Towards Tight Adaptive Security of Non-Interactive Key Exchange

Julia Hesse[1], Dennis Hofheinz[⋆2], Lisa Kohl[3], and Roman Langrehr[⋆2]

[1] IBM Research Europe - Zurich
jhs@zurich.ibm.com
[2] ETH Zurich
{hofheinz,roman.langrehr}@inf.ethz.ch
[3] Cryptology Group, CWI Amsterdam
lisa.kohl@cwi.nl

Abstract. We investigate the quality of security reductions for non-interactive key exchange (NIKE) schemes. Unlike for many other cryptographic building blocks (like public-key encryption, signatures, or zero-knowledge proofs), all known NIKE security reductions to date are non-tight, i.e., lose a factor of at least the number of users in the system. In that sense, NIKE forms a particularly elusive target for tight security reductions.

The main technical obstacle in achieving tightly secure NIKE schemes are adaptive corruptions. Hence, in this work, we explore security notions and schemes that lie between selective security and fully adaptive security. Concretely:

We exhibit a tradeoff between key size and reduction loss. We show that a tighter reduction can be bought by larger public and secret NIKE keys. Concretely, we present a simple NIKE scheme with a reduction loss of $O(N^2 \log(\nu)/\nu^2)$, and public and secret keys of $O(\nu)$ group elements, where N denotes the overall number of users in the system, and ν is a freely adjustable scheme parameter.

Our scheme achieves full adaptive security even against multiple "test queries" (i.e., adversarial challenges), but requires keys of size $O(N)$ to achieve (almost) tight security under the matrix Diffie-Hellman assumption. Still, already this simple scheme circumvents existing lower bounds.

We show that this tradeoff is inherent. We contrast the security of our simple scheme with a lower bound for all NIKE schemes in which shared keys can be expressed as an "inner product in the exponent". This result covers the original Diffie-Hellman NIKE scheme, as well as a large class of its variants, and in particular our simple scheme. Our lower bound gives a tradeoff between the "dimension" of any such scheme (which directly corresponds to key sizes in existing schemes), and the reduction quality. For $\nu = O(N)$, this shows our simple scheme and reduction optimal (up to a logarithmic factor).

We exhibit a tradeoff between security and key size for tight reductions. We show that it is possible to circumvent the inherent

[⋆] Supported in part by ERC CoG grant 724307.

tradeoff above by relaxing the desired security notion. Concretely, we consider the natural notion of semi-adaptive security, where the adversary has to commit to a single test query after seeing all public keys. As a feasibility result, we bring forward the first scheme that enjoys compact public keys *and* tight semi-adaptive security under the conjunction of the matrix Diffie-Hellman and learning with errors assumptions.

We believe that our results shed a new light on the role of adaptivity in NIKE security, and also illustrate the special role of NIKE when it comes to tight security reductions.

Keywords. Tight reductions, non-interactive key exchange, pairings, learning with errors.

1 Introduction

Non-interactive key exchange (NIKE). A non-interactive key exchange (NIKE) scheme assigns any two users P_i, P_j in a system a common shared key $K_{i,j}$. This assignment should happen without any communication, and be based only on a setup like a public-key infrastructure. A well-known example of a NIKE is the original Diffie-Hellman key exchange scheme [14], in which any party has a public key g^{x_i} with associated secret key x_i, and the shared key for parties with public keys g^{x_i}, g^{x_j} is computed as $K_{i,j} = g^{x_i x_j}$. For security, we would like that $K_{i,j}$ remains hidden to an outsider, i.e., without knowing any of the two involved secret keys.

NIKE schemes have been studied as an explicit cryptographic building block by Cash, Kiltz, and Shoup [9], followed by a more in-depth study of NIKE security notions and corresponding schemes by Freire, Hofheinz, Kiltz, and Paterson [19]. There are a variety of different NIKE schemes from various computational assumptions (e.g., [14, 9, 19, 5, 35, 25, 24]), and a number of NIKE applications including wireless networks [8], deniable authentication [15], and interactive key exchange [6].[4]

NIKE and tight security. One interesting particularity of NIKE schemes is the fact that it seems difficult to *tightly* reduce their security to a standard computational assumption. All known security reductions for NIKE schemes (against adaptive corruptions and to non-interactive assumptions in the standard model) lose a factor of at least N, the overall number of users in the system.[5] In fact, two works by Bader, Jager, Li, and Schäge [2] and Hesse, Hofheinz, and Kohl [25] give lower bounds (of $\mathcal{O}(N^2)$, resp. $\mathcal{O}(N)$) on the reduction loss of large classes of NIKE schemes and reductions.

[4] In this work, we focus on the public-key setting, i.e., we assume a public-key infrastructure. We note, however, that NIKE has also been considered in the identity-based setting [38, 17, 33].

[5] This means that we can currently only map NIKE adversaries with success probability ε and runtime t to adversaries on a suitable computational assumption with runtime $t' \approx t$ but success probability no more than $\varepsilon' \approx \varepsilon/N$.

This is quite remarkable, since for most other cryptographic building blocks (such as public-key encryption and digital signatures [26], zero-knowledge proofs [23], or interactive key exchange [1]), tight security proofs are known even in a multi-user setting. But apart from being a theoretical curiosity, this also means that currently, NIKE keysizes should be chosen rather conservatively, in order to account for a potential security loss in scenarios with a large number of users.

Meta-reductions, and what makes tight NIKE security particularly hard to achieve. The mentioned works [2, 25] already give an indication of what the main technical obstacle to a tight NIKE reduction is. Namely, they employ a meta-reduction [4] that turns any reduction that is "too successful" (i.e., suffers only from a low reduction loss) into a stand-alone problem solver. We give more details on this technique in our technical overview below. This meta-reduction technique has been applied also to other settings (like digital signatures [10], key encapsulation [2], and hierarchical identity-based encryption [30]), but it always hinges on one crucial requirement on the investigated scheme and reduction.

To explain this crucial requirement, assume for concreteness a given NIKE security reduction Λ that is "too successful". A meta-reduction requires that Λ is of a special form, namely that Λ essentially simulates the whole NIKE security experiment (including corruptions) for any NIKE adversary that is given in a black-box way. Furthermore, in this simulation, Λ must be "committed" early on to the secret state of this simulation, and in particular to all NIKE shared keys, even if these shared keys are not revealed during the simulation. The reason for this "committed" requirement will become clearer below, but intuitively it enables a "rewinding attack" on the reduction Λ itself.[6]

Now NIKE and other cryptographic primitives differ in this technical requirement for the applicability of meta-reductions. Namely, for primitives like public-key encryption (PKE), it is relatively easy to construct a reduction that is not committed to its secret state in the above sense. To see why this is the case, observe that in a PKE setting, different user secret keys or ciphertexts are not correlated: corrupting one user (or decrypting one ciphertext) gives no information about other users' secret keys (or the decryption of other ciphertexts). Hence, a reduction that answers corruption or decryption queries does not commit itself to, e.g., decryption of a challenge ciphertext in any way.

On the other hand, corrupting one party P_i in a NIKE scheme immediately reveals all shared keys $K_{i,j}$ that P_i has with other (yet-uncorrupted) parties P_j. This also determines the secret keys of such P_j to the extent that the $K_{i,j}$ computed with these not-yet-revealed keys are fixed. Hence, corrupting parties will gradually determine the secret state of a simulation in a NIKE reduction (i.e., the functionality of secret keys of yet-uncorrupted parties). This problem does not appear in, say, PKE or signature schemes, and circumventing this

[6] In a nutshell, the meta-reduction extracts enough shared keys from Λ to take the role of a successful adversary in a rewound Λ-instance. If Λ is "too successful", this causes Λ to solve the underlying computational problem with these extracted keys. Hence, Λ solves the underlying problem essentially by interacting with itself.

"committing" property in NIKE schemes currently seems to be out of reach of known techniques.

This work: beyond linear security loss. Motivated by this difficulty, in this work we examine this "committing" property closer for a general class of group-based NIKE schemes and slightly relaxed security notions. Specifically, for N again denoting the overall number of parties in the system we ask:

> *For which security notions can we obtain NIKE schemes with a security reduction to a standard assumption with a sublinear loss of $o(N)$?*

We obtain positive and negative results:

- We start off with a simple and intuitive "inner-product-based" NIKE scheme $\mathtt{NIKE}_{\mathsf{ip}}$ that enjoys full adaptive security and offers an interesting tradeoff between security loss and key sizes. Specifically, $\mathtt{NIKE}_{\mathsf{ip}}$ is parameterized by $\nu > 0$, has public and secret keys that comprise $\mathcal{O}(\nu)$ group elements, and a reduction loss of $\mathcal{O}(N^2 \log(\nu)/\nu^2)$ to the matrix Diffie-Hellman assumption [18] (a relaxation of the decision linear assumption) in pairing-friendly groups. In particular, it is possible to set $\nu = N$ to obtain a scheme with an (almost) tight reduction to a standard computational assumption, but which also suffers from large keys.
  While the scheme itself is not very efficient for large ν, it shows a conceptually simple way to conduct a "non-committing" reduction. Essentially, our reduction does not have the problematic "committing" property discussed above because each secret key contains enough entropy to be not quite determined by up to ν corruptions of arbitrary other users. This means that previous lower bounds [2, 25] do not apply to this scheme.
  We also note that our $\mathtt{NIKE}_{\mathsf{ip}}$ is the first to obtain tight security against multiple (i.e., up to ν) "test queries", i.e., adversarial challenges. This essentially means that the scheme guarantees the security of not only a single, but many shared keys even after a number of adaptive corruptions. While this property is implied with polynomial loss by security with respect to a single test query, previous reductions (including the previously "most tightly" secure scheme from [25]) did not consider multiple test queries.
  One can view our result also as a feasibility result about the possibility of tight *bounded* security (much like the notion of bounded chosen-ciphertext security for PKE schemes [12]) for NIKE schemes.
- Next, we demonstrate that this tradeoff between reduction loss and key sizes is to some extent inherent when trying to achieve adaptive NIKE security. Concretely, we show that a large class of group-based NIKE schemes (that includes the original Diffie-Hellman scheme, as well as variations such as the scheme from [25] and our $\mathtt{NIKE}_{\mathsf{ip}}$) must become "committed enough" after ν corruptions whenever keys are of size $o(\nu)$ group elements.
  Our result manifests the tradeoff between key sizes and reduction loss of $\mathtt{NIKE}_{\mathsf{ip}}$, and in fact for a large and natural class of NIKE schemes. We stress that the previous lower NIKE bounds [2, 25] do not offer similar tradeoffs, since they did not consider key sizes at all.

– Finally, motivated by the previous tradeoff, we investigate ways to achieve tight security with (asymptotically) compact keys by relaxing the desired security notion. We find a different tradeoff, and now trade security for tightness. Namely, we construct a NIKE $\mathtt{NIKE}_{\mathsf{sa}}$ with keys whose size do not depend on N, and with an (almost) tight security reduction that however only achieves semi-adaptive security. By "semi-adaptive security", we mean that an adversary is restricted not in the type or number of corruptions, but in the timing and number of test queries (i.e., challenges). Concretely, an adversary may not ask any test query after a certain, a-priori bounded number of ν corruptions (or queries for shared keys between honest parties) have been made. Semi-adaptive security interpolates between a mild form of selective security (in which an adversary has to commit in advance to the parties whose shared keys it wants to be challenged on) and full adaptive security.

  Our semi-adaptively secure $\mathtt{NIKE}_{\mathsf{sa}}$ uses $\mathtt{NIKE}_{\mathsf{ip}}$ above as a conceptually simple building block, and additionally relies on FHE techniques. Its security can be reduced (with logarithmic loss) to the conjunction of the matrix Diffie-Hellman problem and the learning with errors (LWE) problem [36].

  We believe that this result shows that even if we cannot achieve full adaptive security with compact keys and tightly, we are not limited to merely selective security. Due to lack of space, we present this contribution in full detail only in the full version of this work.

1.1 Technical overview

Setting. Formally, a NIKE is a tuple of algorithms ($\mathtt{Setup}, \mathtt{KeyGen}, \mathtt{SharedKey}$), where $\mathtt{Setup}$ generates public parameters, $\mathtt{KeyGen}$ on input of the public parameters returns a key pair $(\mathsf{pk}, \mathsf{sk})$, and $\mathtt{SharedKey}$ on input of the public parameters, a public key pk_i and a secret key sk_j returns a shared key $K_{i,j}$. Correctness requires that for all honestly generated key pairs we have $K_{i,j} = K_{j,i}$.

Security model. The simplest NIKE security notion to achieve is *selective security*, where the adversary commits to the key pair of users to be challenged (i.e. for which the adversary either receives the real shared key or a random key) *before* seeing any public key. To model realistic attack scenarios, what we would like to capture in the security notion is *fully adaptive security* (also called *CKS-heavy security* [19] after the inventors Cash, Kiltz and Shoup of the notion [9]). Here, the adversary can arbitrarily query oracles $\mathcal{O}_{\mathsf{extr}}$, $\mathcal{O}_{\mathsf{revH}}$ and $\mathcal{O}_{\mathsf{test}}$. $\mathcal{O}_{\mathsf{extr}}$ models the adversary's ability to corrupt a user and reveals the corresponding secret key and $\mathcal{O}_{\mathsf{revH}}$ models the ability of the adversary to observe shared keys in the system and reveals the shared keys between two users. Finally, the purpose of $\mathcal{O}_{\mathsf{test}}$ is to model that an adversary should still not be able to distinguish the (non-revealed) shared keys between any pair of uncorrupted users from random. More precisely, $\mathcal{O}_{\mathsf{test}}$ given a tuple of users either returns the real shared key between the users or a random key (depending on an initially flipped bit). Giving the adversary the power to ask corruption queries adaptively poses a challenge for the security reduction. Consider for example the Diffie-Hellman key exchange. There, public key/ secret key tuples are of the form (g^x, x) and a shared key is computed as $(g^{x_i})^{x_j} = (g^{x_j})^{x_i}$. Thus, the reduction either *knows* x – and therefore cannot make use of an adversary distinguishing shared keys involving x from random – or *does not know* x, and can therefore not answer with the secret key if the adversary decides to corrupt the user.

From selective to adaptive security with loss $\Omega(N^2)$. This can be solved by partitioning proofs, reducing the adaptive security to selective security. More precisely, the reduction guesses the "test query" of the adversary (i.e., the parties involved in the query that the adversary tries to distinguish from random) ahead of time and embeds the underlying challenge only in the two corresponding public keys. The problem of this approach is the security loss: With N overall users in the system, this strategy will only be successful with probability $1/N^2$. This means that the security guarantee decreases when the number of users in the system grows, which one has to account for by choosing larger concrete parameters (e.g. group sizes). Further, an upper bound on the number of users might not be known at the time of setup. In this paper we therefore aim for directly proving adaptive security.

Relaxing the security notion: Semi-adaptive security. We introduce the notion of *ν-semi-adaptive*, which lies in between selective and adaptive security: Here, the adversary has to ask *all* test queries within the first ν-corruptions (but can ask arbitrary extract and reveal queries later), where any user involved in a extract, reveal or test-query counts as one corruption. In the special case of 2-semi-adaptive security the adversary has to commit to a single test query after seeing all public keys.

Security with dishonest key registration (DKR). The security experiments described so far do not give the adversary the opportunity to register keys dishonestly, i.e., publish arbitrary public keys that are not necessarily in the image of `KeyGen`. This can of course occur in realistic scenarios and is ultimately the security notion to aim for. In this paper we restrict ourselves to security with honest key registration as described above, since the difficulty of constructing NIKEs with tight security occurs when going from selective to adaptive security, rather than going from HKR to DKR security. In fact, using standard methods one can *tightly* transform an HKR-secure NIKE into a DKR-secure one, basically by adding a simulation-sound proof of knowledge of the secret key to the public key (see e.g. [9, 25]).

Related work. We give a comparison of our result with previous work in Tables 1 and 2. In order to explain the challenges when constructing tightly secure NIKE, in the following we give a brief explanation of previous techniques used to give upper and lower bounds on tightly secure NIKE.

We first recall the *commitment problem* that occurs when proving security of the Diffie-Hellman NIKE. Namely, the reduction either *knows* a secret key or *does not know* a secret key, since each group element has a unique discrete logarithm. Building on the ideas put forward by Coron [11], Bader, Jager, Li,

	$\|pk\|$	Sec. model	$\mathcal{O}$(Sec. loss)	Assumption	Pairing
Diffie–Hellman [14]	$1 \times \mathbb{G}$	HKR	N^2	DDH	-
HPS-based [25]	$3 \times \mathbb{G}$	1-HKR	N	DDH	-
CKS08 [9]	$2 \times \mathbb{G}$	DKR	N^2	CDH (ROM)	-
FHKP13 [19]	$1 \times \mathbb{Z}_n$	DKR	N^2	Fact. (ROM)	-
FHKP13 [19]	$2 \times \mathbb{G} + 1 \times \mathbb{Z}_p$	DKR	N^2	DBDH	asymm.
HPS-based [25]	$12 \times \mathbb{G}$	1-DKR	N	DLIN	symm.
ν-dim NIKE$_{\mathsf{ip}}$ (Sec. 3)	$(\nu + 2) \times \mathbb{G}$	HKR	$(N/\nu)^2 \log \nu$	DLIN	symm.
N-dim NIKE$_{\mathsf{ip}}$ (Sec. 3)	$(N + 2) \times \mathbb{G}$	HKR	$\log N$	DLIN	symm.
ν-dim NIKE$_{\mathsf{sa}}$ (fullv.)	$\nu \cdot$ poly	ν-semi-ad.	$\log N$	DLIN, LWE	symm.
2-dim NIKE$_{\mathsf{sa}}$ (fullv.)	poly	semi-ad.	$\log N$	DLIN, LWE	symm.

Table 1. Comparison of existing NIKE schemes. $|pk|$ denotes the size of the public keys, measured in numbers of group elements and exponents. HKR and DKR denote fully adaptive security [19] with honest and dishonest key registrations (where 1-HKR/1-DKR refers to the corresponding notion in the single-test-query setting). N denotes the number of parties the adversary interacts with, $2 \leq \nu \leq N$ is arbitrary and poly is a polynomial independent of ν and N. Further, note that losses of the constructions from [9] and [19] stem from applying a generic transformation to level the security guarantees of compared schemes. DDH and CDH correspond to the decisional and computational Diffie-Hellman assumption, ROM stands for random oracle model and "Fact." for Factoring. DBDH stands for decisional bilinear Diffie-Hellman, DLIN for Decision Linear and LWE for Learning With Errors. Finally, note that in all cases DLIN can be replaced by the 2-Matrix Decision Diffie-Hellman assumption (MDDH). More generally, we can build on the k-MDDH assumption at the cost of increasing the public key size and security loss by a factor of k.

	Diffie-Hellman KE	HPS-based KE [25]	NIKE$_{\mathsf{ip}}$ **(Sec. 3)**
BJLS [2]	$\Omega(N^2)$	-	-
HHK [25]	$\Omega(N)$	$\Omega(N)$	-
This work (Sec. 4)	$\Omega(N)$	$\Omega(N)$	$\Omega(N/\nu)$

Table 2. Lower bounds on the security loss of NIKE. Here, the public keys of NIKE$_{\mathsf{ip}}$ are of size $O(\nu)$. Our lower bound only applies to the HPS-based NIKE [25] when instantiated with the decisional Diffie-Hellman-based hash proof system [13]. We note that (in settings where it applies) the lower bound of [25] gives better constants than ours. We highlight the best known lower bound for each construction in green.

and Schäge [2] presented a lower bound on the tightness of NIKE schemes for which public keys are *fully* committing to their secret keys and therefore their shared keys. Generally, the idea of a meta-reduction is to turn a "too successful" reduction into a stand-alone problem solver for the underlying (non-interactive) cryptographic assumption. The meta-reduction of Bader, Jager, Li, and Schäge [2] systematically rewinds the reduction Λ to run with all N^2 possible pairs of challenge users, arguing that in any run the reduction *either has to abort* or *indeed return the unique secret key.* Now, if the reduction does not abort with probability larger than $1/N^2$ (i.e., the reduction does not abort on at least 2 out of the $\binom{N}{2}$ possible runs), it follows that one can extract *all* secret keys from the reduction, and thereby perfectly simulate an external "perfect" adversary. (Note that for this to be true it is crucial that the reduction is limited to giving out *unique* secret keys, and therefore the shared keys are also unique.) Altogether, this shows that whenever the reduction is successful with probability larger than $1/N^2$, it could have solved the underlying problem itself. Since this is a contradiction to the hardness of the underlying assumption, it shows that the security loss of $\Omega(N^2)$ for Diffie-Hellman (and, more generally, NIKEs with "committing" public keys) is inherent.

Bypassing the commitment problem with semi-functional public keys. Hesse et al. [25] showed how to bypass the lower bound by allowing to switch to non-committing public keys. Essentially, their scheme allows to introduce "semi-functional" public keys which are computationally indistinguishable from public keys produced by KeyGen. This allows a reduction to escape the fully committed setting by introducing semi-functional public keys that do not necessarily fix the shared key with other (semi-functional or normal) public keys in the system. Their construction still suffers from a security loss of $\Omega(N)$, since their semi-functional public keys do not have secret keys and can thus be recognized upon corruption. Since a reduction needs to plant at least one such public key in order to escape full shared key commitment, a security loss of N is inherent. This lower bound on the security loss was formally shown in [25] for all schemes where normal public keys are committing *and* can be efficiently recognized given a corresponding secret key.

Considering weaker NIKE security notions. By allowing an arbitrary number of adaptive test queries but no corruptions, as was done e.g. in [9], tight security turns out easy to achieve. In fact, even the standard Diffie-Hellman key exchange can be shown (almost) tightly secure with respect to this notion, by simply embedding the underlying challenge into all public keys. Tight security (with a loss of factor $O(\log N)$) then follows by the re-randomizability of the decisional Diffie-Hellman assumption. However, going from test-query-only to adaptive security with corruption introduces a security loss of $\Omega(N^2)$. Since we are not aware of a tighter reduction for the scheme of [9] in the setting of adaptive security with corruptions, we do not consider their scheme tight in the sense of our paper.

Hesse et al. [25] consider a restriction of the above described security notion where the adversary is only allowed a single test query (but at any point of time). Since the generic reduction from the single-test-query setting to the multi-test-query introduces an overhead of $\Omega(N^2)$, in this paper we focus on the multi-test-query setting.

Technical idea 1: Overcoming binding public keys

Our construction. In this work we overcome the limitation of [25] with a NIKE scheme $\mathtt{NIKE}_{\mathsf{ip}}$ where both normal and semi-functional public keys have corresponding secret keys. Our construction is based on symmetric pairing groups. Let g be a group generator of the source group. We write $[x]$ for g^x and for a matrix $\mathbf{M} = (m)_{i,j}$ we write $[\mathbf{M}]$ for $([m])_{i,j}$. The public parameters of our NIKE are

$$\mathsf{pp} := ([\mathbf{D}], [\mathbf{MD}]),$$

where $\mathbf{D}$ is a uniformly random $(\nu + 2) \times 2$ matrix and $\mathbf{M}$ is a uniformly random symmetric $(\nu+2)\times(\nu+2)$ matrix. The parameter $\nu \in \mathbb{N}_{\geq 2}$ will become important in the security proof. An normal key pair is now generated as follows: We sample a uniformly random 2-dimensional vector $\mathbf{w}$ and set

$$\mathsf{pk} := [\mathbf{Dw}] \qquad \text{and} \qquad \mathsf{sk} := [\mathbf{MDw}].$$

The shared key between two users is the inner product of one user's public key and the other user's secret key, computed with the pairing. To see that correctness holds, let $(\mathsf{pk}_1 = [\mathbf{Dw}_1], \mathsf{sk}_1 = [\mathbf{MDw}_1])$ and $(\mathsf{pk}_2 = [\mathbf{Dw}_2], \mathsf{sk}_1 = [\mathbf{MDw}_2])$ be two honestly generated key pairs. Then

$$\begin{aligned}\mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}_1, \mathsf{sk}_2) &= e([\mathbf{w}_1^\top \mathbf{D}^\top], [\mathbf{MDw}_2]) = [\mathbf{w}_1^\top \mathbf{D}^\top \mathbf{MDw}_2]_T \\ &= [\mathbf{w}_2^\top \mathbf{D}^\top \mathbf{M}^\top \mathbf{Dw}_1]_T \overset{(*)}{=} [\mathbf{w}_2^\top \mathbf{D}^\top \mathbf{MDw}_1]_T \\ &= e([\mathbf{w}_2^\top \mathbf{D}^\top], [\mathbf{MDw}_1]) = \mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}_2, \mathsf{sk}_1).\end{aligned}$$

The equality $(*)$ uses the symmetry of $\mathbf{M}$.

One can interpret the public parameters by setting $(\mathbf{d}_1|\mathbf{d}_2) := \mathbf{D}$ as two exemplary key pairs

$$\mathsf{pp} := ((\mathsf{pk}_1 = [\mathbf{d}_1], \mathsf{sk}_1 = [\mathbf{Md}_1]), (\mathsf{pk}_2 = [\mathbf{d}_2], \mathsf{sk}_2 = [\mathbf{Md}_2])).$$

The user-generated keys are then random linear combinations of these exemplary key pairs. It is necessary to have at least two exemplary keys, because if the honest user keys would be linear combinations of just one exemplary key, one could use the pairing to check efficiently if a public key is in the subspace spanned by the exemplary public key. This would make it impossible for our reduction to use public keys that are not in the linear span of the exemplary public keys.

Semi-functional public keys with secret keys. To argue security, we have to introduce semi-functional public and secret keys. A semi-functional public key is $[\mathbf{u}]$ where $\mathbf{u}$ is chosen uniformly at random from the full space (instead of only the linear span of $\mathbf{D}$'s column vectors). Accordingly, the corresponding semi-functional secret key is $[\mathbf{Mu}]$.

The semi-functional key pairs are indistinguishable from the normal key pairs by the matrix decisional Diffie-Hellman (MDDH) assumption. It states that vectors (represented in a group) from a 2-dimensional subspace (i.e. our normal keys) are indistinguishable from uniformly random vectors (i.e. our semi-functional keys). The MDDH assumption is implied by the well-known 2-linear assumption [18]. Due to the random self-reducibility of the MDDH assumptions, this implication holds even for arbitrary many vectors with security loss only $\mathcal{O}(\log \nu)$.

The semi-functional keys have the desired "less committing" property. Indeed, note that with publishing the public parameters the reduction is not completely committed to the matrix $\mathbf{M}$, since $\mathbf{MD}$ contains only little information about $\mathbf{M}$. Now for each semi-functional public key $\mathbf{u}$ the corresponding semi-functional secret key $\mathbf{Mu}$ leaks some new information about $\mathbf{M}$ and after ν secret keys have been used, the reduction is completely committed to $\mathbf{M}$. If we would apply a suitable basis change transformation to $\mathbf{M}$ (such that the column vectors of $\mathbf{D}$, and the used semi-functional secret keys become unit vectors), each semi-functional secret key corresponds to a row (due to the symmetry also a column) of $\mathbf{M}$ and each shared key corresponds to one entry of the matrix, as depicted by Figure 1.

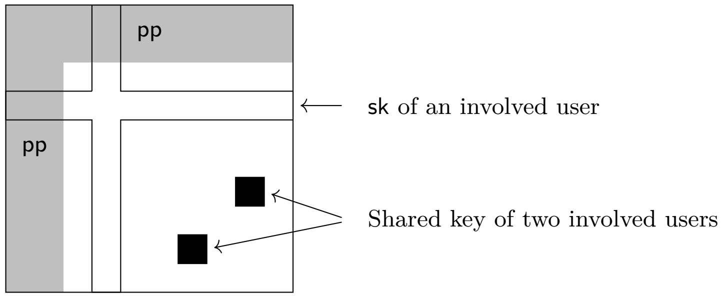

Figure 1. The symmetric matrix $\mathbf{M}$ in the basis where the column vectors of $\mathbf{D}$ and the semi-functional public keys of the involved users are the standard basis vectors. The normal secret keys (and shared keys where at least one user has a normal public key) live in the gray area. The pp can be seen as two key pairs and normal public and secret keys are linear combinations of these public and secret keys.

Since in our scheme there are secret keys for the semi-functional public keys, it circumvents the main bottleneck of the approach of [25]: Our reduction turns all public keys into semi-functional ones, and does not have to rely on any guessing argument. In contrast to [25], our semi-functional keys are committing with respect to normal keys. But, since we turn all keys to semi-functional, it is completely sufficient that semi-functional keys are not committing with respect to other semi-functional public keys. This approach is summarized in Table 3.

	pk_j normal	pk_j semi-functional
pk_i normal	committed committed	not committed committed
pk_i semi-functional	not committed committed	does not exist Up to ν users involved: not committed Beyond: committed

Table 3. Effect of all combinations of normal and semi-functional public keys on the shared key $K_{i,j}$ in the HPS-based NIKE [25] and our $\mathtt{NIKE}_{\mathsf{ip}}$.

Limiting the number of involved users. When ν semi-functional secret keys have been leaked, (i.e., they have been leaked through an $\mathcal{O}_{\mathsf{extr}}$ query or used to answer an $\mathcal{O}_{\mathsf{revH}}$ or $\mathcal{O}_{\mathsf{extr}}$ query,) the reduction is completely committed to $\mathbf{M}$. In this situation we can still argue, that each test query leaked one entry of the matrix $\mathbf{M}$ that was not revealed in any other query and therefore looks uniformly random to the adversary. However, any further leakage of another semi-functional secret key could potentially leak one of the test-query entries. Thus we have to limit the adversary to involve at most ν users in the security game, where a user counts as involved, when he appeared in at least one $\mathcal{O}_{\mathsf{extr}}$, $\mathcal{O}_{\mathsf{revH}}$, or $\mathcal{O}_{\mathsf{test}}$ query. (Users that have only been registered, i.e., only their public key was revealed, do not count as involved.)

We call the security notion that works like the adaptive security, but where the adversary is allowed to involve at most ν users, ν-bounded security. Even though this security notion is not very realistic, it is a helpful tool because it captures the level of adaptivity that $\mathtt{NIKE}_{\mathsf{ip}}$ can achieve and it implies full adaptive security with security loss only $\mathcal{O}((N/\nu)^2)$. Thus, in total $\mathtt{NIKE}_{\mathsf{ip}}$ can be proven adaptively secure with loss $\mathcal{O}((N/\nu)^2 \log \nu)$. This gives us a tradeoff between key size and tightness. The smaller we select the parameter ν, the smaller the size of the matrix $\mathbf{M}$. This gives us smaller keys, but the semi-functional keys will become committing earlier in the security game, leading to a larger security loss.

A curiosity of $\mathtt{NIKE}_{\mathsf{ip}}$ is that the roles of the public key and secret key are completely symmetric. That is, when all users swap their public and secret key, $\mathtt{NIKE}_{\mathsf{ip}}$ is still secure (and in the security proof we simply have to replace $\mathbf{M}$ by $\mathbf{M}^{-1}$).

Our scheme bypasses the lower bound of [25], because their lower bound requires, informally speaking, that whenever two key pairs look like valid to the adversary, the shared key between them is already determined by the public keys. This is not the case here: Two secret keys could differ by an entry of $\mathbf{M}$ that is unknown to adversary (thus both look like corresponding secret keys for the same public key), but, with a suitable public key of another valid key pair, this entry of $\mathbf{M}$ does not cancel out in the secret key computation and thus the two secret keys yield different shared keys.

Technical idea 2: Lower bound for large class of NIKEs

Inner-product NIKE and a new argument for committing reductions. To extend the existing results on lower bounds, we need to further broaden the class of NIKE schemes that the meta-reduction technique works for. The goal is to allow potential reductions to introduce keys that are less "committing" than in the previous bounds described above. Towards this goal, we observe that all DH-like NIKE schemes in the literature, including our $\mathtt{NIKE}_{\mathsf{ip}}$ described above, have the following joint property: public and secret keys can be represented as $\mathbb{Z}_q^d$-vectors $\mathbf{x}, \mathbf{y}$, and shared keys are computed as (an invertible function of) the inner product $\langle \mathbf{x}, \mathbf{y}' \rangle$. We call such NIKE schemes *d-dimensional ip-NIKE.* The Diffie-Hellman key exchange, for example, allows for key pair (g^x, x) to be written as tuple (x, x) of the same one-dimensional vector $x \in \mathbb{Z}_q$. Shared keys between vector tuples $(x_i, x_i), (x_j, x_j)$ are computed as $(g^{x_i})^{x_j} = g^{\langle x_i, x_j \rangle} = g^{\langle x_j, x_i \rangle} = (g^{x_j})^{x_i}$. Intuitively, using only one-dimensional vectors as in DH-KE means that public keys commit already to all shared keys. Vectors of higher dimensions, though, allow a reduction to encode more information, and eventually escape a setting where all shared keys are fixed. We can now formalize this intuition by exploiting linearity of the inner product. Namely, for a d-dimensional inner-product NIKE, a meta-reduction can create a fully committed setting in case vector dimensions are smaller than the number of users. For this, assume unique[7] public key vectors $\mathbf{x}_1, \ldots, \mathbf{x}_m$ of $m \approx N$ corrupted users and public key vectors $\mathbf{x}, \mathbf{x}'$ for yet uncorrupted $\mathsf{pk}, \mathsf{pk}'$. Let further $\mathbf{y}_1, \ldots, \mathbf{y}_m, \mathbf{y}, \mathbf{y}'$ denote corresponding secret key vectors. We stress that the meta-reduction is not able to compute any of these values, and we only use them to argue that the reduction is committed. If d is smaller than m, $\mathbf{x}$ lies in the span of the m other vectors with noticeable probability, yielding $\sum_{i=1}^{m} \beta_i \mathbf{x}_i = \mathbf{x}$ for a $\mathbb{Z}_q^m$-vector β. This already determines the (exponent of the) shared key $\langle \mathbf{x}, \mathbf{y}' \rangle$ between $\mathbf{x}$ and $\mathbf{x}'$ as a linear combination of the (exponents of) shared keys between each $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and $\mathbf{x}'$. To see this, we write

$$\langle \mathbf{x}, \mathbf{y}' \rangle = \langle \sum_{i=1}^{m} \beta_i \mathbf{x}_i, \mathbf{y}' \rangle = \sum_{i=1}^{m} \beta_i \langle \mathbf{x}_i, \mathbf{y}' \rangle = \sum_{i=1}^{m} \beta_i \langle \mathbf{y}_i, \mathbf{x}' \rangle,$$

where the latter equality follows from the correctness of the NIKE. Since the reduction already committed to the m shared key exponents $\langle \mathbf{y}_i, \mathbf{x}' \rangle_{i \in [m]}$ through corruptions of $\mathsf{pk}_1, \ldots, \mathsf{pk}_m$, we can conclude that the secret key between pk and pk' is fixed through its exponent $\langle \mathbf{x}, \mathbf{y}' \rangle$. We refer the reader to the "uniqueness lemma" (Lemma 5) for full details.

[7] For our results we require uniqueness of a corresponding *public* key vector given the public key, which holds for all DH-based schemes from the literature including our first NIKE.

A meta-reduction can exploit this committed setting by rewinding the reduction, a technique that was already used to prove the previous lower bounds [2, 25]. And indeed, we can show that any tight reduction must have key dimensions close to N, in order to avoid the linear dependencies described above that would result in commitment of all shared keys in the span. We now describe our meta-reduction and resulting lower bound in detail.

Our new lower bound. We are now ready to explain our lower bound. The general strategy of a meta-reduction is to first describe an inefficient "hypothetical" adversary $\mathcal{A}$ with success probability $\varepsilon_{\mathcal{A}}$, and then show that the hypothetical adversary can be efficiently simulated by rewinding the reduction except when some event "bad" occurs. Since the reduction has to work with the hypothetical adversary, this means that – except with probability $\Pr[\mathsf{bad}]$ – the reduction must also work with the simulated adversary, i.e., without external help. Since by assumption the reduction on its own cannot have more than negligible advantage in solving the underlying problem, this essentially shows that the success probability of the reduction can be upper bounded by $\Pr[\mathsf{bad}] \cdot \varepsilon_{\mathcal{A}} + \mathrm{negl}$ for a negligible function negl, i.e. lose a factor of $1/\Pr[\mathsf{bad}]$. For arguing that the simulated adversary perfectly simulates the hypothetical adversary we crucially rely on the uniqueness lemma, which ensures that all shared keys are fixed after the reduction gave out sufficiently many secret keys.

2-*step-adaptive security.* For proving our lower bound we introduce the 2-*step-adaptive* security notion, where an adversary after receiving the public keys can first ask the secret keys for an arbitrary large set D, and then has to commit to a challenge tuple of public keys (outside D). The adversary wins if after receiving the remaining secret keys (except the ones involved in the challenge tuple), it returns the shared key between the challenge parties. It is straightforward to see that adaptive security implies this weaker security notion, and therefore any lower bound on 2-step-adaptive security readily carries over to adaptive security.

The hypothetical adversary. The idea of the hypothetical adversary is to enforce uniqueness of the challenge shared key by choosing the set D in a suitable way. By the uniqueness lemma this can be achieved by choosing D such that the corresponding public key vectors span all public keys. (Note that if a NIKE is a d-dimensional inner-product NIKE, there always exist such a set of size at most d.) Once the shared key between the challenge key pairs is fixed, the adversary can simply brute-force any tuple of secret keys corresponding to the challenge public keys that are consistent with all secret keys in D, and use these to compute the shared key. Since the shared key is unique, the hypothetical adversary will always be successful.[8]

Simulating the hypothetical adversary. The problem in simulating the hypothetical adversary is that the following cannot be done efficiently:

[8] To capture adversaries with arbitrary success probability $\varepsilon_{\mathcal{A}}$, the hypothetical adversary can simply flip a biased coin and only output the shared key with probability $\varepsilon_{\mathcal{A}}$.

(1) *Extract* the public key vectors to find a spanning subset D, and
(2) Obtain the secret keys by *brute-force* to compute the challenge shared key.

The strategy of the meta-reduction to is therefore to:

(1) *Guess* a set D (and hope it is spanning), and
(2) Obtain secret keys by *rewinding the reduction* to compute the challenge shared key.

It turns out productive to choose $|D| \approx N/2$. The reason for this is as follows: On the one hand, for maximizing the probability that D is spanning, D should be chosen as large as possible. On the other hand, for extracting the secret keys from the reduction it is crucial that the reduction can be rewound while already being committed to the secret keys in D (since otherwise, the reduction could give out secret keys that are not consistent with the secret keys in D). In order to argue that the reduction either has to return valid secret keys for each $i \in [N] \backslash D$ or abort with high probability, we have to choose $[N] \backslash D$ large (essentially, the success probability will scale with $1 - 1/(N - |D|)$).

Success probability of the simulated adversary. Finally, the meta-reduction can compute the shared key with the help of this extracted secret keys. By the uniqueness lemma we obtain that this shared key is unique if both strategies of the meta-reduction are successful, i.e. if (1) D is indeed *spanning*, and (2) the reduction returns *valid and consistent secret keys* for both public keys involved in the challenge.[9] We can show that the event bad that either of these is not satisfied only occurs with probability in the order of d/N. This results in the following informal theorem:

Theorem (Lower bound): Any simple reduction from a non-interactive complexity assumption to the adaptive-security of a d-dimensional inner-product NIKE has to lose a factor in the order of $\Omega(N/d)$.

Our lower bound thus yields that $\mathtt{NIKE}_{\mathsf{ip}}$, which is a ν-dimensional ip-NIKE (see Definition 5 for the formal definition), with secret keys of size $O(\nu)$ has an inherent security loss of at least $\Omega(N/\nu)$. We contrast that with the security loss of our security proof for the core NIKE, which is $O((N/\nu)^2 \log \nu)$. Thus, for $\nu = N$ the security reduction that we give in Section 3 is essentially optimal. We give a comparison of our lower bound with others in Table 2.

Technical idea 3: Extension to "semi-adaptive" security

MOTIVATION: CONTROLLING ENTROPY LEAKAGE. The lower bound just presented appears to limit what we can prove about our first NIKE scheme $\mathtt{NIKE}_{\mathsf{ip}}$. Specifically, it appears that we require a large setting of ν (i.e., large keys) for

[9] Even though only one secret key is necessary to compute the shared key, we can only be sure that the reduction is committed to the shared key when given both secret keys, since the reduction could switch to a semi-functional public key (without valid secret key).

(almost) a tight security reduction. Taking a step back, the intuitive reason why we cannot obtain a better reduction is the following: every secret key revealed through a corruption query leaks entropy about the hidden matrix $\mathbf{M}$. This is intended, since in fact this fresh entropy is used to statistically blind shared keys. However, since the entropy contained in $\mathbf{M}$ is limited, this argument guarantees fresh entropy only for a bounded number of corruptions. After $\mathcal{O}(\nu)$ corruptions, $\mathbf{M}$ is fully determined, and any *additional* corruptions (or shared key or test queries) will result in (jointly) non-uniform shared keys. In particular, the security argument breaks down completely if more than $\mathcal{O}(\nu)$ corruptions are made, even if those are made after all shared key or test queries.

Our goal: ν-semi-adaptive security. We now set out to mitigate this limitation, and better *control* the entropy released through secret keys. We will unfortunately not be able to achieve full adaptive security with small keys. Instead, our goal will be a NIKE scheme with small keys, but in which more than $\mathcal{O}(\nu)$ corruptions are possible only *after* all test queries have been made. To be more concrete: we will achieve what we call ν-semi-adaptive security, which denotes security against the following type of attacks. An adversary may request up to ν corruptions, shared key, or test queries (in any combination). After that, any number of corruption or shared key queries, *but no test queries* are allowed. This notion is hence weaker than adaptive security, but also does allow for some degree of ("early") adaptivity. Like our basic scheme $\mathtt{NIKE}_{\mathsf{ip}}$, our ν-semi-adaptively secure scheme $\mathtt{NIKE}_{\mathsf{sa}}$ will have keys of size $\mathcal{O}(\nu)$ group elements, and its security reduction will be (almost) tight, i.e., only lose a factor of $\mathcal{O}(\log \nu)$.

As discussed above, our result can also be seen as a tradeoff between security and key size: the larger its keys are, the closer to (full) adaptive security the achieved security notion is. We reach full adaptive security only with large keys (of size $\mathcal{O}(N)$ group elements), but smaller keys still yield a less adaptively but (almost) tightly secure scheme.

Building block: non-interactive tag exchange. We now explain the main technical ideas of our semi-adaptively secure $\mathtt{NIKE}_{\mathsf{sa}}$. In a nutshell, we use $\mathtt{NIKE}_{\mathsf{ip}}$ as a *tag generator*, or as what we call a "non-interactive tag exchange" (NITE) scheme. A NITE is defined like a NIKE, except that (a) we call shared keys "tags" now, and (b) we require "ν-programmability" instead of indistinguishability for security. ν-programmability requires that there is a dedicated "programming algorithm" that allows to semi-adaptively program tags in the following way: given up to ν pairs of parties $(P_{i,1}, P_{i,2})$ and corresponding "target tags" T_i, output corresponding secret keys that yield T_i as tag between $P_{i,1}$ and $P_{i,2}$. This programming succeeds even *after* all public keys are fixed, and in an adaptive way (such that the T_i can be fixed one at a time, depending on all public keys and earlier T_i). For security, we require that this programming is not detectable, even given *all* secret keys (programmed or not). We can interpret $\mathtt{NIKE}_{\mathsf{ip}}$ as a NITE: shared keys are interpreted as tags, and programming works by adjusting $\mathbf{A}$

adaptively so that the desired tag values are computed.[10] Note that this process works only for programming up to $\mathcal{O}(\nu)$ tag values, since the entropy in $\mathbf{A}$ is limited. On the other hand, the notion of programmability also captures the security that $\mathtt{NIKE}_{\mathsf{ip}}$ achieves when eventually all secret keys are revealed.

Leveraging NITE programmability. The security of a NITE scheme requires programmable tags, but does not require "unopened" tags to remain hidden in any way (e.g., in the sense of NIKE indistinguishability). Hence, we cannot immediately use a NITE scheme as NIKE. Instead, our $\mathtt{NIKE}_{\mathsf{sa}}$ uses a NITE scheme to generate common (but not necessarily secret) shared tags for any two parties, who will then employ a "tag-based NIKE" (TNIKE) as a second stage to compute the actual NIKE shared keys. Analogously to tag-based encryption [28], a TNIKE is simply a NIKE in which shared key computation takes a tag as additional input. For correctness of $\mathtt{NIKE}_{\mathsf{sa}}$ in the usual sense, this tag should of course be the same for both parties.

Before describing a concrete TNIKE scheme, we describe its crucial abstract property: our TNIKE scheme has "punctured" secret keys, i.e., secret keys that allow to compute shared keys for all but one tag value. This puncturing point (i.e., the tag upon which shared key computation fails) is uniformly random, but not obvious from the corresponding public key. Similar puncturing techniques have been used as a technical tool to achieve adaptive security in various contexts before (e.g., [16, 32, 7, 34, 39, 27, 37]). In our security proof, we will program the tags output by the NITE scheme such that *all tags that refer to NIKE test queries will be programmed to be exactly the puncturing points of the corresponding secret keys*.[11] This programming is not detectable thanks to the NITE's security, and leads to a situation in which all test queries are randomized.

Our concrete construction. Armed with this intuition, we now give more details on our actual TNIKE construction. To illustrate the main ideas, we only describe a slightly simplified version of our construction for minimal ν, i.e., such that it achieves only a small degree of semi-adaptivity. The construction is based on the learning with errors (LWE) problem, and assumes public parameters $\mathsf{pp} := \mathbf{A} \xleftarrow{\$} \mathbb{Z}_p^{n \times m}$. A public key contains

$$\mathsf{pk} := (\mathbf{SA} + \mathbf{E}, \mathbf{V} = \mathbf{AU} + \tau\mathbf{G}),$$

where $\mathbf{S}$ is a random matrix, $\mathbf{E}$ and $\mathbf{U}$ are a "noise" matrices with small entries, $\mathbf{G}$ is the fixed "gadget matrix" of [31], and τ is the (uniformly random) tag at which the corresponding secret key will be punctured. Note that $\mathbf{V}$ is actually an encryption of τ under the fully homomorphic encryption (FHE) scheme of

[10] This is a slight oversimplification. In fact, programming requires to also make public keys semi-functional, as in the security proof of $\mathtt{NIKE}_{\mathsf{ip}}$ sketched above. Our formal programmability definition will allow for such adjustments during programming.

[11] This is again an oversimplification: for a particular choice of tag, one involved party P_i will not be able to compute the TNIKE shared key, while the other party P_j will be able to compute a shared key that depends on entropy in P_j's secret key.

Gentry, Sahai, and Waters [21].[12] The corresponding secret key is of the form

$$\mathsf{sk} := (\mathbf{S}, \mathbf{U}, \tau).$$

To compute the shared key between two users, assume a public key pk as above, a secret key $\mathsf{sk}' = (\mathbf{S}', \mathbf{U}', \tau')$ from another user, and a tag T. We first homomorphically and deterministically compute an FHE encryption $\mathbf{V}^\star = \mathbf{A}\mathbf{U}^\star + b\mathbf{G}$ from $\mathbf{V}$, where $b \in \{0, 1\}$ with $b = 1$ iff $\tau = T$. (Note that this really denotes the punctured point τ encrypted in $\mathbf{V}$, not the one from sk'. Hence, b is hidden at this point.) The corresponding shared key K is a rounded version of $\mathbf{S}'\mathbf{V}^\star$, i.e.,

$$K = \mathsf{round}(\mathbf{S}'\mathbf{V}^\star).$$

The other involved party, using $\mathsf{pk}' = (\mathbf{S}'\mathbf{A} + \mathbf{E}', \mathbf{V}')$ and $\mathsf{sk} = (\mathbf{S}, \mathbf{U}, \tau)$, computes the same shared key differently: it uses $\mathbf{U}$ to obtain the encryption random coins $\mathbf{U}^\star$ with $\mathbf{V}^\star = \mathbf{A}\mathbf{U}^\star + b\mathbf{G}$ (for b as above) and computes

$$K' = \mathsf{round}((\mathbf{S}'\mathbf{A} + \mathbf{E}')\mathbf{U}^\star) = \mathsf{round}(\mathbf{S}'\mathbf{A}\mathbf{U}^\star + \mathbf{E}'\mathbf{U}^\star) \overset{(*)}{=} \mathsf{round}(\mathbf{S}'\mathbf{A}\mathbf{U}^\star),$$

where $(*)$ holds with high probability for a suitable rounding function, since $\mathbf{E}'$ and $\mathbf{U}$ have small entries. Indeed, $K = K'$ whenever $T \neq \tau$ (so that $b = 0$ and $\mathbf{V}^\star = \mathbf{A}\mathbf{U}^\star$). But for $T = \tau$, the rounded value

$$\mathbf{S}'\mathbf{V}^\star = \mathbf{S}'\mathbf{U}^\star + \mathbf{S}'\mathbf{G}$$

in K contains the term $\mathbf{S}'\mathbf{G}$, which extracts randomness from $\mathbf{S}'$ (that, using a proper setup of $\mathbf{A}$, does not appear in pk'). Hence, the tag $T = \tau$ is special, in that K is randomized by entropy from $\mathbf{S}'$ only for this T. Note that the value K' does not contain this extra term, and so in fact does not satisfy $K' = K$ for $T = \tau$. Of course, since in "normal operation", tags are independently and uniformly random values, $T = \tau$ happens only with negligible probability, and this affects correctness of the scheme only negligibly.

Before going further, we note that this overview over our TNIKE scheme neglects a few things: we did not discuss suitable dimensions, the rounding function, or a suitable encoding of large tags τ. Besides, we did not discuss a generalization to larger values of ν (which require programming more values τ_i into each key). Finally, we did not discuss how both parties coordinate on their role in the computation of K (i.e., on whose $\mathbf{V}$ is used as a basis of computation). All of those questions have simple, albeit sometimes tedious technical answers, and we will discuss all of these issues inside.

The security of our construction. We now briefly sketch the proof of 1-semi-adaptive security of $\mathtt{NIKE}_{\mathsf{sa}}$, which is composed of our NITE and TNIKE schemes. So assume a 1-semi-adaptive adversary $\mathcal{A}$ that obtains all public keys, and then may ask a single test query. After this, and without loss of generality,

[12] In this overview, we neglect the fact that τ should be a small scalar. Our full scheme will actually encrypt τ bitwise.

$\mathcal{A}$ obtains all secret keys of parties not involved in that test query. We need to show that $\mathcal{A}$'s success in distinguishing between real and random answers is negligible, and, for a tight reduction, does not scale in the number of users. To do so, consider the following short sequence of game hops:

Game 0 is the original 1-semi-adaptive NIKE security game with a test query that is answered with the real shared key.

Game 1 changes how the tags for the test query are computed: here, the tag for the test query is adaptively programmed to be the puncturing point τ of the corresponding user. Note that the corresponding shared key can still be computed for $\mathcal{A}$ in the same way that K is computed above. By programmability of the NITE scheme, and using the security of the used FHE scheme, this change goes unnoticed by $\mathcal{A}$.[13]

Game 2 replaces the result of the test query with an independently chosen random shared key. This change is statistical, and can be justified with the observations above about hidden entropy in $\mathbf{S}'$.

We give full details of the proof (and additional discussion) in the full version of this paper.

2 Preliminaries

We use $x \stackrel{\$}{\leftarrow} S$ 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 $\mathcal{D}$. If $\mathcal{A}$ is a (probability) algorithm then we write $x \stackrel{\$}{\leftarrow} \mathcal{A}(b)$ to denote the random variable x outputted by $\mathcal{A}$ on input b. We use $\mathsf{Sym}_n(\mathbb{Z}_q)$ (for $n \in \mathbb{N}$, q prime) to denote the set of symmetric $n \times n$ over $\mathbb{Z}_q$. Initially, all partial maps (denoted by $f : A \dashrightarrow B$) are totally undefined in our games. We write $x[i]$ for the i-th bit of the binary representation of x. We write $(a, \_) := (x, y)$ and $(\_, b) := (x, y)$ to define $a := x$ and $b := y$, respectively. $T(\mathcal{A})$ denotes the running time of $\mathcal{A}$.

2.1 Pairing group assumptions

Throughout this paper, `SymGGen` denotes a probabilistic polynomial-time (PPT) algorithm that on input 1^λ returns a description $\mathcal{PG} := (\mathbb{G}, \mathbb{G}_T, q, g, e)$ of a symmetric pairing group, where $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order q for a λ-bit prime q. The group element g is a generator of $\mathbb{G}$. The function $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is an efficient computable (non-degenerated) bilinear map (i.e., a pairing). Define $g_T := e(g, g)$, which is a generator in $\mathbb{G}_T$.

We use the implicit representation of group elements as in [18]. For $s \in \{\epsilon, T\}$ and $a \in \mathbb{Z}_q$ define $[a]_s = ag_s \in \mathbb{G}_s$ as the implicit representation of a in $\mathbb{G}_s$. Similarly, for a matrix $\mathbf{A} = (a_{ij}) \in \mathbb{Z}_q^{n\times m}$ we define $[\mathbf{A}]_s$ as the implicit

[13] We note that to obtain *tight* security at this point, we will temporarily switch the used FHE scheme into a lossy mode of encryption [3, 22].

representation of $\mathbf{A}$ in $\mathbb{G}_s$. Note that it is efficient to compute $[\mathbf{AB}]_s$ given $([\mathbf{A}]_s, \mathbf{B})$ or $(\mathbf{A}, [\mathbf{B}]_s)$ with matching dimensions. Furthermore, $e([\mathbf{A}], [\mathbf{B}]) := [\mathbf{AB}]_T$ can be efficiently computed given $[\mathbf{A}]$ and $[\mathbf{B}]$ with the pairing e.

Many assumptions in paring groups can be expressed as matrix decisional Diffie-Hellman (MDDH) assumption [18]. For a definition of the (Q-fold) MDDH assumption, see [18] or the full version of this paper.

We use the Q-fold uniform matrix distribution, because the uniform distribution allows us to give a tight reduction to the standard 1-fold version, as shown by the following Lemma. Gay et al. already provided a tight reduction [20], but their proof is flawed[14] as pointed out by [29]. The proof can be found in the full version.

Lemma 1 (Random self-reducibility of $\mathcal{U}_{\ell,k}$-MDDH). *For every $\ell > k$ and every PPT adversary $\mathcal{A}$ there exists an adversary $\mathcal{B}$ with*

$$\mathsf{Adv}^{\mathsf{mddh},Q}_{\mathcal{A},\mathcal{U}_{\ell,k},\mathtt{SymGGen},s}(\lambda) \leq \left\lceil \log\left(\frac{\ell}{k}\right) \right\rceil k \left(\mathsf{Adv}^{\mathsf{mddh}}_{\mathcal{B},\mathcal{U}_k,\mathtt{SymGGen},s}(\lambda) + \frac{3}{q-1} \right),$$

where $\mathcal{PG} \leftarrow \mathtt{SymGGen}(1^\lambda)$ and $T(\mathcal{B}) \approx T(\mathcal{A}) + Q \cdot \mathrm{poly}(\lambda)$, where poly *is a polynomial independent of $\mathcal{A}$.*

2.2 Non-Interactive Key Exchange

Definition 1 (NIKE). *A* NIKE *scheme with identity space $\mathcal{IDS}$ and key space $\mathcal{K}$ consists of three polynomial-time algorithms* $(\mathtt{Setup}, \mathtt{KeyGen}, \mathtt{SharedKey})$*, where*

- $\mathtt{Setup}$ *is a randomized algorithm that takes the unary encoded security parameter 1^λ and samples public parameters* pp
- $\mathtt{KeyGen}$ *is a randomized algorithm that takes the parameters* pp *and an identity* $\mathsf{id} \in \mathcal{IDS}$ *and samples a key pair* $(\mathsf{pk}, \mathsf{sk})$
- $\mathtt{SharedKey}$ *is a deterministic algorithm that takes the parameters* pp*, an identity* id_1 *with its corresponding public key* pk_1 *and another identity* id_2 *with its corresponding secret key* sk_2 *and outputs a shared key K*

Definition 2 (Correctness). *We say that a NIKE* $(\mathtt{Setup}, \mathtt{KeyGen}, \mathtt{SharedKey})$ *for identity space $\mathcal{IDS}$ is* statistically correct*, if the correctness error*

$$\sup_{\mathsf{id}_1,\mathsf{id}_2 \in \mathcal{IDS}} \Pr[\mathtt{SharedKey}(\mathsf{pp}, \mathsf{id}_1, \mathsf{pk}_1, \mathsf{id}_2, \mathsf{sk}_2) \neq \mathtt{SharedKey}(\mathsf{pp}, \mathsf{id}_2, \mathsf{pk}_2, \mathsf{id}_1, \mathsf{sk}_1) \mid$$
$$\mathsf{pp} \leftarrow \mathtt{Setup}(1^\lambda), (\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathtt{KeyGen}(\mathsf{pp}, \mathsf{id}_1), (\mathsf{pk}_2, \mathsf{sk}_2) \leftarrow \mathtt{KeyGen}(\mathsf{pp}, \mathsf{id}_2)]$$

is negligible in λ. A NIKE is perfectly correct *if its correctness error is zero.*

[14] They correctly prove that $\mathcal{U}_{\ell,k}$-MDDH is tightly equivalent to $\mathcal{U}_k$-MDDH, but the proof can not show that Q-fold $\mathcal{U}_{\ell,k}$-MDDH is tightly equivalent to Q-fold $\mathcal{U}_k$-MDDH.

The standard security notion for a NIKE is adaptive security. It is a real-or-random notion that allows the adversary to register users, corrupt users, reveal shared keys and get challenged adaptively and arbitrary often. One could strengthen this security notion by giving the adversary an additional oracle that allows him to learn the shared keys of a user and a self-generated public key (dishonest key registration). This security notion can be achieved tightly with little overhead using the generic transformation of [25].

Our first construction in Section 3 achieves a weaker security notion, that we call ν-bounded security, for any $\nu \in \mathbb{N}_{\geq 2}$ with keys that grow linearly in ν. ν-bounded security is defined as adaptive security, but the adversary may only use up to ν users for corruption, revealing shared keys, and challenges. It can still register arbitrary many users and choose adaptively the subset of ν users for the other queries. While this security notion is arguably too weak for most realistic scenarios, it is useful because it implies adaptive security with security loss only $\mathcal{O}((N/\nu)^2)$.

In the full version we show how to strengthen our result to achieve ν-semi-adaptive security. This notion is defined like ν-bounded security, except that the adversary can still make $\mathcal{O}_{\mathsf{extr}}$, $\mathcal{O}_{\mathsf{revH}}$ (and $\mathcal{O}_{\mathsf{regH}}$) queries after exceeding the limit of ν involved users. Clearly, ν-semi-adaptive security tightly implies ν-bounded security, but is a more realistic security notion.

Definition 3 (Adaptive, ν-bounded, and ν-semi-adaptive security). *We say that a NIKE* $\mathtt{NIKE} = (\mathtt{Setup}, \mathtt{KeyGen}, \mathtt{SharedKey})$ *is ν-bounded, ν-semi-adaptively, or adaptively secure (for $\nu \geq 2$), if for all PPT adversaries $\mathcal{A}$*

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

is negligible for $\mathsf{xxx} = \nu\text{-}\mathsf{bounded}$*,* $\mathsf{xxx} = \nu\text{-}\mathsf{semi\text{-}adaptive}$ *or* $\mathsf{xxx} = \mathsf{adaptive}$*, respectively. The games* $\mathsf{Exp}^{\mathsf{xxx}}_{\mathcal{A},\mathtt{NIKE}}(\lambda)$ *are defined in Figure 2.*

The following argument shows that ν-bounded security implies adaptive security via a non-tight reduction. The reduction forwards the registration queries of up to ν users to the ν-bounded experiment and generates all other keys itself. Then the reduction can randomize the shared keys in the test queries between two users when both of their registrations have been forwarded. Via a hybrid argument, the reduction can randomize all test queries step by step. We defer the formal proof to the full version.

Lemma 2. *For every NIKE* $\mathtt{NIKE}$ *and every PPT adversary $\mathcal{A}$ against the adaptive security of* $\mathtt{NIKE}$*, there exists a PPT adversary $\mathcal{B}$ against the ν-bounded security for any $\nu \in \{2, \ldots, N\}$ with*

$$\mathsf{Adv}^{\mathsf{adaptive}}_{\mathcal{A},\mathtt{NIKE}}(\lambda) \leq \frac{1}{2}\left\lceil \frac{N}{\lfloor \nu/2 \rfloor} + 1 \right\rceil^2 \left(\mathsf{Adv}^{\nu\text{-}\mathsf{bounded}}_{\mathcal{B},\mathtt{NIKE}}(\lambda) + (N_{\mathsf{rev}} + N_{\mathsf{test}})\varepsilon_{\mathtt{NIKE}}(\lambda)\right)$$

and $T(\mathcal{B}) \approx T(\mathcal{A}) + N\,\mathrm{poly}(\lambda)$ for a polynomial poly *independent of $\mathcal{A}$, where N is the maximum number of users that $\mathcal{A}$ registers, N_{rev} and N_{test} are the maximum number of $\mathcal{A}$'s $\mathcal{O}_{\mathsf{revH}}$ and $\mathcal{O}_{\mathsf{test}}$ queries, respectively, and $\varepsilon_{\mathtt{NIKE}}(\lambda)$ is the correctness error of* $\mathtt{NIKE}$*.*

```
$\mathsf{Exp}^{\mathsf{adaptive}}_{\mathcal{A},\mathtt{NIKE}}(\lambda)$:
    $\mathsf{pp} \leftarrow \mathtt{Setup}(1^\lambda)$
    $Q_{\mathsf{extr}} := \emptyset;\ Q_{\mathsf{rev}} := \emptyset;\ Q_{\mathsf{test}} := \emptyset;$
    $\boxed{\dbox{Q_{\mathsf{inv}} := \emptyset}}$
    $\mathsf{pks} : \mathcal{IDS} \dashrightarrow \mathcal{PK}$
    $\mathsf{sks} : \mathcal{IDS} \dashrightarrow \mathcal{SK}$
    $b \xleftarrow{\$} \{0,1\}$
    $b^\star \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{regH}}(\cdot),\mathcal{O}_{\mathsf{extr}}(\cdot),\mathcal{O}_{\mathsf{revH}}(\cdot,\cdot),\mathcal{O}_{\mathsf{test}}(\cdot,\cdot)}(\mathsf{pp})$
    if $Q_{\mathsf{rev}} \cap Q_{\mathsf{test}} = \emptyset \wedge \nexists A \in Q_{\mathsf{test}} : A \cap Q_{\mathsf{extr}} = \emptyset$ $\dbox{\wedge |Q_{\mathsf{inv}}| \leq \nu}$ then
        return $b \overset{?}{=} b^\star$
    else
        return $0$

$\mathcal{O}_{\mathsf{regH}}(\mathsf{id} \in \mathcal{IDS})$:
    if $\mathsf{pks}(\mathsf{id}) \neq \perp$ then return $\perp$
    $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathtt{KeyGen}(\mathsf{pp}, \mathsf{id})$
    $\mathsf{pks}(\mathsf{id}) := \mathsf{pk};\ \mathsf{sks}(\mathsf{id}) := \mathsf{sk}$
    return $\mathsf{pk}$

$\mathcal{O}_{\mathsf{extr}}(\mathsf{id} \in \mathcal{IDS})$:
    if $\mathsf{sks}(\mathsf{id}) \neq \perp$ then
        $Q_{\mathsf{extr}} := Q_{\mathsf{extr}} \cup \{\mathsf{id}\};$
        $\boxed{\dbox{Q_{\mathsf{inv}} := Q_{\mathsf{inv}} \cup \{\mathsf{id}\}}}$
        return $\mathsf{sks}(\mathsf{id})$
    return $\perp$

$\mathcal{O}_{\mathsf{revH}}(\mathsf{id}_1 \in \mathcal{IDS}, \mathsf{id}_2 \in \mathcal{IDS})$:
    if $\mathsf{pks}(\mathsf{id}_1) \neq \perp \wedge \mathsf{sks}(\mathsf{id}_2) \neq \perp$ then
        $Q_{\mathsf{rev}} := Q_{\mathsf{rev}} \cup \{\{\mathsf{id}_1, \mathsf{id}_2\}\}$
        $\boxed{\dbox{Q_{\mathsf{inv}} := Q_{\mathsf{inv}} \cup \{\mathsf{id}_1, \mathsf{id}_2\}}}$
        $\mathsf{pk}_1 := \mathsf{pks}(\mathsf{id}_1);\ \mathsf{sk}_2 := \mathsf{sks}(\mathsf{id}_2)$
        return $\mathtt{SharedKey}(\mathsf{pp}, \mathsf{id}_1, \mathsf{pk}_1, \mathsf{id}_2, \mathsf{sk}_2)$
    return $\perp$

$\mathcal{O}_{\mathsf{test}}(\mathsf{id}_1^\star \in \mathcal{IDS}, \mathsf{id}_2^\star \in \mathcal{IDS})$:
    if $\mathsf{pks}(\mathsf{id}_1^\star) \neq \perp \wedge \mathsf{sks}(\mathsf{id}_2^\star) \neq \perp \wedge \{\mathsf{id}_1^\star, \mathsf{id}_2^\star\} \notin Q_{\mathsf{test}}$ then
        $Q_{\mathsf{test}} := Q_{\mathsf{test}} \cup \{\{\mathsf{id}_1^\star, \mathsf{id}_2^\star\}\}$
        $\boxed{\dbox{Q_{\mathsf{inv}} := Q_{\mathsf{inv}} \cup \{\mathsf{id}_1^\star, \mathsf{id}_2^\star\}}}$
        $\boxed{\textbf{if } |Q_{\mathsf{inv}}| > \nu \textbf{ then return } \perp}$
        $\mathsf{pk}_1 := \mathsf{pks}(\mathsf{id}_1^\star);\ \mathsf{sk}_2 := \mathsf{sks}(\mathsf{id}_2^\star)$
        $K_0^\star \leftarrow \mathtt{SharedKey}(\mathsf{pp}, \mathsf{id}_1^\star, \mathsf{pk}_1, \mathsf{id}_2^\star, \mathsf{sk}_2)$
        $K_1^\star \xleftarrow{\$} \mathcal{K}$
        return $K_b^\star$
    return $\perp$
```

Figure 2. Experiment for adaptive security, $\boxed{\nu\text{-semi-adaptive security}}$, and $\dbox{\nu\text{-bounded security}}$ of a NIKE scheme $\mathtt{NIKE}$ with identity space $\mathcal{IDS}$ and shared key space $\mathcal{K}$. $\mathcal{PK}$ denotes the public key space and $\mathcal{SK}$ denotes the secret key space. The partial maps pks and sks are initially totally undefined. The set Q_{inv} keeps track of all users involved in the game, that is, users that have been used in at least one $\mathcal{O}_{\mathsf{extr}}$, $\mathcal{O}_{\mathsf{revH}}$ or $\mathcal{O}_{\mathsf{test}}$ query (users that have been only registered but not used since then are not counted as involved users). In the ν-bounded experiment the adversary may involve at most ν users. In the ν-semi-adaptive experiment the adversary may not ask $\mathcal{O}_{\mathsf{test}}$ queries any more after more than ν users have been involved.

```
$\mathsf{Exp}^{\text{2-step-adaptive}}_{\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2,\mathcal{A}_3),N,\texttt{NIKE}}(\lambda)$:
  $\mathsf{pp} \xleftarrow{\$} \texttt{NIKE.Setup}(1^\lambda)$
  for $i \in \{1,\ldots,N\}$ do
    $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \texttt{NIKE.KeyGen}(\mathsf{pp})$
  $(st_1, D) \leftarrow \mathcal{A}_1(\mathsf{pp}, \mathsf{pk}_1, \ldots, \mathsf{pk}_N)$
  $(st_2, \{i^\star, j^\star\}) \leftarrow \mathcal{A}_2(st_1, (\mathsf{sk}_i)_{i \in D})$
  $K^\star \leftarrow \mathcal{A}_3(st_2, (\mathsf{sk}_i)_{i \in [N] \setminus (D \cup \{i^\star, j^\star\})})$
  if $K^\star = \texttt{NIKE.SharedKey}(\mathsf{pk}_{i^\star}, \mathsf{sk}_{j^\star})$ then
    return 1
  else
    return 0
```

Figure 3. Experiment for 2-step-adaptive security of a NIKE scheme `NIKE` with shared key space $\mathcal{K}$, for any $N \in \mathbb{N}$. If $i^\star \in D$ or $j^\star \in D$, the experiment aborts.

For our lower bound on tightness of adaptive NIKE security reductions, we define a relatively weak notion called 2-*step-adaptive security*. The experiment is depicted in Figure 3. It allows the adversary to see $n-2$ secret keys in two loads, and commit to one challenge pair of public keys after seeing the first load. Finally, 2-step-adaptive is the only notion in this paper which is computational, meaning that the adversary has to provide the shared key of the challenge pair in order to win the experiment. To ease presentation of our lower bound proof, the adversary is split into three stateful algorithms $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$.

Definition 4 (2-step-adaptive security). *A NIKE* $\texttt{NIKE} = (\texttt{Setup}, \texttt{KeyGen}, \texttt{SharedKey})$ *is* 2*-step-adaptively secure, if for all PPT adversaries* $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$

$$\mathsf{Adv}^{\text{2-step-adaptive}}_{\texttt{NIKE}}(\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3) := \Pr[\mathsf{Exp}^{\text{2-step-adaptive}}_{\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2,\mathcal{A}_3),N,\texttt{NIKE}}(\lambda) \to 1]$$

is negligible. The experiment is defined in Figure 3.

It is straightforward to verify that 2-step-adaptive security is implied by adaptive security. The relations between the security notions considered in this paper are shown in Figure 4.

3 An inner-product-based NIKE scheme

We present our NIKE $\texttt{NIKE}_{\mathsf{ip}}$ in Figure 5 that tightly achieves ν-bounded security for arbitrary $\nu \geq 2$. However, this comes at the price of public and secret key size $\mathcal{O}(\nu)$. Together with Lemma 2, this gives an adaptively secure NIKE with a trade-off between key size and security loss. The security can be based on any MDDH assumption in symmetric pairing groups. Correctness follows from

$$\texttt{SharedKey}(\mathsf{pp}, \mathsf{id}_i, \mathsf{pk}_i, \mathsf{id}_j, \mathsf{sk}_j) = e([(\mathbf{D}\mathbf{w}_i)^\top], [\mathbf{M}\mathbf{D}\mathbf{w}_j]) = [\mathbf{w}_i^\top \mathbf{D}^\top \mathbf{M}\mathbf{D}\mathbf{w}_j]_T = \\ [\mathbf{w}_j^\top \mathbf{D}^\top \mathbf{M}\mathbf{D}\mathbf{w}_i]_T = e([(\mathbf{D}\mathbf{w}_j)^\top], [\mathbf{M}\mathbf{D}\mathbf{w}_i]) = \texttt{SharedKey}(\mathsf{pp}, \mathsf{id}_j, \mathsf{pk}_j, \mathsf{id}_i, \mathsf{sk}_i).$$

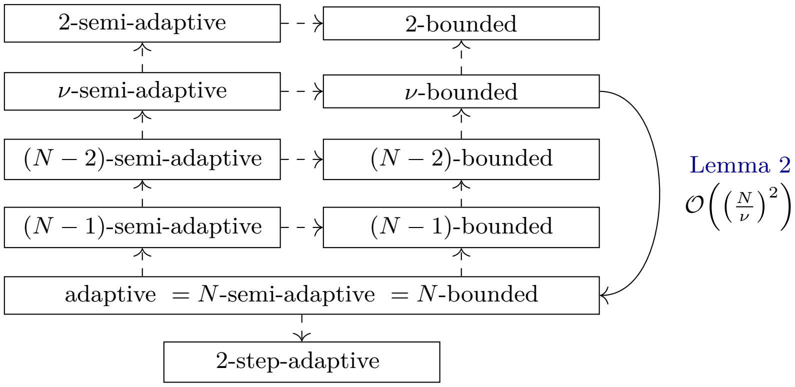

Figure 4. Relations between NIKE security notions for $\nu \in \{2, \ldots, N-2\}$ used in this paper. Dashed arrows mean “tightly implies” and solid arrows mean “implies with specified loss”.

```
Setup(1^λ):
    $\mathcal{G} := (\mathbb{G}, \mathbb{G}_T, q, g, e) \leftarrow \texttt{SymGGen}(1^\lambda)$
    $\mathbf{D} \leftarrow \mathcal{U}_{k+\nu,k}$
    $\mathbf{M} \xleftarrow{\$} \mathsf{Sym}_{k+\nu}(\mathbb{Z}_q)$
    return $\mathsf{pp} := (\mathcal{G}, [\mathbf{D}], [\mathbf{MD}])$

SharedKey(pp, id_1, pk_1, id_2, sk_2):
    return $e(\mathsf{pk}_1^\top, \mathsf{sk}_2)$

KeyGen(pp, id):
    parse $\mathsf{pp} =: (\mathcal{G}, [\mathbf{D}], [\mathbf{MD}])$
    $\mathbf{w} \xleftarrow{\$} \mathbb{Z}_q^k$
    $\mathsf{pk} := [\mathbf{Dw}]$
    $\mathsf{sk} := [\mathbf{MDw}]$
    return $(\mathsf{pk}, \mathsf{sk})$
```

Figure 5. Our inner-product-based $\texttt{NIKE}_{\mathsf{ip}}$ using symmetric pairing groups.

G_0 $\boxed{\mathsf{G}_1}$

```
$\mathsf{Exp}^{\nu\text{-bounded}}_{\mathcal{A},\mathtt{NIKE}}(\lambda)$:
    $\mathcal{G} := (\mathbb{G}, \mathbb{G}_T, q, g, e) \leftarrow \mathtt{SymGGen}(1^\lambda)$
    $\mathbf{D} \leftarrow \mathcal{U}_{k+\nu,k}$
    $\mathbf{M} \xleftarrow{\$} \mathsf{Sym}_{k+\nu}(\mathbb{Z}_q)$
    $\mathsf{pp} := (\mathcal{G}, [\mathbf{D}], [\mathbf{MD}])$
    $Q_{\mathsf{extr}} := \emptyset;\ Q_{\mathsf{rev}} := \emptyset;\ Q_{\mathsf{test}} := \emptyset$
    $Q_{\mathsf{inv}} := \emptyset$
    $\mathsf{pks} : \mathcal{IDS} \dashrightarrow \mathcal{PK}$
    $\mathsf{sks} : \mathcal{IDS} \dashrightarrow \mathcal{SK}$
    $b \xleftarrow{\$} \{0,1\}$
    $b^\star \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{regH}}(\cdot), \mathcal{O}_{\mathsf{extr}}(\cdot), \mathcal{O}_{\mathsf{revH}}(\cdot,\cdot), \mathcal{O}_{\mathsf{test}}(\cdot,\cdot)}(\mathsf{pp})$
    if $Q_{\mathsf{rev}} \cap Q_{\mathsf{test}} = \emptyset \wedge \nexists A \in Q_{\mathsf{test}} : A \cap Q_{\mathsf{extr}} = \emptyset \wedge |Q_{\mathsf{inv}}| \le \nu$ then
        return $b \overset{?}{=} b^\star$
    else
        return 0

$\mathcal{O}_{\mathsf{regH}}(\mathsf{id} \in \mathcal{IDS})$:
    if $\mathsf{pks}(\mathsf{id}_i) \neq \perp$ then return $\perp$
    $\mathbf{w} \xleftarrow{\$} \mathbb{Z}_q^k;\ \mathbf{u} := \mathbf{Dw}$
    $\boxed{\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^{k+\nu}}$
    $\mathsf{pk} := [\mathbf{u}]$
    $\mathsf{sk} := [\mathbf{Mu}]$
    $\mathsf{pks}(\mathsf{id}_i) := \mathsf{pk};\ \mathsf{sks}(\mathsf{id}_i) := \mathsf{sk}$
    return $\mathsf{pk}$
```

```
$\mathcal{O}_{\mathsf{extr}}(\mathsf{id} \in \mathcal{IDS})$:
    if $\mathsf{sks}(\mathsf{id}) \neq \perp$ then
        $Q_{\mathsf{extr}} := Q_{\mathsf{extr}} \cup \{\mathsf{id}\};\ Q_{\mathsf{inv}} := Q_{\mathsf{inv}} \cup \{\mathsf{id}\}$
        return $\mathsf{sks}(\mathsf{id})$
    return $\perp$

$\mathcal{O}_{\mathsf{revH}}(\mathsf{id}_1 \in \mathcal{IDS}, \mathsf{id}_2 \in \mathcal{IDS})$:
    if $\mathsf{pks}(\mathsf{id}_1) \neq \perp \wedge \mathsf{sks}(\mathsf{id}_2) \neq \perp$ then
        $Q_{\mathsf{rev}} := Q_{\mathsf{rev}} \cup \{\{\mathsf{id}_1, \mathsf{id}_2\}\}$
        $Q_{\mathsf{inv}} := Q_{\mathsf{inv}} \cup \{\mathsf{id}_1, \mathsf{id}_2\}$
        $\mathsf{pk}_1 := \mathsf{pks}(\mathsf{id}_1);\ \mathsf{sk}_2 := \mathsf{sks}(\mathsf{id}_2)$
        return $\mathtt{SharedKey}(\mathsf{pp}, \mathsf{id}_1, \mathsf{pk}_1, \mathsf{id}_2, \mathsf{sk}_2)$
    return $\perp$

$\mathcal{O}_{\mathsf{test}}(\mathsf{id}_1^\star \in \mathcal{IDS}, \mathsf{id}_2^\star \in \mathcal{IDS})$:
    if $\mathsf{pks}(\mathsf{id}_1^\star) \neq \perp \wedge \mathsf{sks}(\mathsf{id}_2^\star) \neq \perp \wedge \{\mathsf{id}_1^\star, \mathsf{id}_2^\star\} \notin Q_{\mathsf{test}}$ then
        $Q_{\mathsf{test}} := Q_{\mathsf{test}} \cup \{\{\mathsf{id}_1^\star, \mathsf{id}_2^\star\}\}$
        $Q_{\mathsf{inv}} := Q_{\mathsf{inv}} \cup \{\mathsf{id}_1^\star, \mathsf{id}_2^\star\}$
        $\mathsf{pk}_1 := \mathsf{pks}(\mathsf{id}_1^\star);\ \mathsf{sk}_2 := \mathsf{sks}(\mathsf{id}_2^\star)$
        $K_0^\star \leftarrow \mathtt{SharedKey}(\mathsf{pp}, \mathsf{id}_1^\star, \mathsf{pk}_1, \mathsf{id}_2^\star, \mathsf{sk}_2)$
        $K_1^\star \xleftarrow{\$} \mathcal{K}$
        return $K_b^\star$
    return $\perp$
```

Figure 6. Hybrids for the security proof of the NIKE from Figure 5. The partial maps pks and sks are initially totally undefined.

Theorem 1 (Security). *For every PPT adversary $\mathcal{A}$ against ν-bounded security of $\mathtt{NIKE}_{\mathsf{ip}}$, there exists a PPT adversary $\mathcal{B}$ solving $\mathcal{U}_k$-*MDDH

$$\mathsf{Adv}^{\nu\text{-bounded}}_{\mathcal{A},\mathtt{NIKE}_{\mathsf{ip}}}(\lambda) \le \left\lceil \log\left(1 + \frac{\nu}{k}\right) \right\rceil k \left(\mathsf{Adv}^{\mathsf{mddh}}_{\mathcal{B},\mathcal{U}_k,\mathtt{SymGGen},s}(\lambda) + \frac{1}{q-1} \right) + \frac{1}{q-1}$$

and $T(\mathcal{B}) \approx T(\mathcal{A}) + N\,\mathrm{poly}(\lambda)$ for a polynomial poly *independent of $\mathcal{A}$.*

The proof uses a hybrid argument with hybrids G_0 and G_1 given in Figure 6.

Lemma 3 **($\mathsf{G}_0 \rightsquigarrow \mathsf{G}_1$).** *For every PPT adversary $\mathcal{A}$ there exists an PPT adversary $\mathcal{B}$ such that*

$$\left|\Pr[\mathsf{G}_0^{\mathcal{A}}(\lambda) \Rightarrow 1] - \Pr[\mathsf{G}_1^{\mathcal{A}}(\lambda) \Rightarrow 1]\right| \leq \left\lceil \log\left(1 + \frac{\nu}{k}\right) \right\rceil k \left(\mathsf{Adv}^{\mathsf{mddh}}_{\mathcal{B},\mathcal{U}_k,\mathtt{SymGGen},s}(\lambda) + \frac{1}{q-1} \right)$$

and $T(\mathcal{B}) \approx T(\mathcal{A}) + N\,\mathrm{poly}(\lambda)$ *for a polynomial* poly *independent of* $\mathcal{A}$*.*

Proof. The real game G_0 uses normal keys (i.e. the public key is a chosen from the linear span of $\mathbf{D}$'s column vectors). In the game G_1 all keys are semi-functional (i.e. the public key is a chosen uniformly at random). Given an N-fold $\mathcal{U}_k$-MDDH challenge $[\mathbf{D}], ([\mathbf{u}_i])_{1 \leq i \leq N}$ one can simulate the games G_0 and G_1 efficiently when $\mathbf{A}$ in known over $\mathbb{Z}_q$. If the vectors $\mathbf{u}_i$ are sampled from the linear span of $\mathbf{D}$ this yields the game G_0 and if the vectors $\mathbf{u}_i$ are sampled uniformly random, this yields the game G_1. By reducing the N-fold $\mathcal{U}_k$-MDDH assumption to the $\mathcal{U}_k$-MDDH assumption with Lemma 1, the statement follows. □

Lemma 4 **(G_1).** *For every PPT adversary $\mathcal{A}$ there exists an PPT adversary $\mathcal{B}$ such that*

$$|\Pr[\mathsf{G}_1^{\mathcal{A}}(\lambda) \Rightarrow 1]| \leq \frac{1}{2} + \frac{1}{q-1}$$

and $T(\mathcal{B}) \approx T(\mathcal{A}) + N\,\mathrm{poly}(\lambda)$ *for a polynomial* poly *independent of* $\mathcal{A}$*.*

Proof. Without loss of generality, assume the adversary involves exactly ν users. Let us assume, that the column vectors of $\mathbf{D}$ and the public keys of all involved users (users with $\mathsf{id} \in Q_{\mathsf{inv}}$ at the end of the game) are linearly independent. Since all the public keys are uniformly random vectors in G_1, this happens with probability at least $1 - 1/(q-1)$. Initially, the symmetric bilinear form $B(\mathbf{v}, \mathbf{w}) := \mathbf{v}^\top \mathbf{M} \mathbf{w}$ is uniformly random to the adversary. Now suppose the adversary makes a $\mathcal{O}_{\mathsf{test}}$ query with two users $\mathsf{id}_1^\star$ and $\mathsf{id}_2^\star$. Let $[\mathbf{u}_1^\star]$ and $[\mathbf{u}_2^\star]$ be the public keys of $\mathsf{id}_1^\star$ and $\mathsf{id}_2^\star$, respectively and let $[\mathbf{u}_1], \ldots, [\mathbf{u}_{\nu-2}]$ be the public keys of all other users. We show the shared key between the tested users, $[B(\mathbf{u}_1^\star, \mathbf{u}_2^\star)]_T$, is statistically independent of all the other information the adversary learns about B during the game.

Assume all $\mathcal{O}_{\mathsf{revH}}$ queries and other $\mathcal{O}_{\mathsf{test}}$ queries involve at least one involved user different to $\{\mathsf{id}_1^\star, \mathsf{id}_2^\star\}$, because if the adversary makes a $\mathcal{O}_{\mathsf{revH}}$ query with $\mathsf{id}_1^\star$ and $\mathsf{id}_2^\star$ the adversary has lost trivially and a duplicated $\mathcal{O}_{\mathsf{test}}$ query would only return $\perp$. Thus these queries can not reveal any information about B that is not revealed by $\mathbf{M}(\mathbf{D}|\mathbf{u}_1|\cdots|\mathbf{u}_{\nu-2})$. The public parameters only reveal $\mathbf{MD}$ and any $\mathcal{O}_{\mathsf{extr}}$ query only reveals $\mathbf{M}\mathbf{u}_i$ for an $i \in \{1, \ldots, \nu-2\}$. In total the adversary learns from all queries except the analyzed $\mathcal{O}_{\mathsf{test}}$ query only $\mathbf{M}(\mathbf{D}|\mathbf{u}_1|\cdots|\mathbf{u}_{\nu-2})$. Since the column vectors of $\mathbf{D}$ together with $\{\mathbf{u}_1, \ldots, \mathbf{u}_{\nu-2}, \mathbf{u}_1^\star, \mathbf{u}_2^\star\}$ are assumed to be a linear independent set, $B(\mathbf{u}_1^\star, \mathbf{u}_2^\star) = (\mathbf{u}_1^\star)^\top \mathbf{M} \mathbf{u}_2^\star$ is uniformly random given $\mathbf{M}(\mathbf{D}|\mathbf{u}_1|\cdots|\mathbf{u}_{\nu-2})$. We can apply the above argument to each of the adversaries $\mathcal{O}_{\mathsf{test}}$ queries. Consequently, the adversaries advantage in G_1 is 0 under the stated assumptions. □

Proof (of Theorem 1). Combining Lemmata 3 and 4 proves Theorem 1. □

Corollary 1. *The NIKE* $\texttt{NIKE}_{\mathsf{ip}}$ *is adaptively secure with a security loss of* $\mathcal{O}((N/\nu)^2 \log \nu)$ *under the decision linear (DLIN) assumption.*

Proof. The DLIN assumption implies the $\mathcal{U}_k$-MDDH for $k \geq 2$ [18], so we can set $k = 2$. The NIKE then achieves ν-bounded security with loss $\mathcal{O}(\log \nu)$ by Theorem 1. With Lemma 2 we can achieve adaptive security by increasing the security loss by a factor of $\mathcal{O}((N/\nu)^2)$. □

4 Lower bound

In this section, we show that for all NIKEs that follow a special structure, there is an in some sense inherent trade-off between key sizes and quality of the reduction. Compared to previous lower bounds on NIKE reductions [2, 25], we do not make the generic assumption that pairs of public keys already determine the corresponding shared key. Instead, we leave room for a reduction to adaptively determine secret keys upon corruptions, as long as they follow the inner product structure we require. We use the notions of Non-Interactive Complexity Assumption (NICA, [2], Def. 4 and 5) and Simple Reductions [2], Def. 6 and 7, where the latter is adapted to the reduction breaking our 2-step-adaptive security in a straightforward way.

We now formalize the notion of *inner-product NIKE*. The intuition behind this definition is as follows. Basically, we require an (inefficient) algorithms $\texttt{Extract}$ that can be used to extract the key vectors $(\mathbf{x}, \mathbf{y})$ from a valid pair of public and secret keys, such that $(\mathbf{x}, \mathbf{y})$ can be used to compute the shared key as an inner product. As we will see in the next section, this inner-product structure will enforce uniqueness of the shared keys, as soon as sufficiently many secret keys are fixed. The verification algorithm is necessary to ensure that the reduction can only give out public and secret keys that satisfy some structural requirements (e.g. be of the right form and dimension). The public extraction algorithm $\texttt{PExtract}$ together with the *binding* requirement is necessary to ensure that the public keys are committing to the vector $\mathbf{x}$, even before the secret keys are given out. Finally, we therefore require the function f to be invertible, since it will be crucial that there is a one-to-one correspondence between the inner product and the shared key algorithm (note though that the inverse does not have to be efficiently computable).

Definition 5 (Inner-product NIKE). *Let* $p \in \mathbb{N}$ *a prime. We say a NIKE* $\texttt{NIKE} = (\texttt{Setup}, \texttt{KeyGen}, \texttt{SharedKey})$ *is a d-dimensional* inner-product NIKE (ip-NIKE) *over* $\mathbb{Z}_p$*, if there exists:*

- *a PPT algorithm* $\texttt{Ver}$ *taking as input public parameters* pp*, and a key pair* $(\mathsf{pk}, \mathsf{sk})$ *and returning a bit* $b \in \{0, 1\}$*,*
- *an (inefficient) deterministic extractor* $\texttt{Extract}$ *that takes as input public parameters* pp*, and a key pair* $(\mathsf{pk}, \mathsf{sk})$ *and returns a tuple* $(\mathbf{x}, \mathbf{y}) \in \mathbb{Z}_p^d \times \mathbb{Z}_p^d$*,*

- *an (inefficient) deterministic extractor* $\mathtt{PExtract}$ *taking as input* pp *and* pk *and returning a vector* $\mathbf{x} \in \mathbb{Z}_p^d$*,*
- *and a function* f *taking as input public parameters* pp *and an element* $z \in \mathbb{Z}_p$ *and returning a element in the image of* $\mathtt{NIKE.SharedKey}$*.*

such that the following properties hold.

(i) Verifiable keys. For all $(\mathsf{pk}, \mathsf{sk})$ *in the image of* $\mathtt{NIKE.KeyGen}(\mathsf{pp})$ *it holds* $\mathtt{Ver}(\mathsf{pp}, \mathsf{pk}, \mathsf{sk}) = 1$*.*

(ii) Ip-computable shared keys. For all public parameters pp*, and all key pairs* $(\mathsf{pk}, \mathsf{sk}), (\mathsf{pk}', \mathsf{sk}')$ *with* $\mathtt{Ver}(\mathsf{pp}, \mathsf{pk}, \mathsf{sk}) = \mathtt{Ver}(\mathsf{pp}, \mathsf{pk}', \mathsf{sk}') = 1$*, for* $(\mathbf{x}, \mathbf{y}) \leftarrow \mathtt{Extract}(\mathsf{pp}, \mathsf{pk}, \mathsf{sk})$ *and* $(\mathbf{x}', \mathbf{y}') \leftarrow \mathtt{Extract}(\mathsf{pp}, \mathsf{pk}', \mathsf{sk}')$ *it holds*

$$\mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}, \mathsf{sk}') = f(\mathsf{pp}, \langle \mathbf{x}, \mathbf{y}' \rangle).$$

(iii) Binding public keys. For all $(\mathsf{pk}, \mathsf{sk})$ *with* $\mathtt{Ver}(\mathsf{pk}, \mathsf{sk}) = 1$ *for* $\mathbf{x} \leftarrow \mathtt{PExtract}(\mathsf{pp}, \mathsf{pk})$ *and* $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) \leftarrow \mathtt{Extract}(\mathsf{pp}, \mathsf{pk}, \mathsf{sk})$ *it holds* $\mathbf{x} = \tilde{\mathbf{x}}$*.*

(iv) Invertibility of f*. The induced function* $f_{\mathsf{pp}} = f(\mathsf{pp}, \cdot)$ *is injective with inverse* f_{pp}^{-1}*.*

We call a key pair $(\mathsf{pk}, \mathsf{sk})$ *with* $\mathtt{Ver}(\mathsf{pp}, \mathsf{pk}, \mathsf{sk}) = 1$ valid*. By the* ip-dimension *of an inner-product NIKE* $\mathtt{NIKE}$*, we denote the minimal dimension* d*, such that* $\mathtt{NIKE}$ *satisfies the definition of* d*-dimensional inner-product NIKE.*

4.1 Lower bound for inner-product NIKEs

In order to show our lower bound, we first prove that after giving out sufficiently many public key/secret key pairs for a NIKE that satisfies the inner-product form, the reduction is committed to all shared keys. More precisely, let $\{\mathsf{pk}_i\}_{i \in I}$ such that the corresponding vectors $\mathbf{x}_i \leftarrow \mathtt{PExtract}(\mathsf{pp}, \mathsf{pk}_i)$ span the whole space $\mathbb{Z}_p^d$.[15] We further fix secret keys for pk_i for all $i \in I$ such that $(\mathsf{pk}_i, \mathsf{sk}_i)$ are all valid and *consistent* with each other, meaning that for all i, j, we have $\mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}_i, \mathsf{sk}_j) = \mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}_j, \mathsf{sk}_i)$. We will show that such a set of keys fix not only the shared keys between the public keys within $\{\mathsf{pk}_i\}_{i \in I}$, but the shared keys between *all* possible valid public keys in the system. Therefore, if d is significantly smaller than N, then with high probability, any large enough random subset of key pairs will span the whole space of public keys and thereby already fix all shared keys (computed as inner products) in the system. This will be crucial in our meta reduction, where we use uniqueness of shared keys in order to efficiently simulate a hypothetical perfect adversary, thereby essentially showing that the reduction either has to abort with large probability or would be able to solve the underlying problem itself.

Lemma 5 (Unique shared keys for ip-NIKEs). *Let* $d, p, \lambda \in \mathbb{N}$ *and* $\mathtt{NIKE} = (\mathtt{Setup}, \mathtt{KeyGen}, \mathtt{SharedKey}, \mathtt{Ver}, \mathtt{Extract}, \mathtt{PExtract})$ *a* d*-dimensional ip-NIKE over* $\mathbb{Z}_p$*. Let* $I \subset [N]$*. Let* $\mathsf{pp} \in \{0,1\}^\star$ *and let* $\{(\mathsf{pk}_i, \mathsf{sk}_i)\}_{i \in I}, (\mathsf{pk}, \mathsf{sk}), (\mathsf{pk}', \mathsf{sk}')$ *be such that:*

[15] For simplicity of this explanation we assume for now that such a set of keys exists, but stress that our results do not rely on it.

(a) All key pairs are valid: $\mathtt{Ver}(\mathsf{pp}, \mathsf{pk}_i, \mathsf{sk}_i) = 1$ *for all* $i \in I$ *and* $\mathtt{Ver}(\mathsf{pp}, (\mathsf{pk}, \mathsf{sk})) = \mathtt{Ver}(\mathsf{pp}, \mathsf{pk}', \mathsf{sk}') = 1$.

(b) The key pairs in I are pairwise consistent:
$\mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}_i, \mathsf{sk}_j) = \mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}_j, \mathsf{sk}_i)$ *for all* $i, j \in I$ *with* $i \neq j$.

(c) The key pairs $(\mathsf{pk}, \mathsf{sk})$ and $(\mathsf{pk}', \mathsf{sk}')$ are consistent with the key pairs in I: $\mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}, \mathsf{sk}_i) = \mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}_i, \mathsf{sk})$ *and* $\mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}', \mathsf{sk}_i) = \mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}_i, \mathsf{sk}')$ *for all* $i \in I$.

(d) The public keys $\mathsf{pk}, \mathsf{pk}'$ are "in the span" of $\{\mathsf{pk}_i\}_{i \in I}$: *for* $\mathbf{x}_i \leftarrow \mathtt{PExtract}(\mathsf{pp}, \mathsf{pk}_i)$ *and* $\mathbf{x} \leftarrow \mathtt{PExtract}(\mathsf{pp}, \mathsf{pk})$, $\mathbf{x}' \leftarrow \mathtt{PExtract}(\mathsf{pp}, \mathsf{pk}')$ *it holds that* $\mathbf{x}, \mathbf{x}'$ *are in the span of* $\{\mathbf{x}_i\}_{i \in I}$.

Then it holds that $\mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}, \mathsf{sk}') = \mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}', \mathsf{sk})$. *In other words, the shared key between the users holding* pk' *and* pk *is* consistent.

Proof. Let $(\mathbf{x}_i, \mathbf{y}_i) \leftarrow \mathtt{Extract}(\mathsf{pp}, \mathsf{pk}_i, \mathsf{sk}_i)$, $(\mathbf{x}, \mathbf{y}) \leftarrow \mathtt{Extract}(\mathsf{pp}, \mathsf{pk}, \mathsf{sk})$, and $\mathbf{x}' \leftarrow \mathtt{PExtract}(\mathsf{pp}, \mathsf{pk}')$. Note that $\mathbf{x}$ can be extracted from pk independently of the corresponding secret key due to the "binding public keys" property Def. 5 (iii). We can rely on the latter property since key pairs are valid because of condition (a). Again because of (a), shared keys are ip-computable, and we have

$$\begin{aligned} &\mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}, \mathsf{sk}') = \mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}', \mathsf{sk}) \\ \overset{\text{Def. 5 (ii)}}{\iff}\ & f_{\mathsf{pp}}(\langle \mathbf{x}, \mathbf{y}' \rangle) = f_{\mathsf{pp}}(\langle \mathbf{x}', \mathbf{y} \rangle) \\ \overset{f_{\mathsf{pp}}\text{ invertible}}{\iff}\ & \langle \mathbf{x}, \mathbf{y}' \rangle = \langle \mathbf{x}', \mathbf{y} \rangle, \end{aligned}$$

and thus it suffices to show equality of these inner products. Due to validity of all involved key pairs and condition (b), for all $i, j \in I$ with $i \neq j$, it holds:

$$\begin{aligned} \langle \mathbf{x}_i, \mathbf{y}_j \rangle &\overset{\text{Def. 5 (ii)}}{=} f_{\mathsf{pp}}^{-1}(\mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}_i, \mathsf{sk}_j)) \\ &\overset{\text{Cond. (b)}}{=} f_{\mathsf{pp}}^{-1}(\mathtt{SharedKey}(\mathsf{pp}, \mathsf{pk}_j, \mathsf{sk}_i)) \overset{\text{Def. 5 (ii)}}{=} \langle \mathbf{x}_j, \mathbf{y}_i \rangle. \end{aligned} \tag{1}$$

Analogously, by to validity of all involved key pairs and condition (c) for all $i \in I$, we have:

$$\langle \mathbf{x}_i, \mathbf{y} \rangle = \langle \mathbf{x}, \mathbf{y}_i \rangle \qquad \text{and} \qquad \langle \mathbf{x}_i, \mathbf{y}' \rangle = \langle \mathbf{x}', \mathbf{y}_i \rangle. \tag{2}$$

By condition (d) from the lemma statement, we can find $\beta, \gamma \in \mathbb{Z}_p^{|I|}$ with $\sum_{j=1}^{|I|} \beta_j \mathbf{x}_j = \mathbf{x}$ and $\sum_{i=1}^{|I|} \gamma_i \mathbf{x}_i = \mathbf{x}'$. First, note that for all $i \in I$ we have

$$\langle \mathbf{x}_i, \sum_{j=1}^{|I|} \beta_j \mathbf{y}_j \rangle = \sum_{j=1}^{|I|} \beta_j \langle \mathbf{x}_i, \mathbf{y}_j \rangle \overset{\text{Eq. 1}}{=} \sum_{j=1}^{|I|} \beta_j \langle \mathbf{x}_j, \mathbf{y}_i \rangle = \langle \mathbf{x}, \mathbf{y}_i \rangle \overset{\text{Eq. 2}}{=} \langle \mathbf{x}_i, \mathbf{y} \rangle. \tag{3}$$

With this, it follows that

$$\langle \mathbf{x}, \mathbf{y}' \rangle \overset{\text{Cond. (d)}}{=} \langle \sum_{j=1}^{|I|} \beta_j \mathbf{x}_j, \mathbf{y}' \rangle = \sum_{j=1}^{|I|} \beta_j \langle \mathbf{x}_j, \mathbf{y}' \rangle \overset{\text{Eq. 2}}{=} \sum_{j=1}^{|I|} \beta_j \langle \mathbf{x}', \mathbf{y}_j \rangle = \langle \mathbf{x}', \sum_{j=1}^{|I|} \beta_j \mathbf{y}_j \rangle.$$

Finally, we have

$$\langle \mathbf{x}', \sum_{j=1}^{|I|} \beta_j \mathbf{y}_j \rangle = \sum_{i=1}^{|I|} \gamma_i \langle \mathbf{x}_i, \sum_{j=1}^{|I|} \beta_j \mathbf{y}_j \rangle \overset{\text{Eq. 3}}{=} \sum_{i=1}^{|I|} \gamma_i \langle \mathbf{x}_i, \mathbf{y} \rangle = \langle \mathbf{x}', \mathbf{y} \rangle,$$

which concludes the proof. □

Note that this in particular implies that the shared key is *independent of the choice of secret keys* $\mathsf{sk}, \mathsf{sk}'$ *satisfying conditions (a) and (c).*

Relying on the fact that after giving out sufficiently many secret keys, all shared keys are uniquely determined, we are able to prove a trade-off between the tightness of the reduction and the dimension of an inner-product NIKE. We formalize this in the following theorem, which we prove in the full version, and give an interpretation of our result below.

Theorem 2. *Let $\mathcal{N} = (G, U, V)$ be a non-interactive complexity assumption, let $N, d \in \mathbb{N}$ with $4d + 6 < N$, and let $p \in \mathbb{N}$ a prime. Let* `NIKE` *be a perfectly correct 2-step-adaptively-secure d-dimensional ip-NIKE over $\mathbb{Z}_p$ with shared key space $\mathcal{K}$, public key space $\mathcal{PK}$ and secret key space $\mathcal{SK}$. Then, for any simple (ε_Λ, $\varepsilon_\mathcal{A}$)-reduction from breaking the NICA $\mathcal{N}$ to breaking the N-user 2-step-adaptive security of* `NIKE`*, there exists a PPT adversary $\mathcal{B}$ on the NICA $\mathcal{N}$, such that*

$$\varepsilon_\Lambda \leq \frac{4d+6}{N} \cdot \varepsilon_\mathcal{A} + \mathsf{Adv}^{\mathsf{nica}}_{\mathcal{N},\mathcal{B}}.$$

Interpretation. Theorem 2 says that if any reduction is successfully breaking the underlying NICA $\mathcal{N}$ with probability noticeably larger than $(4d + 6)/N$, the reduction can be turned into a standalone $\mathcal{N}$ solver, without help of an external adversary. More precisely, assuming $\mathcal{N}$ is hard we obtain

$$\varepsilon_\Lambda \leq \frac{4d+6}{N} \cdot \varepsilon_\mathcal{A} + \text{negl}$$

for a negligible function negl. This implies a security loss of at least $N/(4d + 6)$.

We can thus conclude that any inner-product NIKE that satisfies 2-step-adaptive security has to either have a significant loss, or ip-dimension proportional to the number of users N. In particular, this gives strong evidence that a fully-adaptive NIKEs with tight security only exist for an a priori fixed number of users, but not for a dynamic setting where users continuously join or leave. Altogether, using the relations between security notions depicted in Figure 4, we obtain the following informal corollary:

Corollary 2. *Any simple reduction from a non-interactive complexity assumption $\mathcal{N}$ to the X-security of a d-dimensional ip-NIKE has to lose a factor in the order of Y, where N is the number of public keys, $\mathcal{N}$ is assumed to be hard and $(X, Y) \in \{$(2-step-adaptive, $\Omega(N/d)$), (adaptive, $\Omega(N/d)$), (ν-semi-adaptive, $\Omega(\nu^2/(N \cdot d))$)$\}$.*

References

[1] Christoph Bader, Dennis Hofheinz, Tibor Jager, Eike Kiltz, and Yong Li. "Tightly-Secure Authenticated Key Exchange". In: *TCC 2015, Part I*. Ed. by Yevgeniy Dodis and Jesper Buus Nielsen. Vol. 9014. LNCS. Springer, Heidelberg, Mar. 2015, pp. 629–658. DOI: 10.1007/978-3-662-46494-6_26.

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

[3] Mihir Bellare, Dennis Hofheinz, and Scott Yilek. "Possibility and Impossibility Results for Encryption and Commitment Secure under Selective Opening". In: *EUROCRYPT 2009*. Ed. by Antoine Joux. Vol. 5479. LNCS. Springer, Heidelberg, Apr. 2009, pp. 1–35. DOI: 10.1007/978-3-642-01001-9_1.

[4] Dan Boneh and Ramarathnam Venkatesan. "Breaking RSA May Not Be Equivalent to Factoring". In: *EUROCRYPT'98*. Ed. by Kaisa Nyberg. Vol. 1403. LNCS. Springer, Heidelberg, 1998, pp. 59–71. DOI: 10.1007/BFb0054117.

[5] Dan Boneh and Mark Zhandry. "Multiparty Key Exchange, Efficient Traitor Tracing, and More from Indistinguishability Obfuscation". In: *CRYPTO 2014, Part I*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8616. LNCS. Springer, Heidelberg, Aug. 2014, pp. 480–499. DOI: 10.1007/978-3-662-44371-2_27.

[6] Colin Boyd, Wenbo Mao, and Kenneth G. Paterson. "Key Agreement Using Statically Keyed Authenticators". In: *ACNS 04*. Ed. by Markus Jakobsson, Moti Yung, and Jianying Zhou. Vol. 3089. LNCS. Springer, Heidelberg, June 2004, pp. 248–262. DOI: 10.1007/978-3-540-24852-1_18.

[7] Xavier Boyen, Qixiang Mei, and Brent Waters. "Direct Chosen Ciphertext Security from Identity-Based Techniques". In: *ACM CCS 2005*. Ed. by Vijayalakshmi Atluri, Catherine Meadows, and Ari Juels. ACM Press, Nov. 2005, pp. 320–329. DOI: 10.1145/1102120.1102162.

[8] 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.

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

[10] Jean-Sébastien Coron. "On the Exact Security of Full Domain Hash". In: *CRYPTO 2000*. Ed. by Mihir Bellare. Vol. 1880. LNCS. Springer, Heidelberg, Aug. 2000, pp. 229–235. DOI: 10.1007/3-540-44598-6_14.

[11] Jean-Sébastien Coron. "Optimal Security Proofs for PSS and Other Signature Schemes". In: *EUROCRYPT 2002*. Ed. by Lars R. Knudsen. Vol. 2332. LNCS. Springer, Heidelberg, 2002, pp. 272–287. DOI: 10.1007/3-540-46035-7_18.

[12] Ronald Cramer, Goichiro Hanaoka, Dennis Hofheinz, Hideki Imai, Eike Kiltz, Rafael Pass, abhi shelat, and Vinod Vaikuntanathan. "Bounded CCA2-Secure Encryption". In: *ASIACRYPT 2007*. Ed. by Kaoru Kurosawa. Vol. 4833. LNCS. Springer, Heidelberg, Dec. 2007, pp. 502–518. DOI: 10.1007/978-3-540-76900-2_31.

[13] Ronald Cramer and Victor Shoup. "Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption". In: *EURO-*

CRYPT 2002. Ed. by Lars R. Knudsen. Vol. 2332. LNCS. Springer, Heidelberg, 2002, pp. 45–64. DOI: 10.1007/3-540-46035-7_4.

[14] Whitfield Diffie and Martin E. Hellman. “New Directions in Cryptography”. In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654.

[15] Yevgeniy Dodis, Jonathan Katz, Adam Smith, and Shabsi Walfish. “Composability and On-Line Deniability of Authentication”. In: *TCC 2009*. Ed. by Omer Reingold. Vol. 5444. LNCS. Springer, Heidelberg, Mar. 2009, pp. 146–162. DOI: 10.1007/978-3-642-00457-5_10.

[16] Danny Dolev, Cynthia Dwork, and Moni Naor. “Nonmalleable Cryptography”. In: *SIAM Journal on Computing* 30.2 (2000), pp. 391–437.

[17] Régis Dupont and Andreas Enge. “Provably secure non-interactive key distribution based on pairings”. In: *Discrete Applied Mathematics* 154.2 (2006), pp. 270–276.

[18] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. “An Algebraic Framework for Diffie-Hellman Assumptions”. In: *CRYPTO 2013, Part II*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8043. LNCS. Springer, Heidelberg, Aug. 2013, pp. 129–147. DOI: 10.1007/978-3-642-40084-1_8.

[19] Eduarda S. V. Freire, Dennis Hofheinz, Eike Kiltz, and Kenneth G. Paterson. “Non-Interactive Key Exchange”. In: *PKC 2013*. Ed. by Kaoru Kurosawa and Goichiro Hanaoka. Vol. 7778. LNCS. Springer, Heidelberg, 2013, pp. 254–271. DOI: 10.1007/978-3-642-36362-7_17.

[20] Romain Gay, Dennis Hofheinz, Eike Kiltz, and Hoeteck Wee. “Tightly CCA-Secure Encryption Without Pairings”. In: *EUROCRYPT 2016, Part I*. Ed. by Marc Fischlin and Jean-Sébastien Coron. Vol. 9665. LNCS. Springer, Heidelberg, May 2016, pp. 1–27. DOI: 10.1007/978-3-662-49890-3_1.

[21] Craig Gentry, Amit Sahai, and Brent Waters. “Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based”. In: *CRYPTO 2013, Part I*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8042. LNCS. Springer, Heidelberg, Aug. 2013, pp. 75–92. DOI: 10.1007/978-3-642-40041-4_5.

[22] Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. “Leveled Fully Homomorphic Signatures from Standard Lattices”. In: *47th ACM STOC*. Ed. by Rocco A. Servedio and Ronitt Rubinfeld. ACM Press, June 2015, pp. 469–477. DOI: 10.1145/2746539.2746576.

[23] Jens Groth and Amit Sahai. “Efficient Non-interactive Proof Systems for Bilinear Groups”. In: *EUROCRYPT 2008*. Ed. by Nigel P. Smart. Vol. 4965. LNCS. Springer, Heidelberg, Apr. 2008, pp. 415–432. DOI: 10.1007/978-3-540-78967-3_24.

[24] 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*. Ed. by Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas. Vol. 12110. LNCS. Springer, Heidelberg, May 2020, pp. 374–395. DOI: 10.1007/978-3-030-45374-9_13.

[25] Julia Hesse, Dennis Hofheinz, and Lisa Kohl. “On Tightly Secure Non-Interactive Key Exchange”. In: *CRYPTO 2018, Part II*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10992. LNCS. Springer, Heidelberg, Aug. 2018, pp. 65–94. DOI: 10.1007/978-3-319-96881-0_3.

[26] Dennis Hofheinz and Tibor Jager. “Tightly Secure Signatures and Public-Key Encryption”. In: *CRYPTO 2012*. Ed. by Reihaneh Safavi-Naini and Ran Canetti. Vol. 7417. LNCS. Springer, Heidelberg, Aug. 2012, pp. 590–607. DOI: 10.1007/978-3-642-32009-5_35.

[27] Dennis Hofheinz, Eike Kiltz, and Victor Shoup. “Practical Chosen Ciphertext Secure Encryption from Factoring”. In: *Journal of Cryptology* 26.1 (Jan. 2013), pp. 102–118. DOI: 10.1007/s00145-011-9115-0.

[28] Eike Kiltz. “Chosen-Ciphertext Security from Tag-Based Encryption”. In: *TCC 2006*. Ed. by Shai Halevi and Tal Rabin. Vol. 3876. LNCS. Springer, Heidelberg, Mar. 2006, pp. 581–600. DOI: 10.1007/11681878_30.

[29] Roman Langrehr and Jiaxin Pan. “Tightly Secure Hierarchical Identity-Based Encryption”. In: *Journal of Cryptology* 33.4 (Oct. 2020), pp. 1787–1821. ISSN: 1432-1378. DOI: 10.1007/s00145-020-09356-x.

[30] Allison B. Lewko and Brent Waters. “Why Proving HIBE Systems Secure Is Difficult”. In: *EUROCRYPT 2014*. Ed. by Phong Q. Nguyen and Elisabeth Oswald. Vol. 8441. LNCS. Springer, Heidelberg, May 2014, pp. 58–76. DOI: 10.1007/978-3-642-55220-5_4.

[31] Daniele Micciancio and Chris Peikert. “Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller”. In: *EUROCRYPT 2012*. Ed. by David Pointcheval and Thomas Johansson. Vol. 7237. LNCS. Springer, Heidelberg, Apr. 2012, pp. 700–718. DOI: 10.1007/978-3-642-29011-4_41.

[32] Moni Naor, Omer Reingold, and Alon Rosen. “Pseudo-random functions and factoring (extended abstract)”. In: *32nd ACM STOC*. ACM Press, May 2000, pp. 11–20. DOI: 10.1145/335305.335307.

[33] Kenneth G. Paterson and Sriramkrishnan Srinivasan. “Building Key-Private Public-Key Encryption Schemes”. In: *ACISP 09*. Ed. by Colin Boyd and Juan Manuel González Nieto. Vol. 5594. LNCS. Springer, Heidelberg, July 2009, pp. 276–292.

[34] Chris Peikert and Brent Waters. “Lossy trapdoor functions and their applications”. In: *40th ACM STOC*. Ed. by Richard E. Ladner and Cynthia Dwork. ACM Press, May 2008, pp. 187–196. DOI: 10.1145/1374376.1374406.

[35] David Pointcheval and Olivier Sanders. “Forward Secure Non-Interactive Key Exchange”. In: *SCN 14*. Ed. by Michel Abdalla and Roberto De Prisco. Vol. 8642. LNCS. Springer, Heidelberg, Sept. 2014, pp. 21–39. DOI: 10.1007/978-3-319-10879-7_2.

[36] Oded Regev. “Quantum Computation and Lattice Problems”. In: *43rd FOCS*. IEEE Computer Society Press, Nov. 2002, pp. 520–529. DOI: 10.1109/SFCS.2002.1181976.

[37] Amit Sahai and Brent Waters. “How to use indistinguishability obfuscation: deniable encryption, and more”. In: *46th ACM STOC*. Ed. by David B. Shmoys. ACM Press, 2014, pp. 475–484. DOI: 10.1145/2591796.2591825.

[38] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. “Cryptosystems based on Pairing”. In: *SCIS 2000*. Okinawa, Japan, Jan. 2000.

[39] Brent Waters. “Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions”. In: *CRYPTO 2009*. Ed. by Shai Halevi. Vol. 5677. LNCS. Springer, Heidelberg, Aug. 2009, pp. 619–636. DOI: 10.1007/978-3-642-03356-8_36.