \in \mathbb{R}^{(n+1)d \times (n+1)d}.$$

– $k \geq N - 1$ or $k \in \Omega((\mu+1) \cdot \mathsf{superlog}(\lambda))$ for a super-logarithmic function $\mathsf{superlog}$ and for

$$\mu := \frac{(N-1)2^{\rho}(4nB_{\mathbf{s}}\sqrt{\frac{nd\sigma_{\max}(\mathbf{\Sigma})}{\pi}} + 2\sqrt{\frac{(n+1)d\sigma_{\max}(\mathbf{T})}{2\pi}}) + 1)}{q} \leq N\varepsilon_{\mathsf{corr}}(\lambda)$$

Theorem 2 (Security). *The NIKE* NIKE *in Figure 3 is* N*–user adaptive HKR secure if all the requirements mentioned for the above parameters are satisfied. More precisely, for every PPT adversary* $\mathcal{A}$ *against the* N*–user adaptive HKR security of* NIKE *there exist PPT adversaries* $\mathcal{B}$*,* $\mathcal{C}$ *(with approximately the same run time) such that*

$$\mathsf{Adv}^{\mathsf{N\text{-}user\text{-}adaptive\text{-}HKR}}_{\mathcal{A},\mathsf{NIKE}}(\lambda) \leq 4N^2 \left(\mathsf{Adv}^{\mathsf{lwe}}_{\mathcal{R},n,n+1,q,\mathcal{S},\mathcal{D}_{\mathcal{R}_q^{n+1}, \sqrt{\widehat{\mathbf{\Sigma}}}},\mathcal{L}}(\mathcal{B}) + \mathsf{Adv}^{\mathsf{prf}}_{\mathcal{C},\mathsf{PRF}}(\lambda) + 22\varepsilon + \begin{cases} 0 & \text{if } k \geq N-1 \\ (*) & \text{otherwise} \end{cases} \right)$$

$$(*) = N(2\sqrt{2}^{nd}e^{-nd/2} + 2\sqrt{n+1}^{d}e^{-nd/2}) + \varepsilon_{\mathsf{lohl}}(\lambda) + \exp(-\frac{k^2/\mu - 2k + 1/\mu}{1 + k/\mu})$$

Proof. We prove security in a weaker model where the adversary has to specify the index of the $\mathcal{O}_{\mathsf{regHL}}$ and the $\mathcal{O}_{\mathsf{regHR}}$ before making any queries and otherwise proceeds as the N–user adaptive HKR security game. Security in this model implies N–user adaptive HKR security via a standard guessing argument with a security loss of N^2.

We also assume that the adversary makes a $\mathcal{O}_{\mathsf{testL}}$ query and not a $\mathcal{O}_{\mathsf{testR}}$ query. By switching "left" and "right" in the proof, we can get a proof that works for a $\mathcal{O}_{\mathsf{testR}}$ query, and by guessing which of the two cases occurs, we get a proof for the real N–user adaptive HKR security with loss 2.

The proof proceeds via a hybrid argument with the following games:

– G_0 is the real N–user adaptive HKR security game with the restrictions mentioned above.
– G_1 proceeds as G_0, but for both challenge users a truly random function (computed on the fly) is used instead of the PRF to generate the randomness in the $\mathsf{SharedKeyL}$ and $\mathsf{SharedKeyR}$ algorithm. We will use RF_L for the random function replacing the PRF with the left challenge user's secret key and RF_R for the random function replacing the PRF with the right challenge user's secret key. In the following, we will use $\overline{\mathsf{RF}}_L(\mathsf{id})$ and $\overline{\mathsf{RF}}_R(\mathsf{id})$ to denote the first part of the output of the random function that is used to generate the

random coins for the discrete Gaussian sampling algorithm, and $\underline{\mathsf{RF}}_L(\mathsf{id})$ and $\underline{\mathsf{RF}}_R(\mathsf{id})$ for the second part that is used for the rounding function. The differences from game G_0 are also shown in Figure 4.

- G_2 changes conceptually how the unrounded shared keys in $\mathcal{O}_{\mathsf{revR}}$ queries with the right challenge user are computed: Let $\mathsf{pk}_R^\star := \mathbf{A}\mathbf{s}_R^\star + \mathbf{e}_R^\star$ be the right challenge user's public key. In an $\mathcal{O}_{\mathsf{revR}}$ query with the right challenge user and a left user id with public key $\mathsf{pk}_L = \mathbf{s}_L^\top \mathbf{A} + \mathbf{e}_L^\top$ and $e' \leftarrow_{\overline{\mathsf{RF}}_R(\mathsf{id})} \mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$ the shared key is computed as

$$\mathbf{s}_L^\top \mathsf{pk}_R^\star - \mathbf{s}_L^\top \mathbf{e}_R^\star + \mathbf{e}_L^\top \mathbf{s}_R^\star + e'.$$

  For conceptual simplicity, the reduction here stores the error vectors for each user. The differences from game G_1 are also shown in Figure 5.
- In G_3 the right challenge user's public key $\mathsf{pk}_R^\star$ is replaced by a uniformly random value. In that game there is still a secret key and an error vector sampled for the right challenge user to answer $\mathcal{O}_{\mathsf{revR}}$ as in G_2, but this secret key and error vector are independent of the public key. The differences from game G_2 are also shown in Figure 6.
- G_4 makes changes analogous to G_2 to the computation of the unrounded shared keys in $\mathcal{O}_{\mathsf{revL}}$ queries with the left challenge user. Let $\mathsf{pk}_L^\star := (\mathbf{s}_L^\star)^\top \mathbf{A} + (\mathbf{e}_L^\star)^\top$ be the left challenge user's public key. In a $\mathcal{O}_{\mathsf{revL}}$ query with the left challenge user and a right user id with public key $\mathsf{pk}_R = \mathbf{A}\mathbf{s}_R + \mathbf{e}_R$ and $e' \leftarrow_{\overline{\mathsf{RF}}_L(\mathsf{id})} \mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$ the shared key is computed as

$$\mathsf{pk}_L^\star \mathbf{s}_R - (\mathbf{e}_L^\star)^\top \mathbf{s}_R + (\mathbf{s}_L^\star)^\top \mathbf{e}_R + e'.$$

  The differences from game G_3 are also shown in Figure 7.
- In G_5 the left challenge user's public key is uniformly random. Similarly to G_3, there is still an independent secret key and error vector generated to answer $\mathcal{O}_{\mathsf{revL}}$ queries as in G_4. Furthermore, in this hybrid, the $\mathcal{O}_{\mathsf{testL}}$ query will always be answered with a uniform random key. Differences from game G_4 are also shown in Figure 8.

Lemma 4 ($\mathsf{G}_0 \rightsquigarrow \mathsf{G}_1$). *For every PPT adversary $\mathcal{A}$ there exists a PPT adversary $\mathcal{C}$ with*

$$\left|\Pr[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1]\right| \leq 2\mathsf{Adv}^{\mathsf{prf}}_{\mathcal{C},\mathsf{PRF}}(\lambda)$$

Proof. Since the left and right challenge user may not be corrupted, their PRF key is never exposed. Thus, an adversary who is able to distinguish G_0 and G_1 can be reduced straightforwardly to an adversary breaking at least one of the two instances of the PRF security game. □

Lemma 5 ($\mathsf{G}_1 \rightsquigarrow \mathsf{G}_2$).

$$\Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1] = \Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1]$$

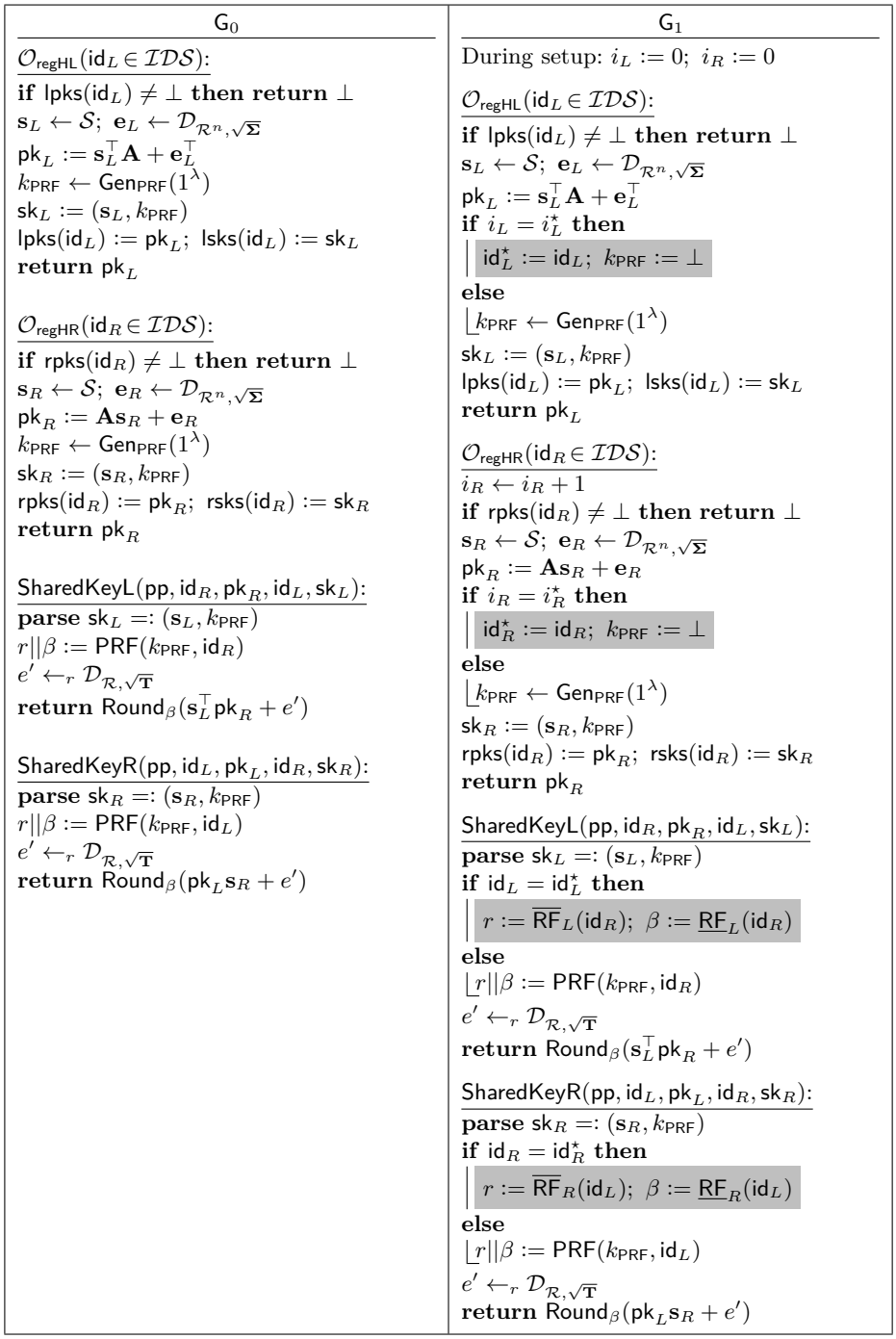

Fig. 4. Difference between the games G_0 and G_1. Changes are highlighted in gray. The variables $i_L^\star$ and $i_R^\star$ denote the index of the left and right challenge users, respectively, which the adversary has to specify at the beginning. The $\mathsf{SharedKeyL}$ algorithm is used in $\mathcal{O}_{\mathsf{revL}}$ and $\mathcal{O}_{\mathsf{testL}}$ oracles, and the $\mathsf{SharedKeyR}$ algorithm is used in $\mathcal{O}_{\mathsf{revR}}$ and $\mathcal{O}_{\mathsf{testR}}$ oracles. The oracles $\mathcal{O}_{\mathsf{extrL}}$ and $\mathcal{O}_{\mathsf{extrR}}$ are identical in both games.

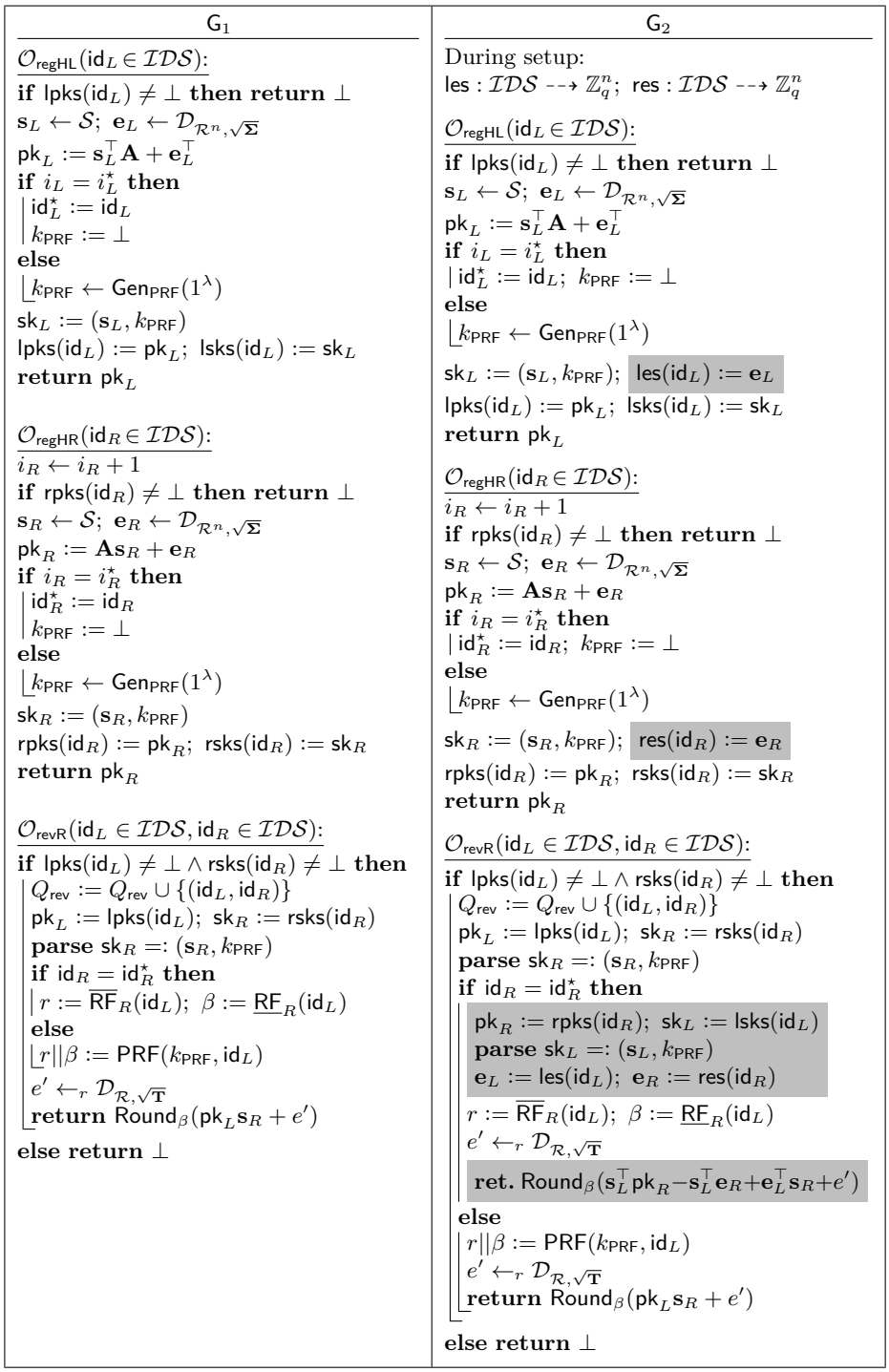

Fig. 5. Difference between the games G_1 and G_2. Changes are highlighted in gray. Oracles that are identical in both games are omitted.

G_2	G_3
$\underline{\mathcal{O}_{\mathsf{regHR}}(\mathsf{id}_R \in \mathcal{IDS})}$:	$\underline{\mathcal{O}_{\mathsf{regHR}}(\mathsf{id}_R \in \mathcal{IDS})}$:
$i_R \leftarrow i_R + 1$	$i_R \leftarrow i_R + 1$
if $\mathsf{rpks}(\mathsf{id}_R) \neq \bot$ **then return** $\bot$	**if** $\mathsf{rpks}(\mathsf{id}_R) \neq \bot$ **then return** $\bot$
$\mathbf{s}_R \leftarrow \mathcal{S};\ \mathbf{e}_R \leftarrow \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}$	$\mathbf{s}_R \leftarrow \mathcal{S};\ \mathbf{e}_R \leftarrow \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}$
$\mathsf{pk}_R := \mathbf{A}\mathbf{s}_R + \mathbf{e}_R$	**if** $i_R = i_R^\star$ **then**
if $i_R = i_R^\star$ **then**	$\quad \mathsf{pk}_R \xleftarrow{\$} \mathbb{Z}_q^n$
$\quad \mathsf{id}_R^\star := \mathsf{id}_R$	$\quad \mathsf{id}_R^\star := \mathsf{id}_R$
$\quad k_{\mathsf{PRF}} := \bot$	$\quad k_{\mathsf{PRF}} := \bot$
else	**else**
$\quad k_{\mathsf{PRF}} \leftarrow \mathsf{Gen}_{\mathsf{PRF}}(1^\lambda)$	$\quad \mathsf{pk}_R := \mathbf{A}\mathbf{s}_R + \mathbf{e}_R$
$\mathsf{sk}_R := (\mathbf{s}_R, k_{\mathsf{PRF}});\ \mathsf{res}(\mathsf{id}_R) := \mathbf{e}_R$	$\quad k_{\mathsf{PRF}} \leftarrow \mathsf{Gen}_{\mathsf{PRF}}(1^\lambda)$
$\mathsf{rpks}(\mathsf{id}_R) := \mathsf{pk}_R;\ \mathsf{rsks}(\mathsf{id}_R) := \mathsf{sk}_R$	$\mathsf{sk}_R := (\mathbf{s}_R, k_{\mathsf{PRF}});\ \mathsf{res}(\mathsf{id}_R) := \mathbf{e}_R$
return pk_R	$\mathsf{rpks}(\mathsf{id}_R) := \mathsf{pk}_R;\ \mathsf{rsks}(\mathsf{id}_R) := \mathsf{sk}_R$
	return pk_R

Fig. 6. Difference between the games G_2 and G_3. Changes are highlighted in gray. Oracles that are identical in both games are omitted.

G_3	G_4
$\underline{\mathcal{O}_{\mathsf{revL}}(\mathsf{id}_R \in \mathcal{IDS}, \mathsf{id}_L \in \mathcal{IDS})}$:	$\underline{\mathcal{O}_{\mathsf{revL}}(\mathsf{id}_R \in \mathcal{IDS}, \mathsf{id}_L \in \mathcal{IDS})}$:
if $\mathsf{rpks}(\mathsf{id}_R) \neq \bot \wedge \mathsf{lsks}(\mathsf{id}_L) \neq \bot$ **then**	**if** $\mathsf{rpks}(\mathsf{id}_R) \neq \bot \wedge \mathsf{lsks}(\mathsf{id}_L) \neq \bot$ **then**
$\quad Q_{\mathsf{rev}} := Q_{\mathsf{rev}} \cup \{(\mathsf{id}_L, \mathsf{id}_R)\}$	$\quad Q_{\mathsf{rev}} := Q_{\mathsf{rev}} \cup \{(\mathsf{id}_L, \mathsf{id}_R)\}$
$\quad \mathsf{pk}_R := \mathsf{rpks}(\mathsf{id}_R);\ \mathsf{sk}_L := \mathsf{lsks}(\mathsf{id}_L)$	$\quad \mathsf{pk}_R := \mathsf{rpks}(\mathsf{id}_R);\ \mathsf{sk}_L := \mathsf{lsks}(\mathsf{id}_L)$
$\quad$ **parse** $\mathsf{sk}_L =: (\mathbf{s}_L, k_{\mathsf{PRF}})$	$\quad$ **parse** $\mathsf{sk}_L =: (\mathbf{s}_L, k_{\mathsf{PRF}})$
$\quad$ **if** $\mathsf{id}_L = \mathsf{id}_L^\star$ **then**	$\quad$ **if** $\mathsf{id}_L = \mathsf{id}_L^\star$ **then**
$\quad\quad r := \overline{\mathsf{RF}}_L(\mathsf{id}_R);\ \beta := \underline{\mathsf{RF}}_L(\mathsf{id}_R)$	$\quad\quad \mathsf{pk}_L := \mathsf{lpks}(\mathsf{id}_L);\ \mathsf{sk}_R := \mathsf{rsks}(\mathsf{id}_R)$
$\quad$ **else**	$\quad\quad$ **parse** $\mathsf{sk}_R =: (\mathbf{s}_R, k_{\mathsf{PRF}})$
$\quad\quad r \| \beta := \mathsf{PRF}(k_{\mathsf{PRF}}, \mathsf{id}_R)$	$\quad\quad \mathbf{e}_L := \mathsf{les}(\mathsf{id}_L);\ \mathbf{e}_R := \mathsf{res}(\mathsf{id}_R)$
$\quad e' \leftarrow_r \mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$	$\quad\quad r := \overline{\mathsf{RF}}_L(\mathsf{id}_R);\ \beta := \underline{\mathsf{RF}}_L(\mathsf{id}_R)$
$\quad$ **return** $\mathsf{Round}_\beta(\mathbf{s}_L^\top \mathsf{pk}_R + e')$	$\quad\quad e' \leftarrow_r \mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$
else return $\bot$	$\quad\quad$ **ret.** $\mathsf{Round}_\beta(\mathsf{pk}_L \mathbf{s}_R - \mathbf{e}_L^\top \mathbf{s}_R + \mathbf{s}_L^\top \mathbf{e}_R + e')$
	$\quad$ **else**
	$\quad\quad r \| \beta := \mathsf{PRF}(k_{\mathsf{PRF}}, \mathsf{id}_R)$
	$\quad\quad e' \leftarrow_r \mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$
	$\quad\quad$ **return** $\mathsf{Round}_\beta(\mathbf{s}_L^\top \mathsf{pk}_R + e')$
	else return $\bot$

Fig. 7. Difference between the games G_3 and G_4. Changes are highlighted in gray. Oracles that are identical in both games are omitted.

G_4	G_5
$\underline{\mathcal{O}_{\mathsf{regHL}}(\mathsf{id}_L \in \mathcal{IDS})}$:	$\underline{\mathcal{O}_{\mathsf{regHL}}(\mathsf{id}_L \in \mathcal{IDS})}$:
if $\mathsf{lpks}(\mathsf{id}_L) \neq \bot$ **then return** $\bot$	**if** $\mathsf{lpks}(\mathsf{id}_L) \neq \bot$ **then return** $\bot$
$\mathbf{s}_L \leftarrow \mathcal{S};\ \mathbf{e}_L \leftarrow \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}$	$\mathbf{s}_L \leftarrow \mathcal{S};\ \mathbf{e}_L \leftarrow \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}$
$\mathsf{pk}_L := \mathbf{s}_L^\top \mathbf{A} + \mathbf{e}_L^\top$	**if** $i_L = i_L^\star$ **then**
if $i_L = i_L^\star$ **then**	$\quad \mathsf{pk}_L \leftarrow \mathbb{Z}_q^n$
$\quad \mathsf{id}_L^\star := \mathsf{id}_L;\ k_{\mathsf{PRF}} := \bot$	$\quad \mathsf{id}_L^\star := \mathsf{id}_L;\ k_{\mathsf{PRF}} := \bot$
else	**else**
$\quad k_{\mathsf{PRF}} \leftarrow \mathsf{Gen}_{\mathsf{PRF}}(1^\lambda)$	$\quad \mathsf{pk}_L := \mathbf{s}_L^\top \mathbf{A} + \mathbf{e}_L^\top$
$\mathsf{sk}_L := (\mathbf{s}_L, k_{\mathsf{PRF}})$	$\quad k_{\mathsf{PRF}} \leftarrow \mathsf{Gen}_{\mathsf{PRF}}(1^\lambda)$
$\mathsf{lpks}(\mathsf{id}_L) := \mathsf{pk}_L;\ \mathsf{lsks}(\mathsf{id}_L) := \mathsf{sk}_L$	$\mathsf{sk}_L := (\mathbf{s}_L, k_{\mathsf{PRF}})$
$\mathsf{les}(\mathsf{id}_L) := \mathbf{e}_L$	$\mathsf{lpks}(\mathsf{id}_L) := \mathsf{pk}_L;\ \mathsf{lsks}(\mathsf{id}_L) := \mathsf{sk}_L$
return pk_L	$\mathsf{les}(\mathsf{id}_L) := \mathbf{e}_L$
	return pk_L
$\underline{\mathcal{O}_{\mathsf{testL}}()}$:	
if $\mathsf{rsks}(\mathsf{id}_R^\star) \neq \bot \wedge \mathsf{lsks}(\mathsf{id}_L^\star) \neq \bot$ **then**	$\underline{\mathcal{O}_{\mathsf{testL}}()}$:
$\quad Q_{\mathsf{test}} := Q_{\mathsf{test}} \cup \{(\mathsf{id}_L^\star, \mathsf{id}_R^\star)\}$	**if** $\mathsf{rsks}(\mathsf{id}_R^\star) \neq \bot \wedge \mathsf{lsks}(\mathsf{id}_L^\star) \neq \bot$ **then**
$\quad \mathsf{pk}_R^\star := \mathsf{rpks}(\mathsf{id}_R^\star);\ \mathsf{sk}_L^\star := \mathsf{lsks}(\mathsf{id}_L^\star)$	$\quad Q_{\mathsf{test}} := Q_{\mathsf{test}} \cup \{(\mathsf{id}_L^\star, \mathsf{id}_R^\star)\}$
$\quad$ **parse** $\mathsf{sk}_L^\star =: (\mathbf{s}_L, k_{\mathsf{PRF}})$	$\quad K_0^\star \xleftarrow{\$} \mathbb{Z}_q$
$\quad r := \overline{\mathsf{RF}}_L(\mathsf{id}_R^\star);\ \beta := \underline{\mathsf{RF}}_L(\mathsf{id}_R^\star)$	$\quad K_1^\star \xleftarrow{\$} \mathbb{Z}_q$
$\quad e' \leftarrow_r \mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$	$\quad$ **return** $K_b^\star$
$\quad K_0^\star := \mathsf{Round}_\beta(\mathbf{s}_L^\top \mathsf{pk}_R^\star + e')$	**else return** $\bot$
$\quad K_1^\star \xleftarrow{\$} \mathbb{Z}_q$	
$\quad$ **return** $K_b^\star$	
else return $\bot$	

Fig. 8. Difference between the games G_4 and G_5. Changes are highlighted in gray. Oracles that are identical in both games are omitted.

Proof. The games G_1 and G_2 are identical. Let $\mathsf{pk}^\star_R := \mathbf{A}\mathbf{s}^\star_R + \mathbf{e}^\star_R$ be the right challenge user's public key. In a $\mathcal{O}_{\mathsf{revR}}$ query with a left user id with public key $\mathsf{pk}_L = \mathbf{s}_L^\top \mathbf{A} + \mathbf{e}_L^\top$ and $e' \leftarrow_{\overline{\mathsf{RF}}_R(\mathsf{id})} \mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$ the shared key is computed in G_2 as

$$\begin{aligned}\mathbf{s}_L^\top \mathsf{pk}^\star_R - \mathbf{s}_L^\top \mathbf{e}^\star_R + \mathbf{e}_L^\top \mathbf{s}^\star_R + e' &= \mathbf{s}_L^\top(\mathbf{A}\mathbf{s}^\star_R + \mathbf{e}^\star_R) - \mathbf{s}_L^\top \mathbf{e}^\star_R + \mathbf{e}_L^\top \mathbf{s}^\star_R + e' \\ &= \mathbf{s}_L^\top \mathbf{A}\mathbf{s}^\star_R + \mathbf{e}_L^\top \mathbf{s}^\star_R + e' = \mathsf{pk}_L \mathbf{s}^\star_R + e'\end{aligned}$$

which matches how this shared key is computed in G_1. □

Lemma 6 ($\mathsf{G}_2 \rightsquigarrow \mathsf{G}_3$)**.** *For every PPT adversary $\mathcal{A}$ there exists a PPT adversary $\mathcal{B}$ with*

$$\begin{aligned}\left|\Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1]\right| &\leq \mathsf{Adv}^{\mathsf{lwe}}_{\mathcal{R},n,n,q,\mathcal{S},\mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}},\mathcal{L}}(\mathcal{B}) + 22\varepsilon \\ &\quad + \begin{cases} 0 & \text{if } k \geq N-1 \\ (*) & \text{otherwise} \end{cases}\end{aligned}$$

$$(*) = N(2\sqrt{2}^{nd} e^{-nd/2} + 2\sqrt{n+1}^{d} e^{-nd/2}) + \varepsilon_{\mathsf{lohl}}(\lambda) + \exp(-\frac{k^2/\mu - 2k + 1/\mu}{1 + k/\mu})$$

Proof. We build a reduction for this game transition as follows: the reduction samples in the beginning $\mathbf{e}_{L,1}, \ldots, \mathbf{e}_{L,k} \leftarrow \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}$ and defines $f(\mathbf{s}) := (\mathbf{e}_{L,1}| \cdots |\mathbf{e}_{L,k})^\top \mathbf{s}$. It then sends f to the $(\mathcal{R}, n, n, q, \mathcal{S}, \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}, \mathcal{L})$-$\mathsf{LWE}$ challenger to get back $(\mathbf{A}, \mathbf{b}, \ell)$ where $\mathbf{b} = (\mathbf{s}^\star_R)^\top \mathbf{A} + (\mathbf{e}^\star_R)^\top$ for $\mathbf{s}^\star_R \leftarrow \mathcal{S}$ and $\mathbf{e}^\star_R \leftarrow \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}$ or $\mathbf{b} \xleftarrow{\$} \mathcal{R}_q^n$ and $\ell = f(\mathbf{s}^\star_R)$.

The reduction sends $\mathsf{pp} := \mathbf{A}^\top$ to the adversary. The reduction then pregenerates $N-1$ public/secret key pairs for the left non-challenge user as follows. Initially, it sets $i := 1$. Now it samples $N-1$ times $\mathbf{s}_L \leftarrow \mathcal{S}$.

Case A: If $\mathbf{b} \cdot \mathbf{s}_L$ is within distance

$$2nB_{\mathbf{s}}\sqrt{2nd\sigma_{\mathsf{max}}(\mathbf{\Sigma})/2\pi} + \sqrt{(n+1)d\sigma_{\mathsf{max}}(\mathbf{T})/2\pi} =: B_{\mathsf{dz}}$$

of an interval boundary of the rounding function or larger than or equal to q' (we will refer to this as the danger zone), the reduction sets $\mathbf{s}_{L,i} := \mathbf{s}_L$ and stores for this user as public key $\mathsf{pk}_{L,i} := \mathbf{s}_{L,i}^\top \mathbf{A} + \mathbf{e}_{L,i}^\top$ and $\mathbf{s}_{L,i}$ as secret key $i \leq k$. If $i > k$, the reduction aborts. Each time case A has occurred, the reduction increments finally i by 1.

Case B: If $\mathbf{b} \cdot \mathbf{s}_L$ is outside the danger zone, the reduction proceeds normally to generate the key, that is, it samples $\mathbf{e}_L \leftarrow \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}$ and stores $\mathbf{s}_L^\top \mathbf{A} + \mathbf{e}_L^\top$ as public key and $\mathbf{s}_L$ as secret key.

For all left users with a case B public key, no correctness error with the right challenge user will occur (with overwhelming probability). For those users with a case A public key, a correctness error might occur, but the reduction can detect this knowing $\mathbf{s}_L^\top \mathbf{e}^\star_R$ and $\mathbf{e}_L^\top \mathbf{s}^\star_R$. The latter term is known to the reduction by the leakage. If $k \geq N-1$, the reduction can pick case A for all key pairs for simplicity. We next describe the steps the reduction takes to also learn the former term.

Let $\mathbf{Z} := (\mathbf{s}_{L,1}|\cdots|\mathbf{s}_{L,k})^\top$ (if less than k left user have a case A public key, set $\mathbf{s}_{L,i} = \mathbf{0}$ for the remaining indices $i \leq k$). Now let $\mathcal{F}$ be the distribution from Lemma 3 when using this lemma with $\sqrt{\mathbf{\Sigma}}$ for the distribution of $\mathbf{e}_1$, $\sqrt{\mathbf{T}}$ for the distribution of $\mathbf{e}_2$ and $\sqrt{\overline{\mathbf{\Sigma}}}$ for the distribution of $\mathbf{e}$ and $\mathbf{Z}$ as the hint matrix. The reduction then samples $(\mathbf{f}_1, \mathbf{f}_2) \leftarrow \mathcal{F}$.

In the registration query for the right challenge user, the reduction returns $\mathbf{b}^\top + \mathbf{f}_1$ as public key. For each left user registration query, except for the left challenge user, the reduction returns one of the pre-generated left public keys. The registration queries for the right non-challenge user and the left challenge user, as well as all $\mathcal{O}_{\mathsf{extrL}}$ and $\mathcal{O}_{\mathsf{extrR}}$ queries are answered honestly.

When the reduction performs a $\mathcal{O}_{\mathsf{revR}}$ with the right challenge user and a left user with a case B public/secret key, the reduction returns

$$\mathsf{Round}_{\mathbf{0}}(\mathbf{s}_L^\top \mathsf{pk}_R^\star).$$

where $\mathbf{s}_L^\top$ is the left user's secret key. When the left user id_L is the i-th user with a case A public/secret key, the reduction returns

$$\mathsf{Round}_{\overline{\mathsf{RF}}_R(\mathsf{id}_L)}(\mathbf{s}_L^\top \mathsf{pk}_R^\star - (\mathbf{f}_2)_i + \ell_i).$$

All other $\mathcal{O}_{\mathsf{revR}}$, all $\mathcal{O}_{\mathsf{revL}}$, and the $\mathcal{O}_{\mathsf{testL}}$ query do not need the right challenge user's secret key and are answered honestly by the reduction.

Analysis. Left users with a case A public/secret key pair have an unrounded shared key with the right challenge user outside of the danger zone, i.e., a correctness error can occur at most with the negligible probability $2\sqrt{2}^{nd} e^{-nd/2} + 2\sqrt{n+1}^{d} e^{-nd/2}$ and thus, with overwhelming probability, the shared key computed by the reduction is identical to the shared key as computed in G_2 or G_3. For left users with a case B public/secret key, observe that $(\mathbf{f}_2)_i = \mathbf{s}_L^\top(\mathbf{e}_R^\star + \mathbf{f}_1) - e'$ is distributed as $\mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$ with probability at least $1 - 22\varepsilon$ by Lemma 3 and $\ell_i = \mathbf{e}_L^\top \mathbf{s}_R^\star$. Thus, when the reduction does not abort, it simulates the game G_2 if it receives a real LWE challenge and G_3 if it receives a random LWE challenge.

If $k \geq N-1$, the reduction never aborts. Otherwise, we use that the probability that a left user has an unrounded shared key in the danger zone with the right challenge user is at most $p := 2^\rho(2B_{\mathsf{dz}} + 1)q$ and this is independent for each left user except with probability $\varepsilon_{\mathsf{lohl}}(\lambda)$ by Equation (2). Let X be the random variable denoting the number of case A left users. Then the expected value of X is $\mu := (N-1)/p$. With the Chernoff bound for Bernoulli distributions, we get the following bound on the probability that X exceeds k:

$$\Pr[X \geq k] \leq \exp(-\frac{(k/\mu - 1)^2 \mu}{1 + k/\mu}) = \exp(-\frac{k^2/\mu - 2k + 1/\mu}{1 + k/\mu}) \tag{3}$$

which is negligible if $k \in \Omega((\mu+1) \cdot \mathsf{superlog}(\lambda))$ for a super-logarithmic function $\mathsf{superlog}$. □

Lemma 7 ($\mathsf{G}_3 \rightsquigarrow \mathsf{G}_4$)**.**

$$\Pr[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1] = \Pr[\mathsf{G}_4^{\mathcal{A}} \Rightarrow 1]$$

The proof is is analogous to Lemma 5, just with the roles of left and right swapped.

Lemma 8 ($\mathsf{G}_4 \rightsquigarrow \mathsf{G}_5$)**.** *For every PPT adversary $\mathcal{A}$ there exists a PPT adversary $\mathcal{B}$ with*

$$\left|\Pr[\mathsf{G}_4^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{G}_5^{\mathcal{A}} \Rightarrow 1]\right| \leq \mathsf{Adv}^{\mathsf{lwe}}_{\mathcal{R},n,n+1,q,\mathcal{S},\mathcal{D}_{\mathcal{R}_q^{n+1},\sqrt{\widehat{\mathbf{\Sigma}}}},\mathcal{L}}(\mathcal{A}) + 22\varepsilon + \begin{cases} 0 & \text{if } k \geq N-1 \\ (*) & \text{otherwise} \end{cases}$$

$$(*) = N(2\sqrt{2}^{nd}e^{-nd/2} + 2\sqrt{n+1}^{d}e^{-nd/2}) + \varepsilon_{\mathsf{lohl}}(\lambda) + \exp(-\frac{k^2/\mu - 2k + 1/\mu}{1 + k/\mu})$$

The reduction for this transition is in large parts similar to the proof for Lemma 6. Here, we describe only the differences in detail.

Proof. The reduction sends a function f distributed as in Lemma 6 to the $(\mathcal{R}, n, n+1, q, \mathcal{S}, \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}, \mathcal{L})$-$\mathsf{LWE}$ challenger to get back $(\mathbf{A}, \mathbf{b}, \ell)$ where $\mathbf{b} = (\mathbf{s}_L^\star)^\top \mathbf{A} + (\mathbf{e}_L^\star)^\top$ for $\mathbf{s}_L^\star \leftarrow \mathcal{S}$ and $\mathbf{e}_L^\star \leftarrow \mathcal{D}_{\mathcal{R}^n,\sqrt{\mathbf{\Sigma}}}$ or $\mathbf{b} \xleftarrow{\$} \mathcal{R}_q^n$ and $\ell = f(\mathbf{s}_L^\star)$. Let $\overline{\mathbf{A}} \in \mathcal{R}_q^{n\times n}$ be the top $n \times n$ matrix of $\mathbf{A}$ and let $\underline{\mathbf{A}} \in \mathcal{R}_q^{1\times n}$ be the bottom row of $\mathbf{A}$. Similarly, let $\overline{\mathbf{b}} \in \mathcal{R}_q^n$ be the top n entries of $\mathbf{b}$ and $\underline{\mathbf{b}} \in \mathcal{R}_q$ the last entry of $\mathbf{b}$.

The reduction sends $\mathsf{pp} := \overline{\mathbf{A}}$ to the adversary. The reduction then pre-generates $N-1$ public/secret key pairs for the right non-challenge user as in Lemma 6, except for using $\overline{\mathbf{b}}$ instead of $\mathbf{b}$ and thereby generates the matrix $\mathbf{Z} \in \mathcal{R}_q^{n\times n}$ whose row vectors are the LWE secrets of right (non-challenge) users whose unrounded shared key with the left challenge user lies in the danger zone. It uses this matrix to obtain the distribution $\mathcal{F}$ from Lemma 3. The reduction then samples $(\mathbf{f}_1, \mathbf{f}_2) \leftarrow \mathcal{F}$.

The reduction uses $\underline{\mathbf{A}}^\top$ as the public key for the right challenge user $\mathsf{id}_R^\star$ and $\overline{\mathbf{b}} + \mathbf{f}_1$ as the public key for the left challenge user. The public keys for left non-challenge users are generated honestly and the right non-challenge users the pre-generated keys are used. All $\mathcal{O}_{\mathsf{extrL}}$ and $\mathcal{O}_{\mathsf{extrR}}$ queries are answered honestly. All $\mathcal{O}_{\mathsf{revR}}$ queries and those $\mathcal{O}_{\mathsf{revL}}$ queries that do not involve the left challenge user are answered as in G_4. The $\mathcal{O}_{\mathsf{revL}}$ with the left challenge user are answered using the right user's secret key and potentially using the leakage ℓ about the left challenge user's secret and the hints $\mathbf{f}_2$ about the left challenge user's error to correct for the other secret key being used, analogously to Lemma 6.

For the challenge shared key, the reduction returns $\mathsf{Round}_\beta(\underline{\mathbf{b}})$ where $\beta = \underline{\mathsf{RF}}_L(\mathsf{id}_R^\star)$.

The reduction analysis is identical to the one for Lemma 6, except for the challenge query. For that, observe that if the reduction received a real LWE challenge, then $\underline{\mathbf{b}} = (\mathbf{s}_L^\star)^\top \underline{\mathbf{A}}^\top + (\underline{\mathbf{e}_L^\star})^\top$ where $(\underline{\mathbf{e}_L^\star})^\top$ is the last entry of $\mathbf{e}_L^\star$ which is distributed as $\mathcal{D}_{\mathcal{R},\sqrt{\mathbf{T}}}$. The reduction simulates G_4 in this case. If the reduction received a random LWE challenge, then $\underline{\mathbf{b}}$ is uniformly random and, since β is uniformly random, too, the output of $\mathsf{Round}_\beta(\underline{\mathbf{b}})$ is uniformly random. Thus, the reduction simulates G_5 in this case. □

Lemma 9 (Adv. in G_5).

$$\Pr[\mathsf{G}_5^{\mathcal{A}} \Rightarrow 1] = 1/2$$

Proof. In G_5, the adversary's view is independent of the challenge bit, and therefore every adversary will win the game with probability exactly 1/2. □
Combining Lemmata 4 – 9 completes the proof of Theorem 2. □

3.1 Reconciliation

In [21, 42] a protocol is shown that can correct potential errors in the shared key resulting from the LWE-based NIKE interactively by sending 1 bit. We show that this approach also works on already rounded keys, which becomes necessary in our work, since rounding is essential for the security proof. This "uses up" the two least significant bits of the shared key, so for a one bit shared key we now need to round it to three bits.

We require for the reconciliation mechanism that $2^\rho \mid q$ (to avoid the case where the shared key is just (pseudo)random) and $\rho \geq 3$. Let K_{12} and K_{21} be the rounded shared keys of a pair of users id_1 and id_2, where K_{12} was computed with the secret key of id_1 and K_{21} was computed with the secret key of id_2. The idea is that K_{12} and K_{21}, interpreted as numbers in $\mathbb{Z}_{2^\rho}$, can differ at most by 1 for suitable noise sizes. Either of the users can now send the second least significant bit of his shared key as a signal to the other user. If this matches the second least significant bit of his shared key, the first $\rho - 2$ bits of their shared key are identical as well. Otherwise, the signal-receiving user looks at the least significant bit. If this is 1, he obtains the same shared key as the other user by adding 1 to his shared key. On the other hand, if this bit is 0, he has to subtract 1 from his shared key. In all cases, the user can then use the $\rho - 2$ most significant bits as the shared key.

Acknowledgments. I would like to thank my supervisor Dennis Hofheinz for feedback on my ideas and an early draft of this paper. I also want to thank the reviewers of TCC 2023 for their suggestions.

References

[1] Shweta Agrawal, Benoît Libert, and Damien Stehlé. "Fully Secure Functional Encryption for Inner Products, from Standard Assumptions". In: *CRYPTO 2016, Part III*. Vol. 9816. LNCS. Aug. 2016, pp. 333–362. DOI: 10.1007/978-3-662-53015-3_12.

[2] Martin R. Albrecht and Russell W. F. Lai. "Subtractive Sets over Cyclotomic Rings - Limits of Schnorr-Like Arguments over Lattices". In: *CRYPTO 2021, Part II*. Vol. 12826. LNCS. Virtual Event, Aug. 2021, pp. 519–548. DOI: 10.1007/978-3-030-84245-1_18.

[3] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. "Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems". In: *CRYPTO 2009*. Vol. 5677. LNCS. Aug. 2009, pp. 595–618. DOI: 10.1007/978-3-642-03356-8_35.

[4] Christoph Bader, Tibor Jager, Yong Li, and Sven Schäge. "On the Impossibility of Tight Cryptographic Reductions". In: *EUROCRYPT 2016, Part II*. Vol. 9666. LNCS. May 2016, pp. 273–304. DOI: 10.1007/978-3-662-49896-5_10.

[5] Shi Bai, Steven D. Galbraith, Liangze Li, and Daniel Sheffield. "Improved Combinatorial Algorithms for the Inhomogeneous Short Integer Solution Problem". In: *Journal of Cryptology* 32.1 (Jan. 2019), pp. 35–83. DOI: 10.1007/s00145-018-9304-1.

[6] Wojciech Banaszczyk. "New bounds in some transference theorems in the geometry of numbers." In: *Mathematische Annalen* 296.4 (1993), pp. 625–636.

[7] Jonathan Bootle, Claire Delaplace, Thomas Espitau, Pierre-Alain Fouque, and Mehdi Tibouchi. "LWE Without Modular Reduction and Improved Side-Channel Attacks Against BLISS". In: *ASIACRYPT 2018, Part I*. Vol. 11272. LNCS. Dec. 2018, pp. 494–524. DOI: 10.1007/978-3-030-03326-2_17.

[8] Katharina Boudgoust, Corentin Jeudy, Adeline Roux-Langlois, and Weiqiang Wen. "Entropic Hardness of Module-LWE from Module-NTRU". In: *INDOCRYPT 2022*. Vol. 13774. LNCS. 2022, pp. 78–99. DOI: 10.1007/978-3-031-22912-1_4.

[9] Katharina Boudgoust, Corentin Jeudy, Adeline Roux-Langlois, and Weiqiang Wen. "Towards Classical Hardness of Module-LWE: The Linear Rank Case". In: *ASIACRYPT 2020, Part II*. Vol. 12492. LNCS. Dec. 2020, pp. 289–317. DOI: 10.1007/978-3-030-64834-3_10.

[10] Colin Boyd, Wenbo Mao, and Kenneth G. Paterson. "Key Agreement Using Statically Keyed Authenticators". In: *ACNS 04*. Vol. 3089. LNCS. June 2004, pp. 248–262. DOI: 10.1007/978-3-540-24852-1_18.

[11] Zvika Brakerski and Nico Döttling. "Hardness of LWE on General Entropic Distributions". In: *EUROCRYPT 2020, Part II*. Vol. 12106. LNCS. May 2020, pp. 551–575. DOI: 10.1007/978-3-030-45724-2_19.

[12] Zvika Brakerski and Nico Döttling. "Lossiness and Entropic Hardness for Ring-LWE". In: *TCC 2020, Part I*. Vol. 12550. LNCS. Nov. 2020, pp. 1–27. DOI: 10.1007/978-3-030-64375-1_1.

[13] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. "(Leveled) fully homomorphic encryption without bootstrapping". In: *ITCS 2012*. Jan. 2012, pp. 309–325. DOI: 10.1145/2090236.2090262.

[14] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. "Classical hardness of learning with errors". In: *45th ACM STOC*. June 2013, pp. 575–584. DOI: 10.1145/2488608.2488680.

[15] Cagatay Capar, Dennis Goeckel, Kenneth G. Paterson, Elizabeth A. Quaglia, Don Towsley, and Murtaza Zafer. "Signal-flow-based analysis of wireless security protocols". In: *Inf. Comput.* 226 (2013), pp. 37–56. DOI: 10.1016/j.ic.2013.03.004.

[16] David Cash, Eike Kiltz, and Victor Shoup. "The Twin Diffie-Hellman Problem and Applications". In: *EUROCRYPT 2008*. Vol. 4965. LNCS. Apr. 2008, pp. 127–145. DOI: 10.1007/978-3-540-78967-3_8.

[17] Wouter Castryck and Thomas Decru. "An Efficient Key Recovery Attack on SIDH". In: *EUROCRYPT 2023, Part V*. Vol. 14008. LNCS. Apr. 2023, pp. 423–447. DOI: 10.1007/978-3-031-30589-4_15.

[18] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. "CSIDH: An Efficient Post-Quantum Commutative Group Action". In: *ASIACRYPT 2018, Part III*. Vol. 11274. LNCS. Dec. 2018, pp. 395–427. DOI: 10.1007/978-3-030-03332-3_15.

[19] Dana Dachman-Soled, Léo Ducas, Huijing Gong, and Mélissa Rossi. "LWE with Side Information: Attacks and Concrete Security Estimation". In: *CRYPTO 2020, Part II*. Vol. 12171. LNCS. Aug. 2020, pp. 329–358. DOI: 10.1007/978-3-030-56880-1_12.

[20] Whitfield Diffie and Martin E. Hellman. "New Directions in Cryptography". In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654. DOI: 10.1109/TIT.1976.1055638.

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

[22] Yevgeniy Dodis, Jonathan Katz, Adam Smith, and Shabsi Walfish. "Composability and On-Line Deniability of Authentication". In: *TCC 2009*. Vol. 5444. LNCS. Mar. 2009, pp. 146–162. DOI: 10.1007/978-3-642-00457-5_10.

[23] Nico Döttling, Dimitris Kolonelos, Russell W. F. Lai, Chuanwei Lin, Giulio Malavolta, and Ahmadreza Rahimi. "Efficient Laconic Cryptography from Learning with Errors". In: *EUROCRYPT 2023, Part III*. Vol. 14006. LNCS. Apr. 2023, pp. 417–446. DOI: 10.1007/978-3-031-30620-4_14.

[24] Sebastian Faust, Markulf Kohlweiss, Giorgia Azzurra Marson, and Daniele Venturi. "On the Non-malleability of the Fiat-Shamir Transform". In: *INDOCRYPT 2012*. Vol. 7668. LNCS. Dec. 2012, pp. 60–79. DOI: 10.1007/978-3-642-34931-7_5.

[25] Eduarda S. V. Freire, Dennis Hofheinz, Eike Kiltz, and Kenneth G. Paterson. "Non-Interactive Key Exchange". In: *PKC 2013*. Vol. 7778. LNCS. 2013, pp. 254–271. DOI: 10.1007/978-3-642-36362-7_17.

[26] Phillip Gajland, Bor de Kock, Miguel Quaresma, Giulio Malavolta, and Peter Schwabe. *Swoosh: Practical Lattice-Based Non-Interactive Key Exchange*. Cryptology ePrint Archive, Report 2023/271. https://eprint.iacr.org/2023/271. 2023.

[27] Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. "Robustness of the Learning with Errors Assumption". In: *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*. 2010, pp. 230–240.

[28] Siyao Guo, Pritish Kamath, Alon Rosen, and Katerina Sotiraki. "Limits on the Efficiency of (Ring) LWE Based Non-interactive Key Exchange". In: *PKC 2020, Part I*. Vol. 12110. LNCS. May 2020, pp. 374–395. DOI: 10.1007/978-3-030-45374-9_13.

[29] Julia Hesse, Dennis Hofheinz, and Lisa Kohl. "On Tightly Secure Non-Interactive Key Exchange". In: *CRYPTO 2018, Part II*. Vol. 10992. LNCS. Aug. 2018, pp. 65–94. DOI: 10.1007/978-3-319-96881-0_3.

[30] Julia Hesse, Dennis Hofheinz, Lisa Kohl, and Roman Langrehr. "Towards Tight Adaptive Security of Non-interactive Key Exchange". In: *TCC 2021, Part III*. Vol. 13044. LNCS. Nov. 2021, pp. 286–316. DOI: 10.1007/978-3-030-90456-2_10.

[31] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. "Pseudo-random Generation from one-way functions (Extended Abstracts)". In: *21st ACM STOC*. May 1989, pp. 12–24. DOI: 10.1145/73007.73009.

[32] David Jao and Luca De Feo. "Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies". In: *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*. 2011, pp. 19–34. DOI: 10.1007/978-3-642-25405-5_2.

[33] Adeline Langlois and Damien Stehlé. "Worst-Case to Average-Case Reductions for Module Lattices". In: 75.3 (2015), 565–599. ISSN: 0925-1022. DOI: 10.1007/s10623-014-9938-4. URL: https://doi.org/10.1007/s10623-014-9938-4.

[34] Hao Lin, Mingqiang Wang, Jincheng Zhuang, and Yang Wang. *Hardness of Module-LWE and Ring-LWE on General Entropic Distributions*. Cryptology ePrint Archive, Report 2020/1238. https://eprint.iacr.org/2020/1238. 2020.

[35] Vadim Lyubashevsky. "Lattice Signatures without Trapdoors". In: *EUROCRYPT 2012*. Vol. 7237. LNCS. Apr. 2012, pp. 738–755. DOI: 10.1007/978-3-642-29011-4_43.

[36] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. "A Toolkit for Ring-LWE Cryptography". In: *EUROCRYPT 2013*. Vol. 7881. LNCS. May 2013, pp. 35–54. DOI: 10.1007/978-3-642-38348-9_3.

[37] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. "On Ideal Lattices and Learning with Errors over Rings". In: *EUROCRYPT 2010*. Vol. 6110. LNCS. 2010, pp. 1–23. DOI: 10.1007/978-3-642-13190-5_1.

[38] Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. "A Direct Key Recovery Attack on SIDH". In: *EUROCRYPT 2023, Part V*. Vol. 14008. LNCS. Apr. 2023, pp. 448–471. DOI: 10.1007/978-3-031-30589-4_16.

[39] Daniele Micciancio. "Generalized Compact Knapsacks, Cyclic Lattices, and Efficient One-Way Functions". In: *Comput. Complex.* 16.4 (2007), pp. 365–411. DOI: 10.1007/s00037-007-0234-9.

[40] Daniele Micciancio. "On the Hardness of Learning With Errors with Binary Secrets". In: *Theory of Computing* 14.1 (2018), pp. 1–17.

[41] Daniele Micciancio and Oded Regev. "Worst-Case to Average-Case Reductions Based on Gaussian Measures". In: *45th FOCS*. Oct. 2004, pp. 372–381. DOI: 10.1109/FOCS.2004.72.

[42] Chris Peikert. "Lattice Cryptography for the Internet". In: *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014*. Oct. 2014, pp. 197–219. DOI: 10.1007/978-3-319-11659-4_12.

[43] Chris Peikert. "Public-key cryptosystems from the worst-case shortest vector problem: extended abstract". In: *41st ACM STOC*. 2009, pp. 333–342. DOI: 10.1145/1536414.1536461.

[44] Chris Peikert, Oded Regev, and Noah Stephens-Davidowitz. "Pseudorandomness of ring-LWE for any ring and modulus". In: *49th ACM STOC*. June 2017, pp. 461–473. DOI: 10.1145/3055399.3055489.

[45] Oded Regev. "On lattices, learning with errors, random linear codes, and cryptography". In: *37th ACM STOC*. May 2005, pp. 84–93. DOI: 10.1145/1060590.1060603.

[46] Damien Robert. "Breaking SIDH in Polynomial Time". In: *EUROCRYPT 2023, Part V*. Vol. 14008. LNCS. Apr. 2023, pp. 472–503. DOI: 10.1007/978-3-031-30589-4_17.

A (In)security against dishonest user

In this section, we analyze for which parameters the LWE-based NIKE achieves adaptive DKR security through the transformation of [29] where every public key is equipped with a NIZKPoK of the secret key. The work of [29] contains a proof that this transformation yields an adaptively DKR-secure NIKE when the underlying NIKE achieves adaptive HKR security and is perfectly correct. We analyze for which parameters this transformation yields an adaptively DKR secure NIKE for the LWE-based NIKE, despite the LWE-based NIKE being not perfectly correct.

We recall the definition of NIZKPoKs.

Definition 10. *A NIZK for a NP-relation $\mathcal{R}$ with associated language $\mathcal{L} := \{x \mid \exists w : (x,w) \in \mathcal{R}\}$ consists of the following four PPT algorithms:*
- $\mathsf{Gen}_{\mathsf{NIZK}}$ *inputs the unary encoded security parameter and outputs at common reference string* crs *with a trapdoor* td*.*
- Prove *inputs* crs *and a statement witness pair $(x,w) \in \mathcal{R}$, and outputs a proof π.*
- Ver *inputs* crs*, a statement x, and a proof π and outputs a bit that indicates a valid or invalid proof.*
- Sim *inputs* crs *with the corresponding trapdoor* td *and a statement x and outputs a proof π.*

We require the following properties from the NIZK:

Definition 11 (Completeness). *A NIZK* $\mathsf{NIZK} = (\mathsf{Gen}_{\mathsf{NIZK}}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{Sim})$ *is correct if for all $\lambda \in \mathbb{N}_+$ and all $(x,w) \in \mathcal{R}$ we have for*
- $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Gen}_{\mathsf{NIZK}}(1^\lambda)$
- $\pi \leftarrow \mathsf{Prove}(\mathsf{crs}, (x,w))$

$$\Pr[\mathsf{Ver}(\mathsf{crs}, x, \pi) = 1] \geq 1 - \mathsf{negl}(\lambda)$$

for a negligible function negl*, where the probability is taken over the randomness used to sample* $(\mathsf{crs}, \mathsf{td})$ *and π and (if applicable) the randomness of* Ver*.*

Definition 12 (Zero-knowledge). *A NIZK* $\mathsf{NIZK} = (\mathsf{Gen}_{\mathsf{NIZK}}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{Sim})$ *is zero-knowledge if for all $\lambda \in \mathbb{N}_+$ and all PPT adversaries $\mathcal{A}$ we have*

$$\mathsf{Adv}^{\mathsf{zk}}_{\mathcal{A},\mathsf{NIZK}}(\lambda) := \left| \Pr[\mathsf{Exp}^{\mathsf{zk}}_{\mathcal{A},\mathsf{NIZK}}(\lambda) = 1] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda)$$

for a negligible function negl *where the experiment is defined in Figure 9.*

The following definition of a proof of knowledge in the presence of simulated proofs is from [29]. It is stronger than the standard proof of knowledge property, where the adversary has no access to simulated proofs.

Definition 13 (Proof of knowledge). *A non-interactive zero-knowledge proof of knowledge (NIZKPoK) in the presence of simulated proofs consists of PPT algorithms* $\mathsf{Gen}_{\mathsf{NIZK}}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{Sim}, \mathsf{Extr}$ *where*

$\underline{\mathsf{Exp}^{\mathsf{zk}}_{\mathcal{A},\mathsf{NIZK}=(\mathsf{Gen}_{\mathsf{NIZK}},\mathsf{Prove},\mathsf{Ver},\mathsf{Sim})}(\lambda)}$:
$(\mathsf{crs},\mathsf{td}) \leftarrow \mathsf{Gen}_{\mathsf{NIZK}}(1^\lambda)$
$b \xleftarrow{\$} \{0,1\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{prove}}(\cdot,\cdot)}(1^\lambda,\mathsf{crs})$
return $b \stackrel{?}{=} b'$

$\underline{\mathcal{O}_{\mathsf{prove}}(x,w)}$:
if $(x,w) \notin \mathcal{R}$ **then return** $\bot$
if $b = 0$ **then**
| **return** $\mathsf{Prove}(\mathsf{crs},(x,w))$
else
⌊**return** $\mathsf{Prove}(\mathsf{crs},\mathsf{td},x)$

Fig. 9. The zero-knowledge experiment for a NIZK.

$\underline{\mathsf{Exp}^{\mathsf{pok}}_{\mathcal{A},\mathsf{NIZKPoK}=(\mathsf{Gen}_{\mathsf{NIZK}},\mathsf{Prove},\mathsf{Ver},\mathsf{Sim},\mathsf{Extr})}(\lambda)}$:
$(\mathsf{crs},\mathsf{td}) \leftarrow \mathsf{Gen}_{\mathsf{NIZK}}(1^\lambda)$
$Q_{\mathsf{sim}} := \emptyset$
$(x^\star,\pi^\star) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{sim}}(\cdot),\mathcal{O}_{\mathsf{extr}}(\cdot,\cdot)}(1^\lambda,\mathsf{crs})$
$w^\star \leftarrow \mathsf{Extr}(\mathsf{crs},\mathsf{td},x^\star,\pi^\star)$
if $\mathsf{Ver}(\mathsf{crs},x^\star,\pi^\star) = 1 \wedge (x^\star,w^\star) \notin \mathcal{R} \wedge x^\star \notin Q_{\mathsf{sim}}$ **then**
| **return** *1*
else
⌊**return** *0*

$\underline{\mathcal{O}_{\mathsf{sim}}(x)}$:
$Q_{\mathsf{sim}} := Q_{\mathsf{sim}} \cup \{x\}$
return $\mathsf{Prove}(\mathsf{crs},\mathsf{td},x)$

$\underline{\mathcal{O}_{\mathsf{extr}}(x,\pi)}$:
if $x \in Q_{\mathsf{sim}}$ **then**
| **return** $\bot$
else
⌊**return** $\mathsf{Extr}(\mathsf{crs},\mathsf{td},x,\pi)$

Fig. 10. The proof of knowledge in the presence of simulated proofs experiment.

- $(\mathsf{Gen}_{\mathsf{NIZK}},\mathsf{Prove},\mathsf{Ver},\mathsf{Sim})$ *is a NIZK*
- Extr *inputs* crs *with the corresponding trapdoor* td *and a statement* x *with a proof* Prove *and outputs a witness* w.

We also require that for all $\lambda \in \mathbb{N}_+$ *and all PPT adversaries* $\mathcal{A}$ *we have*

$$\mathsf{Adv}^{\mathsf{pok}}_{\mathcal{A},\mathsf{NIZKPoK}}(\lambda) := \Pr[\mathsf{Exp}^{\mathsf{pok}}_{\mathcal{A},\mathsf{NIZKPoK}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$$

for a negligible function negl *where the experiment is defined in Figure 10.*

We now recall the transformation of [29], but adapted to the asymmetric NIKE notion with left and right users that we use in this paper. For this, we need two instances of the NIZKPoK for relations $\mathcal{R}_L$, and $\mathcal{R}_R$ such that for

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
- $(\mathsf{pk}_L,\mathsf{sk}_L) \leftarrow \mathsf{KeyGenL}(\mathsf{pp},\mathsf{id}_L)$
- $(\mathsf{pk}_R,\mathsf{sk}_R) \leftarrow \mathsf{KeyGenR}(\mathsf{pp},\mathsf{id}_R)$

with overwhelming probability $((\mathsf{pp},\mathsf{id}_L,\mathsf{pk}_L),\mathsf{sk}_L) \in \mathcal{R}_L$ and $((\mathsf{pp},\mathsf{id}_R,\mathsf{pk}_R),\mathsf{sk}_R) \in \mathcal{R}_R$ holds.

In the transformed NIKE, both crs are generated in Setup and become part of pp. Left public keys come with a NIZKPoK for $\mathcal{R}_L$, and right public keys with a NIZKPoK $\mathcal{R}_R$. The NIZKPoKs are verified in the $\mathsf{SharedKeyL}$ and $\mathsf{SharedKeyR}$ algorithm and if they do not verify, the algorithm outputs $\bot$. Otherwise, it proceeds as before. The formal transformation is shown in Figure 11.

$\underline{\mathsf{Setup}'(1^\lambda)}$:
$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
$\mathsf{crs}_L \leftarrow \mathsf{Gen}_{\mathsf{NIZK},L}(1^\lambda)$
$\mathsf{crs}_R \leftarrow \mathsf{Gen}_{\mathsf{NIZK},R}(1^\lambda)$
return $\mathsf{pp}' := (\mathsf{pp}, \mathsf{crs}_L, \mathsf{crs}_R)$

$\underline{\mathsf{KeyGenL}'(\mathsf{pp}', \mathsf{id}_L)}$:
parse $\mathsf{pp}' =: (\mathsf{pp}, \mathsf{crs}_L, \mathsf{crs}_R)$
$(\mathsf{pk}_L, \mathsf{sk}_L) \leftarrow \mathsf{KeyGenL}(\mathsf{pp}, \mathsf{id}_L)$
$\pi_L := \mathsf{Prove}_L(\mathsf{crs}_L, ((\mathsf{pp}, \mathsf{id}, \mathsf{pk}_L), \mathsf{sk}_L))$
return $(\mathsf{pk}'_L := (\mathsf{pk}_L, \pi_L), \mathsf{sk}_L)$

$\underline{\mathsf{KeyGenR}'(\mathsf{pp}', \mathsf{id}_R)}$:
parse $\mathsf{pp}' =: (\mathsf{pp}, \mathsf{crs}_L, \mathsf{crs}_R)$
$(\mathsf{pk}_R, \mathsf{sk}_R) \leftarrow \mathsf{KeyGenL}(\mathsf{pp}, \mathsf{id}_L)$
$\pi_R := \mathsf{Prove}_R(\mathsf{crs}_R, ((\mathsf{pp}, \mathsf{id}, \mathsf{pk}_R), \mathsf{sk}_R))$
return $(\mathsf{pk}'_R := (\mathsf{pk}_R, \pi_R), \mathsf{sk}_R)$

$\underline{\mathsf{SharedKeyL}'(\mathsf{pp}', \mathsf{id}_R, \mathsf{pk}'_R, \mathsf{id}_L, \mathsf{sk}_L)}$:
parse $\mathsf{pp}' =: (\mathsf{pp}, \mathsf{crs}_L, \mathsf{crs}_R)$
parse $\mathsf{pk}'_R =: (\mathsf{pk}_R, \pi_R)$
if $\mathsf{Ver}_R(\mathsf{crs}_R, (\mathsf{pp}, \mathsf{id}_R, \mathsf{pk}_R), \pi_R) \stackrel{?}{=} 1$ **then**
| **return** $\mathsf{SharedKeyL}(\mathsf{pp}, \mathsf{id}_R, \mathsf{pk}_R, \mathsf{id}_L, \mathsf{sk}_L)$
else
└**return** $\bot$

$\underline{\mathsf{SharedKeyR}'(\mathsf{pp}', \mathsf{id}_L, \mathsf{pk}'_L, \mathsf{id}_R, \mathsf{sk}_R)}$:
parse $\mathsf{pp}' =: (\mathsf{pp}, \mathsf{crs}_L, \mathsf{crs}_R)$
parse $\mathsf{pk}'_L =: (\mathsf{pk}_L, \pi_L)$
if $\mathsf{Ver}_L(\mathsf{crs}_L, (\mathsf{pp}, \mathsf{id}_L, \mathsf{pk}_L), \pi_L) \stackrel{?}{=} 1$ **then**
| **return** $\mathsf{SharedKeyR}(\mathsf{pp}, \mathsf{id}_L, \mathsf{pk}_L, \mathsf{id}_R, \mathsf{sk}_R)$
else
└**return** $\bot$

Fig. 11. The transformation of a NIKE $\mathsf{NIKE} = (\mathsf{Setup}, \mathsf{KeyGenL}, \mathsf{KeyGenR}, \mathsf{SharedKeyL}, \mathsf{SharedKeyR})$, that is required to satisfy distribution correctness adaptive HKR security, to a NIKE $\mathsf{NIKE}' = (\mathsf{Setup}', \mathsf{KeyGenL}', \mathsf{KeyGenR}', \mathsf{SharedKeyL}', \mathsf{SharedKeyR}')$ that satisfies adaptive DKR security. The transformation requires a NIZKPoK $\mathsf{NIZKPoK}_L = (\mathsf{Gen}_{\mathsf{NIZK},L}, \mathsf{Prove}_L, \mathsf{Ver}_L, \mathsf{Sim}_L, \mathsf{Extr}_L)$ for $\mathcal{R}_L$ and a NIZKPoK $\mathsf{NIZKPoK}_R = (\mathsf{Gen}_{\mathsf{NIZK},R}, \mathsf{Prove}_R, \mathsf{Ver}_R, \mathsf{Sim}_R, \mathsf{Extr}_R)$ for $\mathcal{R}_R$.

For the LWE-based NIKE we use the following relations. Let $B_{\mathbf{s}}, B_{\mathbf{e}} \in \mathbb{N}_+$ be such that with overwhelming probability $\|\mathbf{s}\|_\infty \leq B_{\mathbf{s}}$ for $\mathbf{s} \leftarrow \mathcal{S}$ and $\|\mathbf{e}\|_\infty \leq B_{\mathbf{e}}$ for $\mathbf{e} \leftarrow \mathcal{E} := \mathcal{D}_{\mathcal{R}^n, \sqrt{\mathbf{\Sigma}}}$. For our transformation to adaptive DKR security, we need two NIZKPoKs for the relations

$$\mathcal{R}_L := \{((\mathbf{A}, \mathsf{id}, \mathsf{pk}_L), \mathbf{s}_L) \mid \|\mathbf{s}_L\|_\infty \leq B_{\mathbf{s}} \wedge \|\mathsf{pk}_L - \mathbf{s}_L^\top \mathbf{A}\|_\infty \leq B_{\mathbf{e}}\}$$
$$\mathcal{R}_R := \{((\mathbf{A}, \mathsf{id}, \mathsf{pk}_R), \mathbf{s}_R) \mid \|\mathbf{s}_R\|_\infty \leq B_{\mathbf{s}} \wedge \|\mathsf{pk}_R - \mathbf{A}\mathbf{s}_R\|_\infty \leq B_{\mathbf{e}}\}.$$

In this section, we analyze the LWE-based NIKE with different choices for the LWE secret distribution $\mathcal{S}$, the LWE noise distribution $\mathcal{E}$ and the shared key noise distribution $\mathcal{E}'$. For simplicity, we omit the pseudo-random function used to generate the randomness of the shared key generation algorithm in this section. The result of applying the transformation of Figure 11 to this NIKE is shown in Figure 12.

QANIZKPoK. We can allow the NIZKPoKs to depend on the public parameters of the underlying NIKE, but we still need one simulator that works for all public parameters. This notion is captured by a quasi-adaptive NIZKPoK (QANIZKPoK) and allows for more efficient instantiations over groups. Thus, it is used in [29]. We are not aware of any such results for LWE-based constructions and thus ignore this optimization.

$\underline{\mathsf{Setup}(1^\lambda)}$:
$\mathbf{A} \xleftarrow{\$} \mathcal{R}_q^{n\times n}$
return $\mathsf{pp} := \mathbf{A}$

$\underline{\mathsf{KeyGenL}(\mathsf{pp}, \mathsf{id}_L)}$:
$\mathbf{s}_L \leftarrow \mathcal{S};\ \mathbf{e}_L \leftarrow \mathcal{E}$
$\mathsf{pk}_L := \mathbf{s}_L^\top \mathbf{A} + \mathbf{e}_L^\top$
return $(\mathsf{pk}_L, \mathsf{sk}_L = \mathbf{s}_L)$

$\underline{\mathsf{KeyGenR}(\mathsf{pp}, \mathsf{id}_R)}$:
$\mathbf{s}_R \leftarrow \mathcal{S};\ \mathbf{e}_R \leftarrow \mathcal{E}$
$\mathsf{pk}_R := \mathbf{A}\mathbf{s}_R + \mathbf{e}_R$
return $(\mathsf{pk}_R, \mathsf{sk}_R = \mathbf{s}_R)$

$\underline{\mathsf{SharedKeyL}(\mathsf{pp}, \mathsf{id}_R, \mathsf{pk}_R, \mathsf{id}_L, \mathsf{sk}_L)}$:
$\beta \xleftarrow{\$} 0, 1^\rho$
$e' \xleftarrow{\$} \mathcal{E}'$
return $\mathsf{Round}_\beta(\mathsf{sk}_L^\top \mathsf{pk}_R + \psi^{-1}(e'))$

$\underline{\mathsf{SharedKeyR}(\mathsf{pp}, \mathsf{id}_L, \mathsf{pk}_L, \mathsf{id}_R, \mathsf{sk}_R)}$:
$\beta \xleftarrow{\$} 0, 1^\rho$
$e' \xleftarrow{\$} \mathcal{E}'$
return $\mathsf{Round}_\beta(\mathsf{pk}_L \mathsf{sk}_R + \psi^{-1}(e'))$

Fig. 12. The LWE based NIKE we consider in this section.

On the use of identities. To achieve DKR security, it is important that public keys depend on the users' identities, because otherwise there is a simple replay attack against the adaptive DKR security: In this case, an attacker can register the public key of a challenge user as the public key of another dishonest user. The security game now allows the attacker to get the shared key between the other challenge user and the dishonest user, even though this key will be identical to the challenge users' shared key. However, for adaptive HKR secure NIKE schemes, the public keys do not have to depend on the identities (and in most schemes in the adaptive HKR model they do not). Therefore, it is crucial that identities are part of the statement in our NIZKPoKs. For the standard proof of knowledge property (without simulated proofs), it would not offer any additional security guarantees to add the identities to the statement when the underlying public keys (and therefore the truth of the statement) are independent of the identity. However, for the proof of knowledge in the presence of simulated proofs property, adding the identities guarantees that an adversary cannot alter a proof for an identity id to a proof for another identity id' (with the same public key). In concrete proof systems, the identity can be used as a tag (e.g., in [29]) or by hashing the statement (which includes the identity) in Fiat-Shamir based constructions [24].

A.1 Insecurity for polynomial shared key noise

We present an attacker against the adaptive DKR security of LWE-based NIKE in Figure 12 that is able to extract secret keys of other users (and thus win the DKR security game) if the size of the noise added to the shared keys is of polynomial size. For simplicity, we present the attack for the ring $\mathcal{R} = \mathbb{Z}$, but it can be easily generalized to other rings.

We first present a simplified attack that works when no noise is added to the shared keys and the secret keys are binary. We also assume a slightly different

rounding function that will round all values slightly larger or equal to zero to the shared key 0^ρ and all values slightly smaller or equal to $q-1$ to the shared key 1^ρ.

To extract the secret key of a left user with secret key $\mathsf{sk}_L = \mathbf{s}_L$ the adversary will perform several queries, each extracting one bit of the secret key. To extract the i-th bit, the adversary registers a dishonest right user with the public key $\mathsf{pk}_R = -e_i$ where e_i is the i-th unit vector. The adversary honestly computes the corresponding NIZKPoK with $\mathsf{sk}_R = \mathbf{0}$. This clearly is a valid secret key for the public key. It then asks for the shared key of this dishonest user with the left user to attack. The unrounded shared key here is simply the i-th entry of $\mathbf{s}_i$. From this, the adversary learns $\mathsf{Round}_\beta(\mathbf{s}_i)$, which is 0^ρ if $\mathbf{s}_i = 0$ and 1^ρ if $\mathbf{s}_i = 1$. Therefore, with n dishonest key registrations, the adversary can completely extract one user's secret key.

Larger entries in the LWE secret. Next, we describe how to generalize this attack to any LWE secrets with $\|\mathbf{s}\|_\infty \leq B_\mathbf{e}$, that is, LWE secrets that are not larger than the maximum allowed noise for the public keys. We further assume for the attack that the first entry of $\mathbf{s}$ is 1 with noticeable probability, which is satisfied by all natural choices of the LWE secret distribution. The particular choice of 1 is not important; the attack can be generalized to any (small) value with a noticeable probability.

To extract the i-th entry of $\mathbf{s}$ for $i \geq 2$, the adversary now proceeds iteratively while keeping track of an interval $I \subseteq \mathbb{Z}_q$ that contains $\mathbf{s}_i$. Initially, this interval is $[-B_\mathbf{s}, B_\mathbf{s}]$. In each round, the adversary will use one reveal shared key query with a dishonest user to half the size of this interval by a binary search. Therefore, let $p \in I$ be from the middle of the interval, that is, $|\{x \in I \mid x < p\}| = |\{x \in I \mid x \geq p\}| + \delta$ for $\delta \in \{0, 1\}$. To find out whether $\mathbf{s}_i \geq p$ or not, the attacker registers a dishonest right user with public key $p \cdot e_1 + e_i$ (and uses the secret key $\mathsf{sk}_R = \mathbf{0}$ for the NIZKPoK) and asks for the left challenge user's shared key with this user. When $\mathbf{s}_1 = 1$, we learn from this query whether $p + \mathbf{s}_i \geq 0$ or not.

With this attack, we can recover the entire secret key of one challenge user with $\mathcal{O}(n \log(B_\mathbf{s}))$ dishonest key registrations with noticeable probability. The attack can be extended to settings where $\|\mathbf{s}\|_\infty < B_\mathbf{e}$ does not hold, by guessing that the first entry of $\mathbf{s}$ is a larger value than 1 or possibly guessing a constant number of values (e.g., $\mathbf{s}_1 = 1$, $\mathbf{s}_2 = B_\mathbf{e}$, $\mathbf{s}_3 = B_\mathbf{e}^2$).

In the presence of noise. When noise is added to the shared keys, we have to make some more modifications to the attack. Concretely, let $B_{e'}$ be a bound on the noise added to the shared keys. The first modification is to replace the dishonest public keys $p \cdot e_1 + e_i$ with $(p + \lfloor B_\mathbf{e}/2 \rfloor) \cdot e_1 + B_\mathbf{e} \cdot e_i$. This separates the shared keys resulting from different values of $\mathbf{s}_i$. When $B_K \leq \lfloor B_\mathbf{e}/2 \rfloor$, this change already suffices for a successful attack.

If the noise added to the shared keys is larger than this, we have to add a safety margin to the interval boundaries. From a query as described above, we can still learn whether $p + \mathbf{s}_i \geq -(B_K - \lfloor B_\mathbf{e}/2 \rfloor)$ or $p + \mathbf{s}_i \leq (B_K - \lfloor B_\mathbf{e}/2 \rfloor)$. This can still be used to shrink the size of the interval until it reaches size

$\approx 2B_K - B_{\mathbf{e}}$. After this point, the attacker has to perform several queries with the same dishonest public key per halving of the interval size.

Therefore, let $p' \in \mathbb{Z} \cap [-B_K, B_K]$ be such that $\Pr_{x \leftarrow \mathcal{E}'}[x \geq p'] - \Pr_{x \leftarrow \mathcal{E}'}[x \geq p' + 1]$ is non-negligible, where $\mathcal{E}'$ is the distribution of the noise added to the unrounded shared keys. By the pigeonhole principle, there exists one p' where this difference of probabilities is at least $1/2B_K$, which is non-negligible when B_K is polynomial. Now we change the public keys the adversary uses to $(p + p' - 1 + \lfloor B_{\mathbf{e}}/2 \rfloor) \cdot e_1 + B_{\mathbf{e}} \cdot e_i$. With each such public key, the shared key with the challenge user being 0^ρ is more likely by $1/\mathsf{poly}(B_{\mathbf{s}})$ for a polynomial poly if $p + \mathbf{s}_i \geq 0$ than if $p + \mathbf{s}_i < 0$. With a polynomial number of such queries, the attacker can learn whether $p + \mathbf{s}_i \geq 0$ or $p + \mathbf{s}_i < 0$ with overwhelming probability.

Different rounding functions. The attack can also be generalized (with some caveats) to any rounding function with the property that there exists some $v \in \mathbb{Z}_q$ such that all values slightly larger than or equal to v are rounded to one shared key, while values slightly smaller than v are rounded to a different shared key. This captures the rounding function suggested by [26]. The attack presented so far works only for $v = 0$.

The attack for $v \neq 0$ begins by computing short vectors s' and e' such that $\mathsf{pk}_L \dot{s} + e = -v$, where pk_L is the left public key whose secret key we try to extract. When such vectors are known, the attack proceeds as before, except that it adds s' and e' as offsets to the secret and error of all dishonest keys it registers. These offsets shift the shared key with the user to attack by $-v$, so the rounded shared keys are identical to the shared keys one would obtain without the offset and a rounding function with $v = 0$.

Finding such vectors s' and e' is an (approximate) modular subset sum problem. A naive algorithm to solve the (approximate) modular subset sum problem is to sample short s' until $|\mathsf{pk}_L \dot{s} - v| \leq B_e/2$ where B_e is the bound on the error size. (The division by 2 is to ensure that we still have some room for the error vectors required for the attack). This takes time $\mathcal{O}(q/B_e)$. With a meet-in-the-middle approach, the runtime can be reduced to $\mathcal{O}(\sqrt{q/B_e})$. There are better algorithms for the (approximate) modular subset sum/ISIS problems, such as the CPW algorithm [5], but unfortunately the CPW algorithm does not have a rigorous run-time analysis.

For a polynomial modulus-to-noise ratio, this leads to a polynomial-time attack. For a super-polynomial modulus-to-noise ratio, the attack runs in super-polynomial time. However, we consider it still relevant because the modular subset sum problem is potentially much easier to solve than breaking the LWE instance. Most importantly, the modular subset sum can only become easier when we increase the dimension n (because we can always reduce the dimension by fixing some entries in s' to be 0, as long as a short solution still exists) and increasing the dimension n is usually the main tool to increase the hardness of the LWE problem.

In summary, we can get an efficient attacker against the adaptive DKR security of essentially all variants of the LWE-based NIKE when a polynomial

modulus is used and some variants with a super-polynomial modulus (as long as the correctness error is not overwhelming).

A.2 DKR security for LWE-based NIKE

In this section, we show that super-polynomially large noise added to the shared keys can not only prevent the above attack, we can actually prove security very easily when the noise is large enough to smudge the inner product of an LWE secret and error vector. The drawback of this approach is that it requires a super-polynomial modulus-to-noise ratio.

Comparison with Swoosh. The Swoosh paper [26] takes an entirely different approach to make the LWE-based NIKE adaptively DKR secure. Namely, they add an offset to the unrounded shared keys before rounding them. This offset is the output of a (Q)ROM query on the identities and public keys of the pair of users. In this way, both users can compute this offset, but it is still unpredictable during generation of the public and secret key pair. Due to this offset, the NIKE achieves semi-malicious correctness, which informally means that the NIKE has negligible correctness error even for maliciously generated keys. This allows a reduction to change which user's secret key is used in a shared key generation even for maliciously generated users. Together with NIZKPoKs, this is sufficient for adaptive DKR security. Unfortunately, this idea seems to be difficult to realize in the standard model.

Instead, we show that a weaker correctness notion, which we call distribution correctness, suffices to achieve DKR security. Informally, it means that (even for maliciously generated keys), the distribution of the shared key, where the probability is taken over the randomness used in the shared key generation algorithm, does not depend on which user's secret key is used.

Definition 14 (Distribution correctness). *A NIKE satisfies* distribution correctness *with respect to key relations $\mathcal{R}_L$ and $\mathcal{R}_R$ if for all $\lambda \in \mathbb{N}_+$, all pp in the range of Setup, all $\mathsf{id}_L, \mathsf{id}_R \in \mathcal{IDS}$, all $((\mathsf{pp}, \mathsf{id}_L, \mathsf{pk}_L), \mathsf{sk}_L) \in \mathcal{R}_L$ and all $((\mathsf{pp}, \mathsf{id}_R, \mathsf{pk}_R), \mathsf{sk}_R) \in \mathcal{R}_R$ we have*

$$\mathsf{SD}(\mathsf{SharedKeyL}(\mathsf{pp}, \mathsf{id}_R, \mathsf{pk}_R, \mathsf{id}_L, \mathsf{sk}_L), \mathsf{SharedKeyR}(\mathsf{pp}, \mathsf{id}_L, \mathsf{pk}_L, \mathsf{id}_R, \mathsf{sk}_R)) \leq 1 - \varepsilon_{\mathsf{dist}}(\lambda)$$

for a negligible function $\varepsilon_{\mathsf{dist}}$, where the probability is taken over the randomness of the $\mathsf{SharedKeyL}$ *and* $\mathsf{SharedKeyR}$ *algorithm. We call $\varepsilon_{\mathsf{dist}}$ the distribution correctness error.*

This definition implies the usual correctness with a negligible correctness error when $\mathsf{SharedKeyL}$ and $\mathsf{SharedKeyR}$ are deterministic, but it can capture NIKE schemes without statistical correctness when $\mathsf{SharedKeyL}$ and $\mathsf{SharedKeyR}$ are probabilistic. For example, the trivial NIKE where $\mathsf{SharedKeyL}$ and $\mathsf{SharedKeyR}$ output uniformly random shared keys achieves perfect distribution correctness

despite having a large correctness error. Note that we can always make $\mathsf{SharedKeyL}$ and $\mathsf{SharedKeyR}$ deterministic in the end by generating the randomness with a PRF, where each user has its own key for the PRF stored in its secret key, as we did in Section 3. However, here it is not essential for these algorithms to be deterministic.

Distribution correctness of the LWE-based NIKE. We recall the smudging lemma.

Lemma 10 (Smudging Lemma). *Let $B_1(\lambda), B_2(\lambda) \in \mathbb{N}_+$ be positive integers depending on the security parameter. Then for*
- $e_1 \xleftarrow{\$} \mathbb{Z} \cap [-B_1, B_1]$ *and*
- $e_2 \xleftarrow{\$} \mathbb{Z} \cap [-B_2, B_2]$,

$$\mathsf{SD}(e_2, e_1 + e_2) = \frac{B_1^2 + B_1}{B_2},$$

which is negligible if B_2 is super-polynomially larger than B_1.

Let $B_{\mathbf{s}}$, $B_{\mathbf{e}}$, $\mathcal{R}_L$ and $\mathcal{R}_R$ be as described in the beginning of the section. We use the smudging lemma to show that the NIKE shown in Figure 12 satisfies distribution correctness if $\mathcal{E}'$ is the uniform distribution on $\mathbb{Z}^d \cap [-B_K, B_K]^d$ and B_K is super-polynomial in $B_{\mathbf{s}}$, $B_{\mathbf{e}}$, and $\gamma_{\mathcal{R}}$. Let us call this NIKE $\mathsf{NIKE}_{\mathsf{smudge}}$.

Theorem 3. *The NIKE $\mathsf{NIKE}_{\mathsf{smudge}}$ satisfies distribution correctness with*

$$\varepsilon_{\mathsf{dist}}(\lambda) = \frac{(B_{\mathbf{s}} B_{\mathbf{e}} \gamma_{\mathcal{R}})^2 + B_{\mathbf{s}} B_{\mathbf{e}} \gamma_{\mathcal{R}}}{B_K}$$

Proof. Let $\mathsf{pk}_L = \mathbf{s}_L^\top \mathbf{A} + \mathbf{e}_L^\top$, $\mathsf{pk}_R = \mathbf{A}\mathbf{s}_R^\top + \mathbf{e}_R$ and $\mathsf{id}_L, \mathsf{id}_R \in \mathcal{IDS}$ such that $((\mathbf{A}, \mathsf{id}_L, \mathsf{pk}_L), \mathbf{s}_L) \in \mathcal{R}_L$ and $((\mathbf{A}, \mathsf{id}_R, \mathsf{pk}_R), \mathbf{s}_R) \in \mathcal{R}_R$. Then

$$\mathbf{s}_L^\top \mathsf{pk}_R = \mathbf{s}_L^\top \mathbf{A}\mathbf{s}_R + \mathbf{s}_L^\top \mathbf{e}_R \text{ and}$$
$$\mathsf{pk}_L \mathbf{s}_R = \mathbf{s}_L^\top \mathbf{A}\mathbf{s}_R + \mathbf{e}_L^\top \mathbf{s}_R.$$

Since

$$\|\mathbf{s}_L^\top \mathbf{e}_R\|_\infty, \|\mathbf{e}_L^\top \mathbf{s}_R\|_\infty \leq B_{\mathbf{s}} B_{\mathbf{e}} \gamma_{\mathcal{R}}$$

and B_K is of super-polynomial size in $B_{\mathbf{s}}$, $B_{\mathbf{e}}$, and $\gamma_{\mathcal{R}}$, by Lemma 10

$$\mathsf{SD}(\mathbf{s}_L^\top \mathsf{pk}_R + e', \mathsf{pk}_L \mathbf{s}_R + e'') \leq \frac{(B_{\mathbf{s}} B_{\mathbf{e}} \gamma_{\mathcal{R}})^2 + B_{\mathbf{s}} B_{\mathbf{e}} \gamma_{\mathcal{R}}}{B_K},$$

where the probability is taken over $e', e'' \xleftarrow{\$} [-B_K, B_K]$. □

Light security of $\mathsf{NIKE}_{\mathsf{smudge}}$ (if the parameters are suitable for the hardness of LWE) follows by the proof of [21, 42], which we also sketched in Section 1.2.

Theorem 4. *For every PPT adversary $\mathcal{A}$ against the light security of $\mathsf{NIKE}_{\mathsf{smudge}}$ there exists a PPT adversary $\mathcal{B}$ against LWE with*

$$\mathsf{Adv}^{\mathsf{light}}_{\mathcal{A},\mathsf{NIKE}_{\mathsf{smudge}}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{lwe}}_{\mathcal{R},n,n+1,q,\mathcal{S},\mathcal{E}',\perp}(\mathcal{A}),$$

where $\mathcal{E}$ is a distribution that outputs a vector whose first dn components are distributed as $\mathsf{NIKE}_{\mathsf{smudge}}$ and the last d components are sampled uniformly at random from $[-B_K, B_K]$.

It is straightforward to generalize the transformation of [25] from light to adaptive security to NIKEs with distribution correctness.

Theorem 5. *If* NIKE *is distribution correct (for any relations $\mathcal{R}_L$ and $\mathcal{R}_R$) and is lightly secure, it is also adaptive HKR secure. Concretely, let $\varepsilon_{\mathsf{dist}}$ be the distribution correctness error of* NIKE*. Then for every PPT adversary $\mathcal{A}$ against the adaptive HKR security of NIKE, there exists a PPT adversary $\mathcal{B}$ against the light adaptive security of* NIKE *with*

$$\mathsf{Adv}^{\mathsf{adaptive\text{-}HKR}}_{\mathcal{A},\mathsf{NIKE}}(\lambda) \leq N^2 \mathsf{Adv}^{\mathsf{light}}_{\mathcal{B},\mathsf{NIKE}}(\lambda) + q_{\mathsf{rev}} \varepsilon_{\mathsf{dist}}(\lambda),$$

where N is the maximum number of $\mathcal{O}_{\mathsf{regHL}}$ and $\mathcal{O}_{\mathsf{regHR}}$ and q_{rev} the maximum number of $\mathcal{O}_{\mathsf{revL}}$ and $\mathcal{O}_{\mathsf{revR}}$ queries of $\mathcal{A}$

The proof is a straightforward adaptation of [25]. We recall it here for the sake of completeness.

Proof. The reduction forwards pp and guesses the index of the $\mathcal{O}_{\mathsf{regHL}}$ and $\mathcal{O}_{\mathsf{regHR}}$ queries in which the users used for the $\mathcal{O}_{\mathsf{testL}}$ or $\mathcal{O}_{\mathsf{testR}}$ query are registered. With probability $1/N^2$, both guesses will be correct. In these registration queries, the reduction embeds the left and right public keys obtained from the light security game. All other registration queries are answered with honestly generated public keys. This also enables the reduction to answer all other queries, except for $\mathcal{O}_{\mathsf{revL}}$ queries with the left challenge user and $\mathcal{O}_{\mathsf{revR}}$ queries with the right challenge user. The reduction answers these $\mathcal{O}_{\mathsf{revL}}$ by using the $\mathsf{SharedKeyR}$ algorithm with the secret key of the right, non-challenge user and similarly these $\mathcal{O}_{\mathsf{revR}}$ by using the $\mathsf{SharedKeyL}$ algorithm with the secret key of the left, non-challenge user. This changes the output distribution of these queries at most by $\varepsilon_{\mathsf{dist}}(\lambda)$. □

Transformation with NIZKPoK. Next, we show that applying the transformation of [29] to an adaptive HKR secure NIKE with distribution correctness yields an adaptive DKR secure NIKE.

Theorem 6. *Let* NIKE′ *be a NIKE obtained by applying the transformation of Figure 11 to a NIKE* NIKE *that is distribution correct with respect to the relations $\mathcal{R}_L$ and $\mathcal{R}_R$, and to the NIZKPoKs $\mathsf{NIZKPoK}_L$, $\mathsf{NIZKPoK}_R$ for the relations $\mathcal{R}_L$, and $\mathcal{R}_R$. Then, if* NIKE *is adaptively HKR secure,* NIKE′ *is adaptively DKR secure. Concretely, for every PPT adversary $\mathcal{A}$ against the adaptive DKR security of* NIKE′ *there exist PPT adversaries $\mathcal{B}$, $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{D}_1$, and $\mathcal{D}_2$ against the adaptive HKR security of* NIKE *such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{adaptive\text{-}DKR}}_{\mathcal{A},\mathsf{NIKE}'}(\lambda) &\leq \mathsf{Adv}^{\mathsf{adaptive\text{-}HKR}}_{\mathcal{B},\mathsf{NIKE}}(\lambda) + \mathsf{Adv}^{\mathsf{zk}}_{\mathcal{C}_1,\mathsf{NIZKPoK}_L}(\lambda) \\ &+ \mathsf{Adv}^{\mathsf{zk}}_{\mathcal{C}_2,\mathsf{NIZKPoK}_R}(\lambda) + \mathsf{Adv}^{\mathsf{pok}}_{\mathcal{D}_1,\mathsf{NIZKPoK}_L}(\lambda) + \mathsf{Adv}^{\mathsf{pok}}_{\mathcal{D}_2,\mathsf{NIZKPoK}_R}(\lambda) + q_{\mathsf{rev}}\varepsilon_{\mathsf{dist}}(\lambda),\end{aligned}$$

where q_{rev} is the number of $\mathcal{A}$'s $\mathcal{O}_{\mathsf{revL}}$ and $\mathcal{O}_{\mathsf{revR}}$ queries and $\varepsilon_{\mathsf{dist}}$ is the distribution correctness error of NIKE*.*

The changes we make to the proof of [29] are straightforward, but we recall it here for the sake of completeness.

Proof. The proof proceeds with a hybrid argument with the following games:
- G_0 is the real adaptive DKR security game.
- In G_1, the NIZKPoK in the public key of all honestly registered users is simulated with the Sim algorithm.
- In G_2, the game checks when the adversary registers a dishonest user id with public key (pk, π) whether $\mathsf{Ver}_{L/R}(\mathsf{crs}, (\mathsf{pp}, \mathsf{id}, \mathsf{pk}), \pi)$. If this returns 1, it extracts the witness with the $\mathsf{Extr}_{L/R}$ algorithm. If the witness is not valid, that is, $((\mathsf{pp}, \mathsf{id}, \mathsf{pk}), \mathsf{sk}) \notin \mathcal{R}_{L/R}$ for $\mathsf{sk} \leftarrow \mathsf{Extr}_{L/R}(\mathsf{crs}, \mathsf{td}, (\mathsf{pp}, \mathsf{id}, \mathsf{pk}), \mathsf{Prove})$, the adversary wins.
- In G_3, the reduction computes all $\mathcal{O}_{\mathsf{revL}}$ and $\mathcal{O}_{\mathsf{revR}}$ queries where one user is dishonest, and the dishonest user's public key contained a valid NIZKPoK, with the secret key of the dishonest user that it obtained with Extr.

Lemma 11 ($\mathsf{G}_0 \rightsquigarrow \mathsf{G}_1$)**.** *For every PPT adversary $\mathcal{A}$ there exist PPT adversaries $\mathcal{C}_1$, $\mathcal{C}_2$ such that*

$$\left|\Pr[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1]\right| \leq \mathsf{Adv}^{\mathsf{zk}}_{\mathcal{C}_1, \mathsf{NIZKPoK}_L}(\lambda) + \mathsf{Adv}^{\mathsf{zk}}_{\mathcal{C}_2, \mathsf{NIZKPoK}_R}(\lambda)$$

Proof. We first switch to an intermediate hybrid in which the NIZKPoKs of all honestly registered left users are simulated with Sim with a straightforward reduction to the zero-knowledge property of $\mathsf{NIZKPoK}_L$. □

Lemma 12 ($\mathsf{G}_1 \rightsquigarrow \mathsf{G}_2$)**.** *For every PPT adversary $\mathcal{A}$ there exist PPT adversaries $\mathcal{D}_1$, $\mathcal{D}_2$ such that*

$$\left|\Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1]\right| \leq \mathsf{Adv}^{\mathsf{pok}}_{\mathcal{D}_1, \mathsf{NIZKPoK}_L}(\lambda) + \mathsf{Adv}^{\mathsf{pok}}_{\mathcal{D}_2, \mathsf{NIZKPoK}_R}(\lambda).$$

Proof. The games G_1 and G_2 are identical, except that in G_2 there is an additional way for the adversary to win, namely by registering a user with a public key on whose proof the extraction procedure fails. We show that whenever that happens, a reduction can win the proof of knowledge in the presence of simulated proofs game of $\mathsf{NIZKPoK}_L$ or $\mathsf{NIZKPoK}_R$, depending on whether that was a registration query for a left or right user.

The reduction generates all simulated proofs with the $\mathcal{O}_{\mathsf{sim}}$ oracle of the game in Figure 10. Whenever the adversary registers a dishonest user id with public key (pk, π) and $\mathsf{Ver}_{L/R}(\mathsf{crs}, (\mathsf{pp}, \mathsf{id}, \mathsf{pk}), \pi)$ outputs 1, the reduction uses the $\mathcal{O}_{\mathsf{extr}}$ oracle to extract the corresponding secret key sk. If $((\mathsf{pp}, \mathsf{id}, \mathsf{pk}), \mathsf{sk}) \notin \mathcal{R}_{L/R}$, the reduction outputs $((\mathsf{pp}, \mathsf{id}, \mathsf{pk}), \pi)$ to win the proof of knowledge in the presence of simulated proofs game. Since the identity of a dishonestly registered user has to be different from every honest identity, it is guaranteed that the statement $(\mathsf{pp}, \mathsf{id}, \mathsf{pk})$ differs from every statement sent to the $\mathcal{O}_{\mathsf{sim}}$ oracle. □

Lemma 13 ($\mathsf{G}_2 \rightsquigarrow \mathsf{G}_3$)**.**

$$\left|\Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1]\right| \leq q_{\mathsf{rev}} \varepsilon_{\mathsf{dist}}(\lambda)$$

Proof. In all $\mathcal{O}_{\mathsf{revL}}$ and $\mathcal{O}_{\mathsf{revR}}$ with one dishonest user, in G_2 the honest user's secret key is used to compute the shared key, while in G_3 the dishonest user's secret key (obtained via NIZKPoK extraction) is used if that user was registered with a valid NIZKPoK. Otherwise, the shared key is $\perp$ anyway. The distribution correctness property guarantees that these shared keys have statistical distance at most $\varepsilon_{\mathsf{dist}}(\lambda)$. □

Lemma 14 (G_3). *For every PPT adversary $\mathcal{A}$ there exists a PPT adversary $\mathcal{B}$ with*

$$\Pr[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1] \leq \mathsf{Adv}^{\mathsf{adaptive\text{-}HKR}}_{\mathcal{B},\mathsf{NIKE}}(\lambda).$$

Proof. The reduction forwards all queries to the adaptive HKR security game of the underlying NIKE, discarding the NIZKPoKs from all public keys with two exceptions: The reduction handles all $\mathcal{O}_{\mathsf{regDL}}$ and $\mathcal{O}_{\mathsf{regDR}}$ queries itself as in G_3. It also handles all $\mathcal{O}_{\mathsf{revL}}$ and $\mathcal{O}_{\mathsf{revR}}$ with one dishonest user as in G_3. It can do so because it only needs the public key of the honestly registered user here. □
By combining Lemmata 11–14 we obtain the proof of Theorem 6. □

B Separation between light and adaptive security

In this section, we show a separation between light and (3–user) adaptive security for NIKE schemes with non-negligible correctness error. The separation assumes a perfectly correct[6] and lightly secure NIKE and a public key encryption scheme (PKE) with pseudorandomness against chosen-plaintext attacks (PR-CPA security). For convenience, in this section we use the usual definition of a NIKE where there is no distinction between "left"and "right" users. Concretely, we show that, with these building blocks, we can build for any efficiently computable function $p : \mathbb{N} \to [0, 1]$ a NIKE with correctness error $p(\lambda)$ that is lightly secure but can be attacked in the 3–user adaptive security setting with advantage $\approx p(\lambda)$. Thus, there cannot exist a generic transformation from light to adaptive security for NIKEs with a non-negligible correctness error. This is in contrast to the setting with a negligible correctness error [25].

B.1 Preliminaries

Definition 15 (PKE). *A* public key encryption (PKE) *scheme* PKE *for message space $\mathcal{M}$ with ciphertext space $\mathcal{C}$ consists of the following three probabilistic algorithms:*

- $\mathsf{Gen}(1^\lambda)$ *inputs an unary encoded security parameter λ and outputs a public and secret key pair* $(\mathsf{pk}, \mathsf{sk})$*,*
- $\mathsf{Enc}(\mathsf{pk}, \mathsf{msg})$ *inputs a public key* pk *and a message* $\mathsf{msg} \in \mathcal{M}$ *and outputs a ciphertext* $\mathsf{ct} \in \mathcal{C}$*,*
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$ *inputs a secret key* sk *and a ciphertext* $\mathsf{ct} \in \mathcal{C}$ *and outputs a message* $\mathsf{msg} \in \mathcal{M}$ *or* $\perp$ *(indicating a failure).*

[6] Statistical correctness would also suffice.

$\underline{\mathsf{Exp}^{\mathsf{pr\text{-}cpa}}_{\mathcal{A},\mathsf{PKE}=(\mathsf{Gen},\mathsf{Enc},\mathsf{Dec})}(\lambda)}$:	$\underline{\textsc{Chal}(\mathsf{msg}^\star)}$: //only one query
$(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ $b \xleftarrow{\$} \{0,1\}$ $b' \leftarrow \mathcal{A}^{\textsc{Chal}(\cdot)}(\mathsf{pk})$ **return** $b \stackrel{?}{=} b'$	$\mathsf{ct}_0^\star \leftarrow \mathsf{Enc}(\mathsf{pk},\mathsf{msg}^\star)$ $\mathsf{ct}_1^\star \xleftarrow{\$} \mathcal{C}$ **return** $\mathsf{ct}_b^\star$

Fig. 13. Security game for PR-CPA security of a PKE scheme.

For correctness we also require that for every $\mathsf{msg} \in \mathcal{M}$*, we have*

$$|\Pr[\mathsf{Dec}(\mathsf{sk},\mathsf{ct}) = \mathsf{msg} \mid (\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda), \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk},\mathsf{msg})]| \geq 1 - \mathsf{negl}(\lambda).$$

We also require that, given only the public key, one can efficiently sample uniformly from $\mathcal{C}$*. Note that* $\mathcal{C}$ *can be larger than the range of* $\mathsf{Enc}(\mathsf{pk},\cdot)$*.*

Definition 16 (PR-CPA security). *A PKE scheme* PKE *is* pseudorandom against chosen-plaintext attacks (PR-CPA) secure *if for all PPT adversaries* $\mathcal{A}$

$$\mathsf{Adv}^{\mathsf{pr\text{-}cpa}}_{\mathcal{A},\mathsf{PKE}}(\lambda) := 2|\Pr[\mathsf{Exp}^{\mathsf{xxx}}_{\mathcal{A},\mathsf{NIKE}}(\lambda) \Rightarrow 1] - 1/2|$$

is negligible. The game pr-cpa *is defined in Figure 13.*

B.2 Construction

For our separation result we need a NIKE NIKE with shared key space $\mathcal{K}$ and secret key space $\mathcal{SK}$ and a PR-CPA secure PKE with message space $\mathcal{M} \supseteq \mathcal{SK}$ and ciphertext space $\mathcal{C} = \mathcal{K}$. The former can always be achieved by concatenating multiple ciphertexts to enlarge the message space. The latter can be achieved by truncating or expanding the shared keys with a pseudorandom generator, if the shared key space is super-polynomially large. If this is not the case, we can combine multiple instances of the NIKE scheme to get another NIKE scheme with super-polynomially large shared key space. The construction is shown in Figure 14.

Theorem 7 (Correctness). *If the underlying NIKE* NIKE *is perfectly correct, then the NIKE* NIKE′ *from Figure 14 has correctness error at most* $2p(\lambda) - p(\lambda)^2$*.*

Proof. If two users compute the shared key for them, each of them will end up in case B in the SharedKey′ algorithm with probability $1 - p(\lambda)$. With probability $(1 - p(\lambda))^2 = 1 - 2p(\lambda) + p(\lambda)^2$ this will happen for both users, and thus they obtain the same shared key by the perfect correctness of the underlying NIKE NIKE. □

Theorem 8 (Light security). *If the underlying NIKE* NIKE *is lightly secure and the PKE* PKE *is PR–CPA secure, the NIKE* NIKE′ *is lightly secure. More*

$\underline{\mathsf{Setup}'(1^\lambda)}$:	$\underline{\mathsf{SharedKey}'(\mathsf{pp}, \mathsf{id}_1, \mathsf{pk}_1, \mathsf{id}_2, \mathsf{sk}_2)}$:
return $\mathsf{Setup}(1^\lambda)$	**parse** $\mathsf{pk}_1 =: (\mathsf{pk}_{\mathsf{NIKE},1}, \mathsf{pk}_{\mathsf{PKE},1})$
	parse $\mathsf{sk}_2 =: (\mathsf{sk}_{\mathsf{NIKE},2}, \mathsf{sk}_{\mathsf{PKE},2})$
$\underline{\mathsf{KeyGen}'(\mathsf{pp}, \mathsf{id})}$:	**with probability** $p(\lambda)$
$(\mathsf{pk}_{\mathsf{NIKE}}, \mathsf{sk}_{\mathsf{NIKE}}) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, \mathsf{id})$	$\quad$ **return** $\mathsf{Enc}(\mathsf{pk}_{\mathsf{PKE},1}, \mathsf{sk}_{\mathsf{NIKE},2})$ //case A
$(\mathsf{pk}_{\mathsf{PKE}}, \mathsf{sk}_{\mathsf{PKE}}) \leftarrow \mathsf{Gen}(1^\lambda)$	**else**
return $(\mathsf{pk} := (\mathsf{pk}_{\mathsf{NIKE},}, \mathsf{pk}_{\mathsf{PKE},}), \mathsf{sk} := (\mathsf{sk}_{\mathsf{NIKE},}, \mathsf{sk}_{\mathsf{PKE},}))$	$\quad$ **return** $\mathsf{SharedKey}(\mathsf{pp}, \mathsf{id}_1, \mathsf{pk}_{\mathsf{NIKE},1}, \mathsf{id}_2, \mathsf{sk}_{\mathsf{NIKE},2})$ //case B

Fig. 14. The $\mathsf{NIKE}' = (\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{SharedKey}')$ construction for the separation result. $\mathsf{NIKE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{SharedKey})$ is a perfectly correct and lightly secure NIKE and $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is a PR-CPA secure PKE. The function $p : \mathbb{N} \to [0, 1]$ can be any efficiently computable function and determines the correctness error.

precisely, for every PPT adversary $\mathcal{A}$ against the light security of NIKE' there exist PPT adversaries $\mathcal{B}$ and $\mathcal{C}$, each with almost the same run time as $\mathcal{A}$, such that

$$\mathsf{Adv}^{\mathsf{light}}_{\mathcal{A},\mathsf{NIKE}'}(\lambda) \leq \mathsf{Adv}^{\mathsf{light}}_{\mathcal{B},\mathsf{NIKE}}(\lambda) + 2\mathsf{Adv}^{\mathsf{pr\text{-}cpa}}_{\mathcal{C},\mathsf{PKE}}(\lambda)$$

Proof. The proof proceeds via a hybrid argument. The game G_0 is the real light security game for NIKE'. In the game G_1, the adversary gets in its only $\mathcal{O}_{\mathsf{test}}$ query, when case A occurs, a uniformly random ciphertext from the ciphertext space.

Lemma 15 ($\mathsf{G}_0 \rightsquigarrow \mathsf{G}_1$). *For every PPT adversary $\mathcal{A}$ there exists a PPT adversary $\mathcal{C}$ such that*

$$\left|\Pr[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1]\right| \leq 2\mathsf{Adv}^{\mathsf{pr\text{-}cpa}}_{\mathcal{C},\mathsf{PKE}}(\lambda).$$

Proof. The reduction guesses in the beginning which of the two registration queries is for the users whose secret key is used in the $\mathcal{O}_{\mathsf{test}}$ query. We will call this user the second user and the user whose public key is used in the $\mathcal{O}_{\mathsf{test}}$ query the first user (matching the argument order of the $\mathsf{SharedKey}$ algorithm). This guess will be correct with probability $1/2$.

For the first user, the reduction returns the public key from the PR-CPA game instead of generating it itself as the PKE component of its public key. Note that in the games G_0 and G_1 the secret keys for PKE are never used. All other key components are generated as in G_0. When in the $\mathcal{O}_{\mathsf{test}}$ query case A occurs, the reduction sends the NIKE secret key of the second user to the PR-CPA challenge oracle and uses the resulting ciphertext to answer the $\mathcal{O}_{\mathsf{test}}$ query.

If $b = 0$ was used in the PR-CPA security game, the reduction simulates G_0 and if $b = 1$ was used in the PR-CPA game, the reduction simulates G_1. Thus, an adversary capable of distinguishing these two games could be used to win the PR-CPA game for PKE. □

Lemma 16 (G_1). *For every PPT adversary $\mathcal{A}$ there exists a PPT adversary $\mathcal{B}$ such that*

$$\Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1] \leq \mathsf{Adv}^{\mathsf{light}}_{\mathcal{B},\mathsf{NIKE}}(\lambda).$$

Proof. The reduction forwards both $\mathcal{O}_{\mathsf{regH}}$ queries from the game G_1 to the light security game of the underlying NIKE NIKE and returns the resulting public keys together with a self-generated public key for PKE.

In the $\mathcal{O}_{\mathsf{test}}$ query, the reduction returns with probability $p(\lambda)$ a uniformly random ciphertext (which is also a uniformly random shared key), simulating case A. In this case, the adversary's view is independent of b and thus has advantage 0. The reduction outputs a random guess for the light security game of NIKE in this case.

In the remaining case, the reduction forwards the test query as the test query for light security game of NIKE and forwards the result, simulating case B. The reduction outputs whatever the adversary outputs and thus wins the light security game for NIKE when the adversary wins G_1. □

The proof of Theorem 8 follows by combining Lemmata 15 and 16. □

Theorem 9 (Adaptive HKR insecurity). *There exists a PPT adversary $\mathcal{A}$ against the 3–user adaptive HKR security of the NIKE NIKE' (assuming $|\mathcal{IDS}| \geq 3$) with advantage*

$$\mathsf{Adv}^{\text{3-user-adaptive-HKR}}_{\mathcal{A},\mathsf{NIKE}'}(\lambda) = 2p(\lambda) - p(\lambda)^2 - \frac{1}{|\mathcal{K}|},$$

where $\mathcal{K}$ is the shared key space of NIKE'. Note that by construction this is exponentially large.

Proof. The adversary $\mathcal{A}$ proceeds as shown in Figure 15.

We analyze the success probability of the adversary $\mathcal{A}$.

If $b = 1$, $K^\star$ is a uniformly random value, independent of all values returned in the other oracles and pp. Therefore, with probability $1 - 1/|\mathcal{K}|$, the adversary will output 1 and win the game.

If $b = 0$, with probability $2p(\lambda) - p(\lambda)^2$ at least one of the two $\mathcal{O}_{\mathsf{rev}}$ queries will return a "case A" shared key, i.e., the shared key is a ciphertext of the first or second user's secret key. By the correctness of PKE, the adversary returns "0" in this case and wins the game. In the remaining case, the adversary loses the game.

This leads to the following advantage of $\mathcal{A}$:

$$\begin{aligned}\mathsf{Adv}^{\text{3-user-adaptive-HKR}}_{\mathcal{A},\mathsf{NIKE}'}(\lambda) &= 2\left(\frac{1}{2}\cdot\left(2p(\lambda) - p(\lambda)^2\right) + \frac{1}{2}\cdot\left(1 - \frac{1}{|\mathcal{K}|}\right) - \frac{1}{2}\right)\\ &= 2p(\lambda) - p(\lambda)^2 - \frac{1}{|\mathcal{K}|}\end{aligned}$$

□

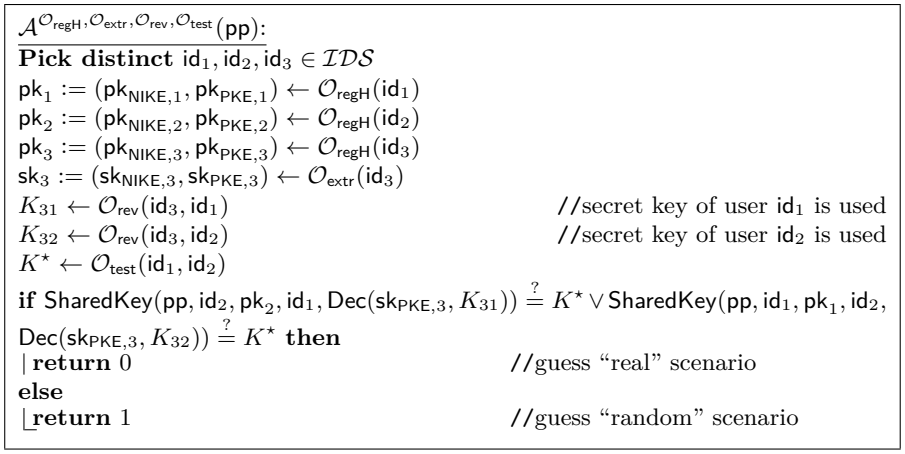

Fig. 15. The adversary $\mathcal{A}$ breaking the 3–user adaptive HKR security of the NIKE NIKE'.