Fine-Grained Non-Interactive Key Exchange without Idealized Assumptions, and Lower Bounds *

Yuyu Wang [1] Chuanjie Su [1] Jiaxin Pan [2] Chunxiang Xu [1]

[1] University of Electronic Science and Technology of China, Chengdu, China
wangyuyu@uestc.edu.cn, chuanjie.su@hotmail.com, chxxu@uestc.edu.cn
[2] University of Kassel, Kassel, Germany
jiaxin.pan@uni-kassel.de

Abstract

In this paper, we study multi-party non-interactive key exchange (NIKE) in the fine-grained setting. More precisely, we propose three multi-party NIKE schemes in three computational models, namely, the bounded-parallel-time, bounded-time, and bounded-storage models. The bounded-parallel-time and bounded-time constructions rely on mild fine-grained assumptions, whereas the bounded-storage construction is unconditional. This improves the recent work of Afshar, Couteau, Mahmoody, and Sadeghi (EUROCRYPT 2023) that requires idealized assumptions, such as random oracles or generic groups.

Additionally, we show that all our constructions satisfy a natural desirable property that we refer to as extendability, and we give generic transformations from extendable multi-party NIKEs to multi-party identity-based NIKEs in the fine-grained setting.

Furthermore, we generalize the lower bound on users' storage consumption in the bounded-storage model by Dziembowski and Maurer (Eurocrypt 2004) to encompass any multi-party NIKE with extendability and local key independence. This lower bound shows that the users' storage consumption of our multi-party NIKE in the bounded-storage model is optimal up to constant factors within this natural class.

Keywords: Multi-party non-interactive key exchange, fine-grained cryptography, complexity assumption, lower bound, identity-based cryptography, bit-fixing pseudorandom function.

1 Introduction

1.1 Background

Non-interactive key exchange (NIKE) is a central primitive in cryptography, enabling a group of parties to simultaneously release a single message (i.e., the public key), and then securely agree on a session key in the presence of any potential adversaries after observing the public keys. Existing two-party NIKE schemes are constructed based on various cryptographic assumptions. These include well-established assumptions such as the Diffie-Hellman assumption [17], Learning With Errors [24], and assumptions related to factoring [21], and more exotic ones such as the existence of generic groups [36, 33] and algebraic groups [22]. Joux [28] proposed a three-party NIKE based on the bilinear Diffie-Hellman assumption over pairings. When it comes to multi-party NIKE involving more than four parties, all known candidates (against polynomial-time adversaries) are in the world of "obfustopia", where obfuscation exists [11]. This raises a natural question of whether it is possible to construct multi-party NIKE from much more established assumptions.

Multi-party NIKE based on mild assumptions. Very recently, Afshar et al. [2] provided an affirmative answer to the above question by introducing n_p-party NIKE schemes for any constant n_p in the

*A preliminary version of this paper appeared at Crypto 2024 [40]. This full version was published in the Journal of Cryptology [41].

random oracle model and Shoup's $((\mathsf{n_p}/2-1)$-linear) generic group model (i.e., the idealized multilinear map model where the number of groups is $\mathsf{n_p}/2$). These schemes are in the context of *fine-grained cryptography [16]*, where adversaries can only have limited resources and honest users do not possess more resources than adversaries. One of the main motivations of fine-grained cryptography is to base security on weak assumptions, or in some models no computational assumptions at all. Hence, the use of random oracles or generic groups makes their work less desirable.

> *Our Goal: Can we construct a fine-grained multi-party NIKE without idealized assumptions, such as random oracles and generic groups?*

We take a closer look at the fine-grained setting. Essentially, there are different ways to restrict users' and adversaries' computing power. It can be in terms of running time (e.g., [34, 9, 13]), parallel running time (i.e., circuit depths) (e.g., [25, 16, 15]), or storage (e.g., [31, 14, 18]). The multi-party NIKE of Afshar et al. was treated in the bounded-time model. In their scheme, an honest user runs in time $O(\lambda^{\mathsf{n_p}-1})$ (respectively, $O(\lambda)$) and the scheme is secure against adversaries running in time $o(\lambda^{\mathsf{n_p}})$ in the random oracle model (respectively, $o(\lambda^2)$ in Shoup's $(\mathsf{n_p}/2-1)$-linear generic group model). Although the assumptions of the random oracle and generic group model are already considered weak for multi-party NIKE schemes, they still fall into the worlds of "minicrypt" (the random oracle model), "cryptomania" (the generic group model), or even "obfustopia" (the multilinear generic group model) when dealing with a large number of parties. This raises the question of whether we can have multi-party NIKE even if we are in, say, the Pessiland, where one-way functions do not exist.

Concurrent work. An independent and concurrent recent work by Couteau et al. [7] introduced a 4-party NIKE protocol with a quadratic gap between users and adversaries (with honest users running in time $O(\lambda)$ and secure against adversaries running in time $o(\lambda^2)$), assuming the existence of exponentially secure injective pseudorandom generators (exp-iPRGs) and the sub-exponential hardness of the computational Diffie-Hellman problem (seCDH). Their results can also be extended to a 6-party NIKE under exp-iPRGs and CDH in bilinear groups, as well as to a 6-party NIKE (respectively, 9-party NIKE) with honest users running in time $O(\lambda)$ and secure against adversaries running in time $o(\lambda^{1.5})$ under exp-iPRGs and CDH without pairings (respectively, with pairings). They further showed that their 4-party protocol is secure in Maurer's generic group model with a smaller gap between users and adversaries (with honest users running in time $O(\lambda)$ and secure against adversaries running in time $o(\lambda^{1.6})$), assuming the existence of non-uniformly exponentially secure injective pseudorandom generators.

1.2 Our Contributions

We revisit multi-party NIKE in the fine-grained setting and propose constructions in the bounded-parallel-time, bounded-time, and bounded-storage models based on weak fine-grained complexity assumptions or, in the BSM, no computational assumptions. More precisely:

- In the bounded-parallel-time model, we construct an $\mathsf{n_p}$-party NIKE for any constant $\mathsf{n_p}$ computable in $\mathsf{AC}^0[2]$ and secure against adversaries in NC^1 (i.e., circuits with logarithmic depth and polynomial size). Clearly, the honest users consume less computational resources than adversaries, since $\mathsf{AC}^0[2] \subsetneq \mathsf{NC}^1$ [35, 37]. Similar to prior works in the same setting (e.g., [16, 15]), our security is based on the worst-case assumption $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$, which is widely believed to hold. All existing NC^1-fine-grained schemes are based on $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$ [16, 15, 5, 39, 20, 38]. According to Barrington, $\oplus\mathsf{L/poly}$ is the class of languages with polynomial-sized branching programs and NC^1 languages have polynomial-sized branching programs of constant width [6]. Hence, this assumption holds if there exists one language having only branching programs of super-constant width. It would be quite surprising if this does not hold.
- In the bounded-time model, we construct an $\mathsf{n_p}$-party NIKE for any constant $\mathsf{n_p}$ computable in $\tilde{O}(\lambda^{\mathsf{n_p}+k-1})$ and secure against adversaries running in time $\tilde{O}(\lambda^{\mathsf{n_p}+k-\Omega(1)})$. The underlying assumption is the average-case Zero-k-Clique hypothesis for any constant $k>2$, which is exactly the one used in [29] for constructing fine-grained two-party NIKE. It is also widely used in the fine-grained complexity to derive conditional lower bounds for other problems (e.g., [1, 12, 30, 4]). Note that no known approach for this assumption is faster than essentially λ^k, and the worst-case Zero-k-Clique hypothesis for any constant k does not appear to imply $\mathsf{P} \neq \mathsf{NP}$ [29]. Moreover, although subsequent work [3] shows that the conjecture fails for sufficiently large k in the absence

of one-way functions, our construction already achieves the largest security gap at $k = 3$, which is the weakest instantiation of the assumption since its strength increases with k. Therefore, the large-k separation of [3] does not apply to our small-constant regime and does not imply that our (strictly weaker) assumption entails one-way functions.

- In the bounded-storage model (BSM) [14], we extend the two-party NIKE in [14] to an $\mathsf{n_p}$-party NIKE for any constant $\mathsf{n_p}$. The resulting scheme is unconditionally secure against adversaries with memory bounds $O(\lambda^{\mathsf{n_p}+1})$ and each user consumes $O(\lambda^{\mathsf{n_p}})$ bits of memory. Also, the total communication cost between the parties (i.e., the total size of the public keys in this case) is $O(\lambda)$. This is also part of our efficiency measure.

These results show that weak (non-idealized) fine-grained assumptions, and in the BSM even without any computational assumptions, suffice to realize multi-party NIKE, which was previously achieved only via obfuscation- or multilinear-map-based constructions against polynomial-time adversaries, or under idealized assumptions in the fine-grained setting.

Remarks on the uniformly random string (URS). While our construction in the bounded-time model is proposed in the plain model, where no trusted public parameter is exploited, the constructions in the bounded-parallel-time model and BSM mentioned above are instantiated in the URS model, where there exists a URS accessible to all parties and adversaries. It is important to note that this model is weaker than the common reference string model, since the URS is generated in a "nothing up my sleeve" manner, ensuring that it contains no hidden information or trapdoors. Moreover, our scheme in the bounded-parallel-time model can also be instantiated without using a URS in the random oracle model, by deriving the URS from a hash function modeled as a random oracle. The same argument does not apply to our construction in the BSM, since it crucially relies on the URS disappearing after a certain period of time.

Comparison. In Tables 1 and 2, we provide a comparison of all existing fine-grained multi-party NIKE constructions. The tables contrast the schemes with respect to the number of parties, adversarial and user complexity, the use of a uniform random string (URS), correctness error, security error, computational model, the underlying assumptions, and key lengths.

Scheme	#P	Adv. Comp.	User Comp.	URS	Correctness Error	Security Error	Model	Assumption
ACMS23[2]	$\mathsf{n_p}$	$\lambda^{\mathsf{n_p}/(\mathsf{n_p}-1)}$	$\tilde{O}(\lambda)$	No	$\mathsf{negl}(\lambda)$	$\mathsf{negl}(\lambda)$	T	ROM
	$2\mathsf{n_p}$	$o(\lambda^2)$	$O(\lambda)$	No	$\mathsf{negl}(\lambda)$	$\mathsf{negl}(\lambda)$	T	$(\mathsf{n_p}-1)$-linear SGGM
BCS24[7]	4	$o(\lambda^2)$	$O(\lambda)$	No	$\mathsf{negl}(\lambda)$	$\mathsf{negl}(\lambda)$	T	exp-iPRGs, seCDH
	6	$o(\lambda^2)$	$O(\lambda)$	No	$\mathsf{negl}(\lambda)$	$\mathsf{negl}(\lambda)$	T	exp-iPRGs, seCDH2
	6	$o(\lambda^{1.5})$	$O(\lambda)$	No	$\mathsf{negl}(\lambda)$	$\mathsf{negl}(\lambda)$	T	exp-iPRGs, seCDH
	9	$o(\lambda^{1.5})$	$O(\lambda)$	No	$\mathsf{negl}(\lambda)$	$\mathsf{negl}(\lambda)$	T	exp-iPRGs, seCDH2
	4	$o(\lambda^{1.6})$	$O(\lambda)$	No	$\mathsf{negl}(\lambda)$	$\mathsf{negl}(\lambda)$	T	exp-iPRGs, MGGM
Section 3	$\mathsf{n_p}$	NC^1	$\mathsf{AC}^0[2]$	Yes	0	$\mathsf{negl}(\lambda)$	P	$\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$
Section 4	$\mathsf{n_p}$	$\tilde{O}(\lambda^{\mathsf{n_p}+k-\delta})$	$\tilde{O}(\lambda^{\mathsf{n_p}+k-1})$	No	$\tilde{O}(\lambda^{k+\mathsf{n_p}-k\mathsf{n_p}})$	$1/\lambda^{o(1)}$	T	Zero-k-Clique
Section 5	$\mathsf{n_p}$	$O(\lambda^{\mathsf{n_p}+1})$	$O(\lambda^{\mathsf{n_p}})$	Yes	$2/\lambda$	$O(1/\sqrt{\lambda}), \mathsf{negl}(\lambda)$	M	None

Table 1: Comparison of fine-grained multi-party NIKEs. Here, λ denotes the security parameter and $\mathsf{n_p}$ the (constant) number of parties. negl is an unspecified negligible function. Let $k \geq 3$ and $0 < \delta < 1$ be constants; larger k implies a stronger Zero-k-Clique assumption. The column "#P" denotes the number of parties, "Adv. Comp." the complexity of the adversary, "User Comp." the complexity of the user, and "URS" whether a uniform random string is used. In the "Model" column, "T", "P", and "M" denote the bounded-time, bounded-parallel-time, and bounded-storage models, "seCDH2" denotes the seCDH assumption with bilinear pairings. For Section 5, the "Adv. Comp." and "Security Error" columns denote that the construction satisfies the $(O(\lambda^{\mathsf{n_p}+1}), O(1/\sqrt{\lambda}), \mathsf{negl}(\lambda))$-security in the BSM as defined in Section 2.1.

Note that although our schemes in the bounded-time model and the BSM incur non-negligible correctness errors, and the BSM construction also has the stated information-theoretic security loss, these errors remain acceptable for concrete parameter choices in practice (see the concrete examples in Sections 4 and 5).

Extension to multi-party identity-based NIKE (IB-NIKE). We further show that our NIKEs can be transformed into IB-NIKEs for any constant number of parties. Notice that such IB-NIKEs can be trivially constructed from NIKE by including NIKE public keys of all users in the IB-NIKE key for

Scheme	#P	|Key|	|PK|	|SK|
ACMS23[2]	$\mathsf{n_p}$	$O(\log \lambda)$	$O(\lambda^{\mathsf{n_p}/(\mathsf{n_p}-1)})$	$O(\lambda \log \lambda)$
	$2\mathsf{n_p}$	$O(\log \lambda)$	$O(\lambda \log \lambda)$	$O(\lambda \log \lambda)$
BCS24[7]	4	$O(\log \lambda)$	$O(\lambda \log \lambda)$	$O(\lambda \log \lambda)$
	6	$O(\log \lambda)$	$O(\lambda \log \lambda)$	$O(\lambda \log \lambda)$
	6	$O(\log \lambda)$	$O(\lambda \log \lambda)$	$O(\lambda \log \lambda)$
	9	$O(\log \lambda)$	$O(\lambda \log \lambda)$	$O(\lambda \log \lambda)$
	4	$O(\log \lambda)$	$O(\lambda \log \lambda)$	$O(\lambda \log \lambda)$
Section 3	$\mathsf{n_p}$	$\mathsf{poly}(\lambda)$	$\mathsf{poly}(\lambda)$	$\mathsf{poly}(\lambda)$
Section 4	$\mathsf{n_p}$	1	$\tilde{O}(\lambda^{(\mathsf{n_p}+2)})$	$\tilde{O}(\lambda^{(\mathsf{n_p}-1)})$
Section 5	$\mathsf{n_p}$	$O(\lambda)$	$O(\lambda)$	$O(\lambda^{\mathsf{n_p}})$

Table 2: Comparison of fine-grained multi-party NIKEs. Here, λ denotes the security parameter. Let $k \geq 3$ be a constant in the Zero-k-Clique assumption. The column "|Key|" denotes shared key length, "|PK|" denotes the public key length, and "|SK|" denotes the secret key length.

Multi-Party IB-NIKE $\overset{\text{Sec. 6.2}}{\Longleftarrow}$ (Restricted) BPRF $\overset{\text{Sec. 6.3}}{\Longleftarrow}$ Extendable Multi-Party NIKE

Figure 1: Overview of our approaches in constructing IB-NIKE in the fine-grained setting.

each identity. However, our construction of IB-NIKE is significantly better than this trivial one, since the identity-space of IB-NIKE is much larger. If the underlying NIKE supports a super-constant (say, polynomial) number of users, then our IB-NIKEs can support an exponentially large identity-space. Indeed, for our results in the BSM, making the number of users super-constant does not spoil the security proof since it is based on no assumptions. Similarly, our results in the bounded-time model can also support a super-constant number of users if we make the underlying assumption stronger, i.e., Zero-k-Clique assumption for super-constant k, which, however, may imply $\mathsf{P} \neq \mathsf{NP}$ [29].

Our transformation contains several intermediate steps, as described figuratively in Figure 1. Specifically, we follow the technique in [27] to show that in the fine-grained setting, bit-fixing pseudorandom functions (BPRFs) can be converted into multi-party IB-NIKEs, and then provide a generic construction of BPRF from any NIKE with a natural property we call extendability, which is satisfied by all our constructions (and indeed also the ones in previous works [28, 2]) except for those based on multilinear maps. Here we note that our multi-party NIKE in the bounded-parallel-time model only satisfies restricted extendability and can only be converted into a restricted version of BPRF. Nonetheless, this restricted version is sufficient for achieving multi-party IB-NIKE. Moreover, when the number of users is constant, instantiating the above transformation with any of our NIKE constructions preserves all parameters up to asymptotic equivalence. In particular, the adversary's resources, user complexity, and key length remain asymptotically the same as in the underlying NIKE. We refer the reader to Section 6 for the full details.

Negative result in the BSM. In the BSM, in addition to our positive result, we extend the negative result in [19] for two-party NIKEs to show that for an $\mathsf{n_p}$-party NIKE with (even restricted) extendability and a natural property we call local key independence, the entropy of the shared key conditioned on the public keys is bounded by roughly $\mathsf{m_u}^{\mathsf{n_p}}/\mathsf{m_a}^{\mathsf{n_p}-1}$, where $\mathsf{m_u}$ and $\mathsf{m_a}$ represent the amounts of memory consumed by each user and adversary respectively. Hence, for sharing λ bits, $\mathsf{m_u} = \Omega(\sqrt[\mathsf{n_p}]{\lambda \cdot \mathsf{m_a}^{\mathsf{n_p}-1}})$. Local key independence is the BSM-specific regularity condition formalized in Definition 2.4. Here local keys are the intermediate keys produced during the extendability procedure. Roughly, it says that public keys and independently sampled dummy key pairs used in the lower-bound argument reveal only limited information about local keys, and that adding one more key pair, independently generated conditioned on the same URS, does not create much extra leakage through this common URS. Additionally, as mentioned earlier, (restricted) extendability is satisfied by all existing multi-party NIKE schemes (except for those based on multilinear maps) to the best of our knowledge. Both properties are quite natural, and it would be surprising if any construction could circumvent our lower bound. This suggests that our multi-party NIKE in the BSM is optimal up to constant factors within the natural class of schemes satisfying extendability and local key independence.

As far as we know, this is the first negative result for multi-party NIKE in the BSM. In contrast to [19], our analysis does not require the schemes to satisfy perfect security. In this sense, our result applies more broadly even in the two-party setting.

Practical implications. Other than relying on weak assumptions and having low computational

complexity, our results have the following practical implications.

Our constructions in the bounded (parallel-)time setting are particularly well-suited for systems in which attacks are only meaningful if they succeed within a short time window. For example, users may employ the shared key to protect information that is valuable only temporarily and can be safely revealed or discarded afterward. Concretely, the key can be used to transmit short-lived information (e.g., verification codes) that are revealed only after a prescribed deadline. Similarly, the key can be used to encrypt files that, although logically deleted by users, remain temporarily retained by the cloud provider before permanent erasure. Security is required only during this enforced retention window.

Another direct application is the generation of message authentication codes (MACs) using the shared key. The verifier imposes a strict upper bound on the allowed computation time for generating a valid tag, and any party exceeding this bound is rejected. Thus, adversaries who require asymptotically more time gain no advantage, since tags produced after the deadline are invalid.

To illustrate the concrete magnitude of this gap, in our construction in the bounded-parallel-time model, the honest user runs in $\mathsf{AC}^0[2]$ and hence in NC^1, whereas the known distinguishing approach for $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L}/\mathsf{poly}$ proceeds via determinant computation in NC^2 for matrices from $\mathsf{ZeroSamp}_\lambda$ and $\mathsf{OneSamp}_\lambda$. For $\lambda = 2^{50}$ and any constant $\mathsf{n_p}$, honest evaluation has depth $O(\log \lambda)$, which is about 50 ignoring constant factors (and can in fact be implemented in $\mathsf{AC}^0[2]$ using only a constant number of layers). In contrast, this determinant-based approach has depth $O((\log \lambda)^2)$, which is about 2500 ignoring constant factors. This concrete separation makes the timeout mechanism practically enforceable.

In the bounded-time model, although the security gap is smaller, the same paradigm applies. For example, when the number of parties is 3, each honest party runs in time about $\tilde{O}(\lambda^5)$, whereas the scheme remains secure against adversaries below the λ^6 barrier by a polynomial factor. By enforcing a strict sequential time budget for generating valid MACs, the system ensures that honest users succeed within the permitted window, while slower adversaries are rejected.

In the BSM, our construction is optimal up to constant factors within the natural class of schemes satisfying extendability and local key independence, as shown by our matching impossibility result. This setting naturally captures scenarios where a high-speed broadcast channel generates URSs that are too large for an adversary to store but can be processed on the fly by honest parties.

More generally, as observed in [16], combining fine-grained primitives with standard cryptographic primitives yields hybrid schemes that are secure against restricted adversaries under weak (or even no) assumptions, while retaining security against polynomial-time adversaries under stronger assumptions.

1.3 Technical Details

Multi-party NIKE based on $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L}/\mathsf{poly}$. We now give technical details on how we construct our fine-grained multi-party NIKE in the bounded-parallel-time model. All the adversaries we consider in this part are in NC^1. Similar to previous works [16, 15, 5, 39, 20, 38], we exploit the indistinguishability of two specific distributions $\mathsf{ZeroSamp}_\lambda$ and $\mathsf{OneSamp}_\lambda$ outputting $\lambda \times \lambda$ matrices with rank $\lambda - 1$ and full rank respectively. It is implied by $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L}/\mathsf{poly}$.

Our starting point is to utilize the fine-grained hash proof system proposed in [20] to construct an NIKE via the technique proposed in [26]. Roughly we can let the projection and the statement of the hash proof system serve as the public keys in a two-party NIKE and let the parties share the proof with the (secret) verification key and the witness respectively. Security is guaranteed by the smoothness of the hash proof system. To extend it to a multi-party one, the first problem we must address is that the key pairs are vectors generated by different types of algorithms. When more parties are involved, it is unclear how to combine multiple vectors to generate a session key.

A straw-man solution is to extend the vectors to matrices. Taking the three-party case as an example, the public keys of the parties P_1, P_2, and P_3 are generated as $\mathbf{R}_1\mathbf{M}$, $\mathbf{M}\mathbf{R}_2$, and $\mathbf{M}\mathbf{R}_3$ respectively, where $\mathbf{M} \xleftarrow{\$} \mathsf{ZeroSamp}_\lambda$ is the public parameter and $\mathbf{R}_i$'s are the random secret matrices in $\{0,1\}^{\lambda\times\lambda}$. The parties can share $\mathbf{R}_1\mathbf{M}\mathbf{R}_2\mathbf{M}\mathbf{R}_3$ with certain entropy. However, unfortunately, it does not satisfy correctness. Specifically, $\mathbf{R}_1\mathbf{M}\mathbf{R}_2\mathbf{M}\mathbf{R}_3$ is indeed computable by P_1 and P_2 with their secret matrices, while it is not computable by P_3 due to the lack of knowledge on $\mathbf{R}_2\mathbf{M}$. To address this issue, we need to additionally publish $\mathbf{R}_2\mathbf{M}$. Nevertheless, this change will result in too much leakage about $\mathbf{R}_2$, compromising the security proof.

To overcome the technical hurdles mentioned above, we exploit the power of symmetric matrices.

Specifically, we make the public parameter symmetric by computing $\widetilde{\mathbf{M}} = \mathbf{M}^\top\mathbf{M}$, and generate $\widetilde{\mathbf{M}}\mathbf{R}_i$ and $\mathbf{R}_i$ as the public/secret key pair for each party, where $\mathbf{R}_i \xleftarrow{\$} \mathsf{SymR}_\lambda$ and SymR_λ is the uniform distribution over symmetric matrices in $\{0,1\}^{\lambda\times\lambda}$. The shared key is in the form of $\mathbf{R}_1\widetilde{\mathbf{M}}\mathbf{R}_2\cdots\widetilde{\mathbf{M}}\mathbf{R}_{\mathsf{n_p}}$. Thanks to the symmetry of the matrices, each party knows both $\widetilde{\mathbf{M}}\mathbf{R}_i$ and $\mathbf{R}_i\widetilde{\mathbf{M}} = \mathbf{R}_i^\top\widetilde{\mathbf{M}}^\top = (\widetilde{\mathbf{M}}\mathbf{R}_i)^\top$, enabling each party to compute the shared key, ensuring correctness.

Since the secret keys are not uniformly random in $\{0,1\}^{\lambda\times\lambda}$ now, we can no longer rely on the smoothness of the fine-grained hash proof system to prove security. To find a counterpart compatible with symmetric matrices, we define two new distributions $\mathcal{D}_0$ and $\mathcal{D}_1$:

$$\underbrace{\{\widetilde{\mathbf{M}}\mathbf{R} : \mathbf{R} \xleftarrow{\$} \mathsf{SymR}_\lambda\}}_{=\mathcal{D}_0} \text{ and } \underbrace{\{\widetilde{\mathbf{M}}\mathbf{R} + \mathbf{I}_\lambda : \mathbf{R} \xleftarrow{\$} \mathsf{SymR}_\lambda\}}_{=\mathcal{D}_1},$$

where $\widetilde{\mathbf{M}} = \mathbf{M}^\top\mathbf{M}$ for the public parameter $\mathbf{M} \xleftarrow{\$} \mathsf{ZeroSamp}_\lambda$ and $\mathbf{I}_\lambda$ is the identity matrix. We then prove the indistinguishability of $\mathcal{D}_0$ and $\mathcal{D}_1$ by proposing a new technique to leverage the symmetry and the indistinguishability of the matrices. For more details, we refer the reader to Section 3.

A useful property of the above distribution is that the last column vector in a matrix in $\mathcal{D}_0$ (respectively, $\mathcal{D}_1$) must be in (respectively, outside) the span of $\widetilde{\mathbf{M}}$. Hence, by embedding a random bit into the secret key of one party via vectors in $\mathrm{Ker}(\mathbf{M})$, and therefore in $\mathrm{Ker}(\widetilde{\mathbf{M}})$ since $\widetilde{\mathbf{M}} = \mathbf{M}^\top\mathbf{M}$, and switching the distributions of public keys of other parties between $\mathcal{D}_0$ and $\mathcal{D}_1$ in a flexible way, we can prove that the bottom-right entry of the shared matrix is uniformly random in the view of an NC^1 adversary. By running the scheme multiple times in parallel, they can share a session key with any polynomial size, without increasing the complexity of the circuits' depth.

We note that [38] has shown the existence of a public-coin algorithm (denoted by $\mathsf{RandSamp}_\lambda$ in this work, see Figure 3) with the output distribution being identical to $\mathsf{ZeroSamp}_\lambda$ and $\mathsf{OneSamp}_\lambda$ with probability one half each. Therefore, we can use this algorithm to generate a public parameter indistinguishable from the original one, resulting in a fine-grained multi-party NIKE in the URS model.

Multi-party NIKE based on Zero-k-Clique. Our construction in the bounded-time model, which is based on the Zero-k-Clique hypothesis, extends the one presented in [29]. In the construction in [29], two parties exchange lists containing ℓ instances of the Zero-k-Clique problem, out of which $\sqrt{\ell}$ instances have solutions. The expectation is that there would be exactly a single instance in common having a solution, allowing the parties to share the index of this instance by brute forcing among the $\sqrt{\ell}$ instances. The Zero-k-Clique hypothesis says that finding the correct solution among the ℓ instances requires essentially solving all instances, ensuring security.

To extend this construction to a multi-party setting, a natural approach is to let all parties exchange their lists of instances, and share the common indices of the instances with one solution. The primary challenge lies in the rapid increase in the error probability of correctness as the number of users increases. To address this issue, we increase the number of instances with solutions for each party and allow users to share the sum of the overlapping part. For a rigorous analysis of correctness, we exploit a set of indicators in a careful manner. Each indicator, denoted by index i, equals 1 if the ith random index chosen by the last party is also chosen by all other parties, and 0 otherwise. By bounding the probability that all indicators are equal to 0, we obtain the small error probability.

Our reduction of security aims to find the correct solution among ℓ instances of the Zero-k-Clique problem by making use of an adversary that can recover the shared key. To simulate the view of the adversary, which consists of the transcript (i.e., the public keys) among different parties, the reduction splits the list into multiple ones with a solution in the same location, and plants $\ell^{(\mathsf{n_p}-1)/\mathsf{n_p}}$ solutions into the list. Then the reduction can extract the index of the solution by using the common indices in the shared key. Specifically, our security proof introduces a generalized splitting algorithm with a binary tree structure, which splits one instance into multiple ones having the same number of solutions as the split one. By introducing additional checking procedures the proof can proceed without security loss potentially caused by the generalized splitting algorithm. The base construction only satisfies weak recovery security, namely, it prevents the adversary from recovering the shared key. To use the Goldreich–Levin extractor [23] to obtain full indistinguishability security, the recovery probability must first be made sufficiently small. We reduce it by parallel repetition [8] before using the Goldreich–Levin extractor. Moreover, to keep the Goldreich–Levin reconstruction efficient, we let each party compress the shared indices by summing them into a single short value. This ensures that the running time of the

Goldreich–Levin reconstruction remains acceptable.

To ensure a sufficient gap between users and adversaries, many more details need to be considered, including the number of solutions to be planted into instances during the simulation. We omit them here and refer the reader to Section 4.

Multi-party key exchange in the bounded-storage model. Recall that in the BSM, all parties and the adversary have access to a long URS. When the URS becomes unavailable, all parties broadcast public keys and share a session key with no additional interaction. The state-of-the-art NIKE (with public discussion) in this model is the two-party construction proposed in [14]. Each party stores a set of bits in the URS, and these bits are indexed in a way that they are (approximately) pairwise independent. Once the URS disappears, the parties exchange the indices and save the bits in common. It is crucial that the indices in the intersection are also pairwise independent so that the shared bits have high entropy. This allows the parties to apply privacy amplification. To ensure efficiency in terms of memory and communication, the parties utilize strongly 2-universal hash functions to represent the pairwise independent indices.

We apply the security analysis to the multi-user setting by proving that the common indices obtained from several approximately pairwise-independent index sets still satisfy the pairwise distribution needed for privacy amplification. While this is a natural extension of the two-party analysis, it has not been rigorously treated before, as far as we know. For correctness, we also need to show that the intersection contains sufficiently many distinct indices. Even in the two-party construction of [14], only the expected intersection size was analyzed. We prove a concentration bound by introducing indicators for the indices chosen by the last party. These indicators are not independent, while they are pairwise negatively correlated, which is enough for a Chebyshev-type bound. The injectivity of the hash family ensures that their sum counts distinct common indices, and thus the proof analyzes the actual shared bits rather than multiplicity hits.

We would like to highlight that in [2], the Chernoff bound is used in an elegant way for proving the correctness of a fine-grained multi-party NIKE. Compared to their approach, ours offers conceptual simplicity by defining the indices differently, eliminating the need for a chain of Chernoff bounds, and is more general, in the sense that it works for pairwise-independent indices rather than only for completely independent ones.

Extendability and IB-NIKE. Without considering the steps for privacy amplification, the sharing procedures of all our constructions (as well as the multi-party NIKEs in [2]) involve combining the secret key with public keys of other users sequentially. Each intermediate value, called a "local key", essentially represents a shared key (without privacy amplification) amongst a subgroup of all parties. We refer to this inherent property as extendability. With this property, one can easily extend the shared (local) key amongst a certain group of parties to a larger group by simply using the public keys of newly joined parties, without the need to re-run the entire sharing procedure. Besides its independent interest, we observe that such a property provides us with a "multilinear map-like" structure: the public keys can be viewed as group elements, and together with a local key, they form a "multilinear Diffie-Hellman-like" tuple. By carefully combining this insight with the technique introduced by Boneh and Waters [10], which constructs BPRFs using multilinear maps, we obtain a generic construction of BPRFs from multi-party NIKEs with extendability. Utilizing existing generic techniques [10, 27], we subsequently achieve fine-grained multi-party IB-NIKEs from BPRFs.

Negative results on multi-party NIKE in the BSM. For adversaries with memory bound $\mathsf{m_a}$, each user in our $\mathsf{n_p}$-party NIKE in the BSM consumes $\mathsf{m_u} = O(\sqrt[\mathsf{n_p}]{\lambda \cdot \mathsf{m_a}^{\mathsf{n_p}-1}})$ bits of memory, which might seem a bit disappointing. However, we extend the negative result of Dziembowski and Maurer [19] for two-party NIKEs and show that this memory consumption is optimal up to constant factors within the natural class of multi-party NIKEs satisfying extendability and local key independence.

The proof proceeds by induction. First, the entropy of the local key shared by two parties is bounded by roughly $\mathsf{m_u}^2/\mathsf{m_a}$. Then, assuming that the entropy of the local key shared by $i-1$ parties is at most roughly $\mathsf{m_u}^{i-1}/\mathsf{m_a}^{i-2}$, we show that the entropy of the local key shared by i parties is at most roughly $\mathsf{m_u}^{i}/\mathsf{m_a}^{i-1}$. The key observation is that extending an $(i-1)$-party local key using the public key of a new user can be viewed as a two-party key agreement step between the subgroup and the new user.

In the original two-party argument of [19], the parties' stored information is almost independent of Eve's view. This is no longer automatic when one party's state is replaced by an intermediate local

key, which may already have some leakage. We handle this using local key independence, which bounds both the leakage of local keys and the extra leakage caused by the common URS. Together with the Dziembowski-Maurer averaging lemma [19] and Maurer's public-discussion bound [32], this yields the desired iterative lower bound. We refer the reader to Section 5.2 for the full details.

1.4 Comparison with the Proceedings Version

This is the extended version of the paper that appeared at Crypto 2024 [40]. In this version, we show new limitations of multi-party NIKEs in the BSM, which imply the optimality of our construction up to constant factors within the natural class of schemes satisfying extendability and local key independence, in Section 5.2. Additionally, we give definitions, constructions, and security proofs of the fine-grained BPRFs and multi-party IB-NIKEs in Section 6.

2 Preliminaries

Notations. In this paper, algorithms are modeled as *functions* or *(streaming) word-RAMs*. A function represents a circuit with a specified domain and range, and we measure its complexity by its depth (i.e., the parallel running time). A word-RAM has access to memory and registers, each holding one word. Its running time is measured in the number of basic operations (e.g., addition, subtraction, multiplication, bit-shifting, and memory access). A word-RAM is a streaming one if it receives inputs in a streaming manner and its local memory is measured in bits.

We write $a \xleftarrow{\$} \mathcal{A}(b)$ (respectively, $a = \mathcal{A}(b)$) to denote the random variable output by a probabilistic (respectively, deterministic) algorithm $\mathcal{A}$ on input b. By $x \xleftarrow{\$} \mathcal{S}$ we denote sampling x uniformly from a set $\mathcal{S}$; when $\mathcal{S}$ is a distribution, we sample according to $\mathcal{S}$. By negl we denote an unspecified negligible function.

By $\mathbf{x} \in \{0,1\}^n$ we denote a column vector with size n, by x_i we denote the ith element of a vector $\mathbf{x}$. By $[n]$ we denote the set $\{1, \cdots, n\}$, by $[t_1, t_2]$ we denote the set $\{t_1, t_1+1, \cdots, t_2\}$, and by $\mathsf{MSB}_k(\mathbf{x})$ we denote the most significant k bits in (the bit string representing) $\mathbf{x}$. By $\log n$ we mean $\lceil \log n \rceil$. For some set or sequence $\mathcal{S}$, by $\mathbf{x}_\mathcal{S}$ we denote the sequence of all bits in $\mathbf{x}$ with indices (ordered lexicographically) in $\mathcal{S}$. For a matrix $\mathbf{A}$, we denote the set $\{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{0}\}$ by $\mathrm{Ker}(\mathbf{A})$. By $\mathbf{I}_n$ we denote an identity matrix in $\{0,1\}^{n \times n}$. By $\mathbf{0}$ we denote a zero vector or matrix.

By $H(x)$ we denote the *Shannon entropy* of a random variable x with alphabet $\mathcal{X}$ defined as

$$H(x) = -\sum_{\hat{x} \in \mathcal{X}} \Pr[x = \hat{x}] \log \Pr[x = \hat{x}].$$

By $H(x \mid y)$ we denote the *conditional entropy* of x conditioned on a random variable y with alphabet $\mathcal{Y}$ defined as

$$H(x \mid y) = \sum_{\hat{y} \in \mathcal{Y}} \Pr[y = \hat{y}] \cdot H(x \mid y = \hat{y}).$$

By $I(x; y)$ we denote the *mutual information* of x and y defined as

$$I(x; y) = H(x) - H(x \mid y),$$

which is the reduction of the uncertainty of x when y is learned. By $\mathbb{E}[x]$ and $\mathsf{Var}[x]$ we denote the *expected value* and *variance* of the variable x. By $\Delta(x, y)$ we denote the *total variation distance* of two random variables x, y with alphabets $\mathcal{X}$ and $\mathcal{Y}$ defined as

$$\Delta(x, y) = (1/2) \sum_{\hat{x} \in \mathcal{X} \cup \mathcal{Y}} |\Pr[x = \hat{x}] - \Pr[y = \hat{x}]|.$$

A sequence of random variables $x_1, \cdots, x_q$ with alphabet $\mathcal{X}$ is said to be *independent* if for any $\hat{x}_1, \cdots, \hat{x}_q \in \mathcal{X}$, we have

$$\Pr[\wedge_{i=1}^{q} x_i = \hat{x}_i] = \prod_{i=1}^{q} \Pr[x_i = \hat{x}_i].$$

If we only require that

$$\Pr[x_i = \hat{x}_i \wedge x_j = \hat{x}_j] = \Pr[x_i = \hat{x}_i] \cdot \Pr[x_j = \hat{x}_j]$$

holds for any $1 \le i < j \le q$, then the sequence is said to be *pairwise independent*.

2.1 Fine-Grained Multi-Party NIKE

In this section, we define fine-grained multi-party NIKE and multi-party key exchange in the BSM. In the following, a circuit (respectively, word-RAM) family is a family of (possibly randomized) circuits (respectively, word-RAMs) with respect to all security parameters $\lambda \in \mathbb{N}$, denoted by $\{f_\lambda\}_{\lambda\in\mathbb{N}}$. For brevity, we refer to it as $\{f_\lambda\}$.

We now define the multi-party NIKE and its security. We require that the users be in $\mathcal{C}_1$ and the adversaries be in $\mathcal{C}_2$. In this work, $\mathcal{C}_1$ and $\mathcal{C}_2$ represent classes of circuit families in the bounded-parallel-time model, and represent classes of word-RAM families in the bounded-time model. In the BSM, $\mathcal{C}_1$ refers to a class of streaming word-RAM families with bounded storage, which receive inputs in a streaming manner and store the outputs.

Definition 2.1 (Multi-party non-interactive key exchange). A $\mathcal{C}_1$-$\mathsf{n_p}$*-party non-interactive key exchange scheme (NIKE)* consists of two algorithm families $\mathsf{NIKE} = \{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\} \in \mathcal{C}_1$ such that

- Gen_λ on input $\mathsf{urs} \in \{0,1\}^n$ for some $n = n(\lambda)$ outputs a public key pk and a secret key sk.
- Share_λ is a deterministic algorithm that on input a set of public keys $(\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}$, an index i, and a secret key sk_i outputs either $\mathsf{key} \in \mathcal{K}_\lambda$ or the abort symbol $\perp \notin \mathcal{K}_\lambda$, where $i \in [\mathsf{n_p}]$ and $\mathcal{K}_\lambda$ is the session key space.

ϵ*-correctness* is satisfied if for all $\lambda \in \mathbb{N}$, all $i, j \in [\mathsf{n_p}]$, we have

$$\Pr[\exists \mathsf{key} \in \mathcal{K}_\lambda : \mathsf{Share}_\lambda(i, \mathsf{sk}_i, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}) = \mathsf{Share}_\lambda(j, \mathsf{sk}_j, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}) = \mathsf{key}] \ge 1 - \epsilon$$

where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$ and $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for all $i \in [\mathsf{n_p}]$.

$\mathcal{C}_2$-ϵ*-security* (respectively, $\mathcal{C}_2$-ϵ*-weak security*) is satisfied if for any adversary $\mathcal{A} = \{\mathsf{adv}_\lambda\} \in \mathcal{C}_2$ and all sufficiently large $\lambda \in \mathbb{N}$, we have

$$|\Pr[1 \xleftarrow{\$} \mathsf{adv}_\lambda(\mathsf{urs}, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}, \mathsf{key})] - \Pr[1 \xleftarrow{\$} \mathsf{adv}_\lambda(\mathsf{urs}, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}, \mathsf{key}')]| \le \epsilon$$

$$(\text{respectively, } \Pr[\mathsf{key} = \mathsf{adv}_\lambda(\mathsf{urs}, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]})] \le \epsilon)$$

where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for all $i \in [\mathsf{n_p}]$, $\mathsf{key} = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]})$, and $\mathsf{key}' \xleftarrow{\$} \mathcal{K}_\lambda$. In the plain model, we use the convention $\mathsf{urs} = \perp$.

$(\mathsf{m_a}, \epsilon, \theta)$*-security in the BSM* is satisfied if for any function $f : \{0,1\}^n \to \{0,1\}^{\mathsf{m_a}}$, we have

$$\exists \mathsf{E} : \ \Pr[\mathsf{E}] \ge 1 - \epsilon \text{ and } I(\mathsf{key}; (f(\mathsf{urs}), (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}) \mid \mathsf{E}) \le \theta,$$

where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for all $i \in [\mathsf{n_p}]$, and $\mathsf{key} = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]})$.

Remarks on the BSM. Here we note that the security in the BSM captures the situation where all parties have access to urs and the adversary is allowed to apply any (potentially unbounded) storage function f with an output length limit of $\mathsf{m_a}$ and store information on urs. Afterwards, urs becomes inaccessible, and all parties broadcast public keys, which are typically required to be short, and share a session key in a non-interactive way. Notice that we consider the traditional BSM [14] rather than the streaming BSM proposed by Dodis, Quach, and Wichs [18]. In the streaming BSM, the parties are allowed to communicate interactively with perhaps multiple rounds of long messages.

Extendability of Multi-Party NIKE. We now define a property of multi-party NIKE called extendability. Roughly, the extendability of a multi-party NIKE says that the sharing procedure involves an algorithm called $\mathsf{Combine}_\lambda$, which sequentially combines a user's secret key with other public keys, and the intermediate value produced by $\mathsf{Combine}_\lambda$, referred to as a local key, can be used to derive the shared key by using another algorithm $\mathsf{Extract}_\lambda$.

Definition 2.2 (Extendability). A $\mathcal{C}_1$-$\mathsf{n_p}$-party non-interactive key exchange scheme $\mathsf{NIKE} = \{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\} \in \mathcal{C}_1$ satisfies ϵ*-extendability* if there exist two deterministic algorithm families $\{\mathsf{Combine}_\lambda, \mathsf{Extract}_\lambda\} \in \mathcal{C}_1$ such that, for every $\lambda \in \mathbb{N}$, the following holds. For some $n = n(\lambda)$, let $\mathsf{urs} \xleftarrow{\$}$

$\{0,1\}^n$ and $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for all $i \in [\mathsf{n_p}]$. There exists an event E such that $\Pr[\mathsf{E}] \geq 1-\epsilon$, where E is over the choice of urs and the coins of all calls to $\mathsf{Gen}_\lambda(\mathsf{urs})$. Conditioned on E, for every $\mathcal{V} \subseteq [\mathsf{n_p}]$, nonempty $\mathcal{V}' \subseteq \mathcal{V}$, $j \in \mathcal{V}$, and $i' \in \mathcal{V}'$, we have

$$\mathsf{Combine}_\lambda(\mathsf{localkey}_{\mathcal{V}'}, \{\mathsf{pk}_i\}_{i\in\mathcal{V}\setminus\mathcal{V}'}) = \mathsf{Combine}_\lambda(\mathsf{sk}_j, \{\mathsf{pk}_i\}_{i\in\mathcal{V}\setminus\{j\}})$$

and

$$\mathsf{Extract}_\lambda(\mathsf{localkey}_{[\mathsf{n_p}]}, \mathsf{pk}_{\mathsf{n_p}}) = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}),$$

where $\mathsf{localkey}_{\mathcal{V}'} = \mathsf{Combine}_\lambda(\mathsf{sk}_{i'}, \{\mathsf{pk}_i\}_{i\in\mathcal{V}'\setminus\{i'\}})$ and $\mathsf{localkey}_{[\mathsf{n_p}]} = \mathsf{Combine}_\lambda(\mathsf{sk}_1, \{\mathsf{pk}_i\}_{i>1})$.

We also give the definition of restricted extendability, which requires that $\mathsf{Combine}_\lambda$ only generates the local key of the users with consecutive indices.

Definition 2.3 (Restricted extendability). *ϵ-restricted extendability* is defined in exactly the same way as ϵ-extendability except that the set $\mathcal{V} \subseteq [\mathsf{n_p}]$ is restricted to $\mathcal{V} = [\mathsf{t}_1, \mathsf{t}_2] \subseteq [\mathsf{n_p}]$ for any $\mathsf{t}_1, \mathsf{t}_2 \in [\mathsf{n_p}]$ and $\mathcal{V}'$ is restricted to $\mathcal{V}' = [\mathsf{s}_1, \mathsf{s}_2]$ for any $\mathsf{s}_1, \mathsf{s}_2 \in [\mathsf{n_p}]$ such that $\mathsf{t}_2 \geq \mathsf{s}_2 \geq \mathsf{s}_1 \geq \mathsf{t}_1$.

Local key independence in the BSM. Next we define a natural property called local key independence for NIKE in the BSM. Roughly, it says that public information and auxiliary key pairs reveal only limited information about local keys, and that adding one more independently generated key pair reveals only limited additional information through the common URS.

Definition 2.4 (Local key independence). A $\mathsf{n_p}$-party NIKE in the BSM with ϵ-(restricted) extendability satisfies $((\mu_i)_{i\in[\mathsf{n_p}]}, t)$-*local key independence in the BSM*, for $t \in \mathbb{N}$, if the following holds. In the key-generation experiment of Definition 2.2 (or Definition 2.3), let E be the corresponding extendability event. Conditioned on the same urs, independently sample $(\mathsf{pk}'_j, \mathsf{sk}'_j) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for $j \in [t]$ and set $\mathsf{localkey}_{[i]} = \mathsf{Combine}_\lambda(\mathsf{sk}_1, (\mathsf{pk}_j)_{2\leq j\leq i})$. Then, for every $i \in [\mathsf{n_p}]$, we have

$$I(\mathsf{localkey}_{[i]}; (\mathsf{pk}_j)_{j\in[i]}, (\mathsf{pk}'_j, \mathsf{sk}'_j)_{j\in[t]} \mid \mathsf{E}) \leq \mu_i,$$

and for every $i \in [\mathsf{n_p}]$ with $i > 1$, we have

$$\begin{aligned} &I(\mathsf{localkey}_{[i-1]}; (\mathsf{pk}_i, \mathsf{sk}_i) \mid (\mathsf{pk}_j)_{j\in[i-1]}, (\mathsf{pk}'_j, \mathsf{sk}'_j)_{j\in[t]}, \mathsf{E}) \\ &\leq I(\mathsf{localkey}_{[i-1]}; (\mathsf{pk}_i, \mathsf{sk}_i) \mid (\mathsf{pk}_j)_{j\in[i-1]}, (\mathsf{pk}'_j, \mathsf{sk}'_j)_{j\in[t]}, \mathsf{urs}, \mathsf{E}) + \mu_i. \end{aligned}$$

3 Fine-Grained Multi-Party NIKE based on $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$

In this section, we propose a fine-grained multi-party NIKE running in $\mathsf{AC}^0[2]$ and secure against NC^1 based on $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$. All arithmetic computations are over $GF(2)$ in this section.

3.1 Background on the NC^1-Fine-Grained Setting

We recall and define sampling procedures in NC^1, along with several lemmata.

Function families. We now recall the definitions of NC^1 circuits, $\mathsf{AC}^0[2]$ circuits, and $\oplus\mathsf{L/poly}$ circuits. Note that $\mathsf{AC}^0[2] \subsetneq \mathsf{NC}^1$ [35, 37].

Definition 3.1 (NC^1). The class of (non-uniform) NC^1 *function families* is the set of all function families $\mathcal{F} = \{f_\lambda\}$ for which there is a polynomial $p(\cdot)$ and constant c such that for each λ, f_λ can be computed by a (randomized) circuit of size $p(\lambda)$, depth $c\log(\lambda)$, and fan-in 2 using AND, OR, and NOT gates.

Definition 3.2 ($\mathsf{AC}^0[2]$). The class of (non-uniform) $\mathsf{AC}^0[2]$ *function families* is the set of all function families $\mathcal{F} = \{f_\lambda\}$ for which there is a polynomial $p(\cdot)$ and constant c such that for each λ, f_λ can be computed by a (randomized) circuit of size $p(\lambda)$, depth c, and unbounded fan-in using AND, OR, NOT, and PARITY gates.

Definition 3.3 ($\oplus\mathsf{L/poly}$). $\oplus\mathsf{L/poly}$ is the set of all boolean function families $\mathcal{F} = \{f_\lambda\}$ for which there is a constant c such that for each λ, there is a non-deterministic Turing machine $\mathcal{M}_\lambda$ such that for each input x with length λ, $\mathcal{M}_\lambda(x)$ uses at most $c\log(\lambda)$ space, and $f_\lambda(x)$ is equal to the parity of the number of accepting paths of $\mathcal{M}_\lambda(x)$.

We now recall the definitions of four sampling procedures $\{\mathsf{LSamp}_\lambda, \mathsf{RSamp}_\lambda, \mathsf{ZeroSamp}_\lambda, \mathsf{OneSamp}_\lambda\}$ in Figure 2. Note that the output of $\mathsf{ZeroSamp}_\lambda$ is always a matrix of rank $\lambda - 1$ and the output of $\mathsf{OneSamp}_\lambda$ is always a matrix of full rank [16].

$\underline{\mathsf{LSamp}_\lambda}$:
For all $i, j \in [\lambda]$ and $i < j$:
$r_{i,j} \xleftarrow{\$} \{0,1\}$
Return
$$\begin{pmatrix} 1 & r_{1,2} & \cdots & r_{1,\lambda-1} & r_{1,\lambda} \\ 0 & 1 & \cdots & r_{2,\lambda-1} & r_{2,\lambda} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & r_{\lambda-1,\lambda} \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

$\underline{\mathsf{RSamp}_\lambda}$:
For $i = 1, \cdots, \lambda - 1$
$r_i \xleftarrow{\$} \{0,1\}$
Return
$$\begin{pmatrix} 1 & 0 & \cdots & 0 & r_1 \\ 0 & 1 & \cdots & 0 & r_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & r_{\lambda-1} \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

$\underline{\mathsf{ZeroSamp}_\lambda}$:
$\mathbf{R}_0 \xleftarrow{\$} \mathsf{LSamp}_\lambda \in \{0,1\}^{\lambda\times\lambda}$
$\mathbf{R}_1 \xleftarrow{\$} \mathsf{RSamp}_\lambda \in \{0,1\}^{\lambda\times\lambda}$
Return $\mathbf{R}_0\mathbf{M}_0^\lambda\mathbf{R}_1 \in \{0,1\}^{\lambda\times\lambda}$

$\underline{\mathsf{OneSamp}_\lambda}$:
$\mathbf{R}_0 \xleftarrow{\$} \mathsf{LSamp}_\lambda$
$\mathbf{R}_1 \xleftarrow{\$} \mathsf{RSamp}_\lambda$
Return $\mathbf{R}_0\mathbf{M}_1^\lambda\mathbf{R}_1 \in \{0,1\}^{\lambda\times\lambda}$

Figure 2: Definitions of LSamp_λ, RSamp_λ, $\mathsf{ZeroSamp}_\lambda$, and $\mathsf{OneSamp}_\lambda$. By $\mathbf{M}_0^\lambda$ and $\mathbf{M}_1^\lambda$ we denote the $\lambda \times \lambda$ matrices $\mathbf{M}_0^\lambda = \begin{pmatrix} \mathbf{0} & 0 \\ \mathbf{I}_{\lambda-1} & \mathbf{0} \end{pmatrix}$ and $\mathbf{M}_1^\lambda = \begin{pmatrix} \mathbf{0} & 1 \\ \mathbf{I}_{\lambda-1} & \mathbf{0} \end{pmatrix}$ respectively.

Lemma 3.4 (Lemma 3 in [20]). *For all $\lambda \in \mathbb{N}$ and all $\mathbf{M} \in \mathsf{ZeroSamp}_\lambda$, it holds that $\mathrm{Ker}(\mathbf{M}) = \{\mathbf{0}, \mathbf{k}\}$ where $\mathbf{k}$ is a vector such that $\mathbf{k} \in \{0,1\}^{\lambda-1} \times \{1\}$.*

Lemma 3.5 (Lemma 4.3 in [16]). *If $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$, for any $\{\mathsf{adv}_\lambda\} \in \mathsf{NC}^1$ and all sufficiently large λ, we have*

$$|\Pr[\mathsf{adv}_\lambda(\mathbf{M}) = 1 \mid \mathbf{M} \xleftarrow{\$} \mathsf{ZeroSamp}_\lambda] - \Pr[\mathsf{adv}_\lambda(\mathbf{M}) = 1 \mid \mathbf{M} \xleftarrow{\$} \mathsf{OneSamp}_\lambda]| \leq \mathsf{negl}(\lambda).$$

Here, we follow [20, 39, 38] to adhere to the stronger notion of $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$ mentioned in the second paragraph of Remark 3.1 in [16] for the sake of simplicity. One can easily see that if we only assume the infinitely-often version of $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$, which holds only for an infinitely large number of values of λ, then our scheme, given later, satisfies infinitely-often security.

Next, we recall a sampling algorithm $\mathsf{RandSamp}_\lambda$ that is implicitly defined in [38] and presented in Figure 3. Additionally, an inverse algorithm $\mathsf{RandSamp}_\lambda^{-1}$ is introduced. Roughly, as proven in [38], the distribution of $(\mathbf{F}||\mathbf{s}) \in \{0,1\}^{\lambda(\lambda-1)} \times \{0,1\}^\lambda$ generated by $\mathsf{RandSamp}_\lambda(\mathbf{u})$ for $\mathbf{u} \xleftarrow{\$} \{0,1\}^{\lambda(\lambda+1)/2}$ is the same as $\mathsf{ZeroSamp}_\lambda$ (respectively, $\mathsf{OneSamp}_\lambda$) if $\mathbf{s}$ is in (respectively, outside) the span of $\mathbf{F}$, which happens with half probability. Hence, $\mathsf{RandSamp}_\lambda$ is indistinguishable from $\mathsf{ZeroSamp}_\lambda$ against NC^1 adversaries. One advantage of using $\mathsf{RandSamp}_\lambda$ is that it solely relies on public coins for matrix generation, and these public coins can be easily reconstructed from the matrices using $\mathsf{RandSamp}_\lambda^{-1}$. This will help us generate the public parameter of our construction given later by making use of a URS (i.e., $\mathbf{u}$), rather than generating it by using $\mathsf{OneSamp}_\lambda$ or $\mathsf{ZeroSamp}_\lambda$ with secret randomness.

Lemma 3.6 (Lemma 6 in [38]). *If $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$, for any $\{\mathsf{adv}_\lambda\} \in \mathsf{NC}^1$, we have*

$$|\Pr[\mathsf{adv}_\lambda\big(\mathsf{RandSamp}_\lambda(\mathbf{u})\big) = 1 \mid \mathbf{u} \xleftarrow{\$} \{0,1\}^{\lambda(\lambda+1)/2}] - \Pr[\mathsf{adv}_\lambda(\mathbf{M}) = 1 \mid \mathbf{M} \xleftarrow{\$} \mathsf{ZeroSamp}_\lambda]| \leq \mathsf{negl}(\lambda)$$

for all sufficiently large $\lambda \in \mathbb{N}$.

3.2 Our Construction

In this subsection, we give our construction of multi-party NIKE.

Core lemma. Before giving our construction, we define an algorithm SymR_λ outputting uniformly random symmetric matrices, and then propose a core lemma upon which we will heavily rely later.

$\underline{\mathsf{RandSamp}_\lambda(\mathbf{u} \in \{0,1\}^{\lambda(\lambda+1)/2})}$:
Parse $\mathbf{u} = (\mathbf{r}, \mathbf{s}) \in \{0,1\}^{\lambda(\lambda-1)/2} \times \{0,1\}^\lambda$
Parse $\mathbf{r} = (r_{i,j})_{1 \le i < j \le \lambda}$ and set

$$\mathbf{F} = \begin{pmatrix} r_{1,2} & r_{1,3} & \cdots & r_{1,\lambda-1} & r_{1,\lambda} \\ 1 & r_{2,3} & \cdots & r_{2,\lambda-1} & r_{2,\lambda} \\ 0 & 1 & \cdots & r_{3,\lambda-1} & r_{3,\lambda} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & r_{\lambda-1,\lambda} \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

Return $\mathbf{F}||\mathbf{s} \in \{0,1\}^{\lambda \times \lambda}$

$\underline{\mathsf{RandSamp}_\lambda^{-1}(\mathbf{M} \in \{0,1\}^{\lambda \times \lambda})}$:
Parse $\mathbf{M} = \mathbf{F}||\mathbf{s}$ where

$$\mathbf{F} = \begin{pmatrix} r_{1,2} & r_{1,3} & \cdots & r_{1,\lambda-1} & r_{1,\lambda} \\ 1 & r_{2,3} & \cdots & r_{2,\lambda-1} & r_{2,\lambda} \\ 0 & 1 & \cdots & r_{3,\lambda-1} & r_{3,\lambda} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & r_{\lambda-1,\lambda} \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

Abort if $\mathbf{M}$ does not conform to the form
Set $\mathbf{r} = (r_{i,j})_{1 \le i < j \le \lambda}$
Return $\mathbf{u} = (\mathbf{r}, \mathbf{s}) \in \{0,1\}^{\lambda(\lambda+1)/2}$

Figure 3: The definition of $\mathsf{RandSamp}_\lambda$ and $\mathsf{RandSamp}_\lambda^{-1}$.

$\underline{\mathsf{SymR}_\lambda}$:

Output a matrix $\mathbf{R} \in \{0,1\}^{\lambda \times \lambda}$ given by $\mathbf{R}_{ij} = \begin{cases} \text{a random bit, if } i \le j \\ \mathbf{R}_{ji}, \text{if } i > j \end{cases}$

Figure 4: The definition of SymR_λ.

Lemma 3.7 *If* $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$ *holds, then for any* $\{\mathsf{adv}_\lambda\} \in \mathsf{NC}^1$*, any sufficiently large* $\lambda \in \mathbb{N}$*, and any polynomial* $n = n(\lambda)$ *in* λ*, we have:*

$$|\Pr[\mathsf{adv}_\lambda(\mathbf{M}, (\widetilde{\mathbf{M}}\mathbf{R}_i)_{i\in[n]}) = 1] - \Pr[\mathsf{adv}_\lambda(\mathbf{M}, (\widetilde{\mathbf{M}}\mathbf{R}_i + \mathbf{I}_\lambda)_{i\in[n]}) = 1]| \le \mathsf{negl}(\lambda)$$

where $\mathbf{M} \xleftarrow{\$} \mathsf{ZeroSamp}_\lambda$*,* $\widetilde{\mathbf{M}} = \mathbf{M}^\top\mathbf{M}$ *and* $\mathbf{R}_1, \cdots, \mathbf{R}_n \xleftarrow{\$} \mathsf{SymR}_\lambda$*.*

Proof. Let $\lambda \in \mathbb{N}$ be any sufficiently large security parameter. Let $\mathbf{M} \xleftarrow{\$} \mathsf{ZeroSamp}_\lambda$, $\mathbf{M}' \xleftarrow{\$} \mathsf{OneSamp}_\lambda$, $\widetilde{\mathbf{M}} = \mathbf{M}^\top\mathbf{M}$, and $\widetilde{\mathbf{M}}' = \mathbf{M}'^\top\mathbf{M}'$. Since the multiplication of two matrices can be performed in NC^1, according to Lemma 3.5, we immediately have $|\Pr[\mathsf{adv}_\lambda(\mathbf{M}, \widetilde{\mathbf{M}}) = 1] - \Pr[\mathsf{adv}_\lambda(\mathbf{M}', \widetilde{\mathbf{M}}') = 1]| \le \mathsf{negl}(\lambda)$, i.e.,

$$|\Pr[\mathsf{adv}_\lambda(\mathbf{M}, (\widetilde{\mathbf{M}}\mathbf{R}_i)_{i\in[n]}) = 1] - \Pr[\mathsf{adv}_\lambda(\mathbf{M}', (\widetilde{\mathbf{M}}'\mathbf{R}_i)_{i\in[n]}) = 1]| \le \mathsf{negl}(\lambda), \tag{1}$$

$$|\Pr[\mathsf{adv}_\lambda(\mathbf{M}', (\widetilde{\mathbf{M}}'\mathbf{R}_i + \mathbf{I}_\lambda)_{i\in[n]}) = 1] - \Pr[\mathsf{adv}_\lambda(\mathbf{M}, (\widetilde{\mathbf{M}}\mathbf{R}_i + \mathbf{I}_\lambda)_{i\in[n]}) = 1]| \le \mathsf{negl}(\lambda), \tag{2}$$

where $\mathbf{R}_1, \cdots, \mathbf{R}_n \xleftarrow{\$} \mathsf{SymR}_\lambda$.

Recall that any $\mathbf{M}' \in \mathsf{OneSamp}_\lambda$ is of full rank and has an inverse matrix $\mathbf{M}'^{-1}$. Hence, $\widetilde{\mathbf{M}}' = \mathbf{M}'^\top\mathbf{M}'$ also has an inverse matrix $\widetilde{\mathbf{M}}'^{-1} = \mathbf{M}'^{-1}\mathbf{M}'^{-1^\top}$ and is of full rank. Furthermore, since $\widetilde{\mathbf{M}}'^{-1}$ is a symmetric matrix, for all $i \in [n]$, the following distributions are identical:

$$(\mathbf{R}_i + \widetilde{\mathbf{M}}'^{-1})_{i\in[n]} \text{ and } (\mathbf{R}_i)_{i\in[n]},$$

where $\mathbf{R}_1, \cdots, \mathbf{R}_n \xleftarrow{\$} \mathsf{SymR}_\lambda$. Hence, the following two distributions are also identical:

$$\left(\mathbf{M}', (\widetilde{\mathbf{M}}'\mathbf{R}_i)_{i\in[n]}\right) \text{ and } \left(\mathbf{M}', (\widetilde{\mathbf{M}}'(\mathbf{R}_i + (\widetilde{\mathbf{M}}')^{-1}))_{i\in[n]} = (\widetilde{\mathbf{M}}'\mathbf{R}_i + \mathbf{I})_{i\in[n]}\right).$$

Combining this with Equations (1) and (2), Lemma 3.7 immediately follows. □

Construction. We now give our construction of multi-party NIKE NCNIKE in the bounded-parallel-time model in Figure 5.

Theorem 3.8 *If* $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$ *holds, then for any constant* $\mathsf{n_p}$*,* NCNIKE *is an* $\mathsf{AC}^0[2]$-$\mathsf{n_p}$*-NIKE with* 0*-correctness (i.e., perfect correctness) and* NC^1-$\mathsf{negl}(\lambda)$*-security.*

$\underline{\mathsf{Gen}_\lambda(\mathsf{urs} \in \{0,1\}^{\lambda(\lambda+1)/2})}$:	$\underline{\mathsf{Share}_\lambda(i, \mathsf{sk}_i, (\mathsf{pk}_j)_{j\in[\mathsf{n_p}]})}$:
$\mathbf{M} = \mathsf{RandSamp}_\lambda(\mathsf{urs})$	Parse $(\mathsf{pk}_j)_{j\in[\mathsf{n_p}]} = (\mathbf{P}_j)_{j\in[\mathsf{n_p}]}$ and $\mathsf{sk}_i = \mathbf{R}_i$
$\widetilde{\mathbf{M}} = \mathbf{M}^\top\mathbf{M}$	$\mathbf{K} = (\prod_{j<i}\mathbf{P}_j^\top)\mathbf{R}_i(\prod_{j>i}\mathbf{P}_j)$
$\mathsf{sk} = \mathbf{R} \xleftarrow{\$} \mathsf{SymR}_\lambda$, $\mathsf{pk} = \widetilde{\mathbf{M}}\mathbf{R}$	Let key be the bottom-right bit of $\mathbf{K}$
Return $(\mathsf{pk}, \mathsf{sk})$	return key

Figure 5: Definition of $\mathsf{NCNIKE} = \{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\}$.

Proof. **Complexity.** First we note that $\{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\}$ are computable in $\mathsf{AC}^0[2]$ since they only involve operations including multiplication of a constant number of matrices and sampling random bits.
Correctness. Let $(\mathbf{P}_i = \widetilde{\mathbf{M}}\mathbf{R}_i, \mathbf{R}_i)_{i\in[\mathsf{n_p}]}$ be $\mathsf{n_p}$ public/secret key pairs generated by Gen_λ. For all $i \in [\mathsf{n_p}]$, we have $\mathbf{P}_i^\top = (\widetilde{\mathbf{M}}\mathbf{R}_i)^\top = \mathbf{R}_i^\top\widetilde{\mathbf{M}}^\top = \mathbf{R}_i\widetilde{\mathbf{M}}$. Moreover, for any $i \in [\mathsf{n_p}]$, we have

$$\prod_{j<i}\mathbf{P}_j^\top\mathbf{R}_i\prod_{j>i}\mathbf{P}_j = \prod_{j<i}(\mathbf{R}_j\widetilde{\mathbf{M}})\mathbf{R}_i\prod_{j>i}\mathbf{P}_j = \mathbf{R}_1\prod_{1<j\le i}(\widetilde{\mathbf{M}}\mathbf{R}_j)\prod_{j>i}\mathbf{P}_j = \mathbf{R}_1\prod_{j>1}\mathbf{P}_j.$$

Therefore, for all $i, j \in [\mathsf{n_p}]$, we have

$$\mathsf{Share}_\lambda(i, \mathsf{sk}_i, (\mathsf{pk}_j)_{j\in[\mathsf{n_p}]}) = \mathsf{Share}_\lambda(j, \mathsf{sk}_j, (\mathsf{pk}_j)_{j\in[\mathsf{n_p}]}) = \mathsf{key},$$

where key is the bottom-right bit of $\mathbf{R}_1\prod_{j>1}\mathbf{P}_j$, completing the proof of correctness.

Security. Let λ be the security parameter and $\mathcal{A} = \{\mathsf{adv}_\lambda\} \in \mathsf{NC}^1$ be any adversary against the security of NCNIKE. We prove the security of NCNIKE via a sequence of hybrid games as in Figure 6.

Games G_0, $\boxed{\mathsf{G}_1}$, $\boxed{\mathsf{G}_2}$, G_3, G_4

$\mathbf{M} = \mathsf{RandSamp}_\lambda(\mathsf{urs})$ where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^{\lambda(\lambda+1)/2}$ $\boxed{\mathbf{M} \xleftarrow{\$} \mathsf{ZeroSamp}_\lambda,\ \mathsf{urs} = \mathsf{RandSamp}_\lambda^{-1}(\mathbf{M})}$

$\widetilde{\mathbf{M}} = \mathbf{M}^\top\mathbf{M}$

$\mathbf{R}_i \xleftarrow{\$} \mathsf{SymR}_\lambda$ for $i = 1, \cdots, \mathsf{n_p}$

$\mathbf{P}_i = \widetilde{\mathbf{M}}\mathbf{R}_i$ for $i = 1, \cdots, \mathsf{n_p}$ $\boxed{\mathbf{P}_1 = \widetilde{\mathbf{M}}\mathbf{R}_1,\ \mathbf{P}_i = \widetilde{\mathbf{M}}\mathbf{R}_i + \mathbf{I}_\lambda \text{ for } i = 2, \cdots, \mathsf{n_p}}$

$\mathbf{K} = \mathbf{R}_1\mathbf{P}_2\cdots\mathbf{P}_{\mathsf{n_p}}$ $\mathbf{K} = (\mathbf{R}_1 + \mathbf{m}\cdot\mu\cdot\mathbf{m}^\top)\mathbf{P}_2\cdots\mathbf{P}_{\mathsf{n_p}}$

Set key as the bottom-right bit of $\mathbf{K}$ $\mathsf{key} \xleftarrow{\$} \{0,1\}$

Return $(\mathsf{urs}, (\mathsf{pk}_i = \mathbf{P}_i)_{i\in[\mathsf{n_p}]}, \mathsf{key})$ to adv_λ

Output whatever adv_λ outputs

Figure 6: The challengers in G_0, G_1, G_2, G_3, and G_4 for the proof of Theorem 3.8. $\mathbf{m}$ is the non-zero vector in $\mathrm{Ker}(\mathbf{M})$ guaranteed by Lemma 3.4.

G_0. This is the original security game where adv_λ receives $\mathsf{urs} \xleftarrow{\$} \{0,1\}^{\lambda(\lambda+1)/2}$, $(\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}$ where $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for all $i \in [\mathsf{n_p}]$, and $\mathsf{key} = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]})$, and outputs a bit β.

G_1. G_1 and G_0 only differ in the way that $\mathbf{M}$ is generated, namely, $\mathbf{M}$ is generated by $\mathsf{ZeroSamp}_\lambda$ rather than $\mathsf{RandSamp}_\lambda$ in G_1.

Lemma 3.9 $|\Pr[\mathsf{G}_1^{\mathsf{adv}_\lambda} \Rightarrow 1] - \Pr[\mathsf{G}_0^{\mathsf{adv}_\lambda} \Rightarrow 1]| \le \mathsf{negl}(\lambda)$.

Proof. We construct an adversary $\mathcal{B}^0 = \{\mathsf{advb}_\lambda^0\} \in \mathsf{NC}^1$ in the game described in Lemma 3.6 as follows.

On receiving $\mathbf{M}$ sampled as $\mathbf{M} = \mathsf{RandSamp}_\lambda(\mathsf{urs})$ where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^{\lambda(\lambda+1)/2}$ or $\mathbf{M} \xleftarrow{\$} \mathsf{ZeroSamp}_\lambda$, advb_λ^0 sets $\mathsf{urs} = \mathsf{RandSamp}_\lambda^{-1}(\mathbf{M})$, runs $\mathbf{R}_i \xleftarrow{\$} \mathsf{SymR}_\lambda$ and $\mathbf{P}_i = \widetilde{\mathbf{M}}\mathbf{R}_i$ for all $i = 1, \cdots, \mathsf{n_p}$, computes

$\mathbf{K} = \mathbf{R}_1 \prod_{i>1} \mathbf{P}_i$, and sets key as the bottom-right bit of $\mathbf{K}$. Next it sends $(\mathsf{urs}, (\mathbf{P}_i)_{i\in[\mathsf{n_p}]}, \mathsf{key})$ to adv_λ. When adv_λ returns $\beta \in \{0,1\}$, advb^0_λ returns β as well.

First we note that all operations in advb^0_λ are NC^1, since advb^0_λ only samples random bits, computes multiplication of a constant number of matrices, and runs adv_λ. Moreover, when $\mathbf{M}$ is generated as $\mathbf{M} = \mathsf{RandSamp}_\lambda(\mathsf{urs})$ where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^{\lambda(\lambda+1)/2}$ (respectively, $\mathbf{M} \xleftarrow{\$} \mathsf{ZeroSamp}_\lambda$), the view of adv_λ is identical to its view in G_0 (respectively, G_1). Hence, the advantage of advb^0_λ is

$$|\Pr[\mathsf{G}_1^{\mathsf{adv}_\lambda} \Rightarrow 1] - \Pr[\mathsf{G}_0^{\mathsf{adv}_\lambda} \Rightarrow 1]|,$$

which is negligible according to Lemma 3.6, completing this part of the proof. □

G_2. G_2 and G_1 only differ in the way that $\mathbf{P}_2, \cdots, \mathbf{P}_{\mathsf{n_p}}$ are generated, namely, $\mathbf{P}_i$ is generated as $\mathbf{P}_i = \widetilde{\mathbf{M}}\mathbf{R}_i + \mathbf{I}_\lambda$ where $\mathbf{R}_i \xleftarrow{\$} \mathsf{SymR}_\lambda$ for $i = 2, \cdots, \mathsf{n_p}$ in G_2.

Lemma 3.10 $|\Pr[\mathsf{G}_2^{\mathsf{adv}_\lambda} \Rightarrow 1] - \Pr[\mathsf{G}_1^{\mathsf{adv}_\lambda} \Rightarrow 1]| \leq \mathsf{negl}(\lambda)$.

Proof. We construct an adversary $\mathcal{B}^1 = \{\mathsf{advb}^1_\lambda\} \in \mathsf{NC}^1$ in the game described in Lemma 3.7 as follows.

On receiving $\mathbf{M} \xleftarrow{\$} \mathsf{ZeroSamp}_\lambda$ and $\mathbf{P}_i = \widetilde{\mathbf{M}}\mathbf{R}_i$ or $\mathbf{P}_i = \widetilde{\mathbf{M}}\mathbf{R}_i + \mathbf{I}_\lambda$, where $\widetilde{\mathbf{M}} = \mathbf{M}^\top\mathbf{M}$ and $\mathbf{R}_i \xleftarrow{\$} \mathsf{SymR}_\lambda$, for $i = 2, \cdots, \mathsf{n_p}$, advb^1_λ sets $\mathsf{urs} = \mathsf{RandSamp}^{-1}_\lambda(\mathbf{M})$, samples $\mathbf{R}_1 \xleftarrow{\$} \mathsf{SymR}_\lambda$, sets $\mathbf{P}_1 = \widetilde{\mathbf{M}}\mathbf{R}_1$, computes $\mathbf{K} = \mathbf{R}_1 \prod_{i=2}^{\mathsf{n_p}} \mathbf{P}_i$, and sets key as the bottom-right bit of $\mathbf{K}$. Next it sends $(\mathsf{urs}, (\mathbf{P}_i)_{i\in[\mathsf{n_p}]}, \mathsf{key})$ to adv_λ. When adv_λ returns $\beta \in \{0,1\}$, advb^1_λ returns β as well.

First we note that all operations in advb^1_λ are NC^1, since advb^1_λ only samples random bits, computes multiplication of a constant number of matrices, and runs adv_λ. Moreover, when $\mathbf{P}_i$ is generated as $\mathbf{P}_i = \widetilde{\mathbf{M}}\mathbf{R}_i$ (respectively, $\mathbf{P}_i = \widetilde{\mathbf{M}}\mathbf{R}_i + \mathbf{I}_\lambda$) for all $i \in [\mathsf{n_p}]\backslash\{1\}$, the view of adv_λ is identical to its view in G_1 (respectively, G_2). Hence, the advantage of advb^1_λ is

$$|\Pr[\mathsf{G}_2^{\mathsf{adv}_\lambda} \Rightarrow 1] - \Pr[\mathsf{G}_1^{\mathsf{adv}_\lambda} \Rightarrow 1]|,$$

which is negligible according to Lemma 3.7, completing this part of the proof. □

G_3. G_3 is exactly the same as G_2 except that when generating $\mathbf{K}$, we replace $\mathbf{R}_1$ by $(\mathbf{R}_1 + \mathbf{m} \cdot \mu \cdot \mathbf{m}^\top)$. Here, $\mu \xleftarrow{\$} \{0,1\}$ and $\mathbf{m} \in \{0,1\}^{\lambda-1} \times \{1\}$ is the non-zero vector in $\mathrm{Ker}(\mathbf{M})$, which must exist according to Lemma 3.4.

Lemma 3.11 $\Pr[\mathsf{G}_3^{\mathsf{adv}_\lambda} \Rightarrow 1] = \Pr[\mathsf{G}_2^{\mathsf{adv}_\lambda} \Rightarrow 1]$.

Proof. First we note that the distribution of $\mathbf{R}_1 + \mathbf{m} \cdot \mu \cdot \mathbf{m}^\top$ is uniform in SymR_λ for $\mathbf{R}_1 \xleftarrow{\$} \mathsf{SymR}_\lambda$, since $\mathbf{m} \cdot \mu \cdot \mathbf{m}^\top$ is a symmetric matrix. Moreover, we have $\mathbf{P}_1 = \widetilde{\mathbf{M}}(\mathbf{R}_1 + \mathbf{m} \cdot \mu \cdot \mathbf{m}^\top) = \widetilde{\mathbf{M}}\mathbf{R}_1$. Then Lemma 3.11 immediately follows from the fact that the view of adv_λ in G_3 is identical to its view in G_2. □

G_4. G_4 is exactly the same as G_3 except that the challenger sets the session key key as a random bit.

Lemma 3.12 $\Pr[\mathsf{G}_4^{\mathsf{adv}_\lambda} \Rightarrow 1] = \Pr[\mathsf{G}_3^{\mathsf{adv}_\lambda} \Rightarrow 1]$.

Proof. First we note that $\mathbf{P}_2 = \widetilde{\mathbf{M}}\mathbf{R}_2 + \mathbf{I}_\lambda$ in G_3 and G_4. Assuming that $\prod_{j=2}^{i} \mathbf{P}_j = \widetilde{\mathbf{M}}\mathbf{S}_i + \mathbf{I}_\lambda$ for some $2 \leq i < \mathsf{n_p}$ and some $\mathbf{S}_i \in \{0,1\}^{\lambda\times\lambda}$, we have

$$\begin{aligned}\prod_{j=2}^{i+1} \mathbf{P}_j &= (\widetilde{\mathbf{M}}\mathbf{S}_i + \mathbf{I}_\lambda)\mathbf{P}_{i+1} = \widetilde{\mathbf{M}}\mathbf{S}_i\mathbf{P}_{i+1} + \mathbf{P}_{i+1} \\ &= \widetilde{\mathbf{M}}\mathbf{S}_i\mathbf{P}_{i+1} + \widetilde{\mathbf{M}}\mathbf{R}_{i+1} + \mathbf{I}_\lambda = \widetilde{\mathbf{M}}(\mathbf{S}_i\mathbf{P}_{i+1} + \mathbf{R}_{i+1}) + \mathbf{I}_\lambda,\end{aligned}$$

namely, we have $\prod_{j=2}^{i+1} \mathbf{P}_j = \widetilde{\mathbf{M}}\mathbf{S}_{i+1} + \mathbf{I}_\lambda$ for $\mathbf{S}_{i+1} = \mathbf{S}_i\mathbf{P}_{i+1} + \mathbf{R}_{i+1}$.

Then by mathematical induction, we have $\prod_{j=2}^{\mathsf{n_p}} \mathbf{P}_j = \widetilde{\mathbf{M}}\mathbf{S}_{\mathsf{n_p}} + \mathbf{I}_\lambda$ for some $\mathbf{S}_{\mathsf{n_p}} \in \{0,1\}^{\lambda\times\lambda}$, i.e., the rightmost column vector in $\prod_{j=2}^{\mathsf{n_p}} \mathbf{P}_j$ is in the form of $\widetilde{\mathbf{M}}\mathbf{s}_k + \mathbf{e}_\lambda$ for some $\mathbf{s}_k \in \{0,1\}^\lambda$ and $\mathbf{e}_\lambda = (0, \cdots, 0, 1)^\top$. Moreover, since the last bit in $\mathbf{m}$ is 1 according to Lemma 3.4, the bottom row vector of

$\mathbf{R}_1 + \mathbf{m} \cdot \mu \cdot \mathbf{m}^\top$ must be in the form of $\mathbf{r}_1^\top + \mu \cdot \mathbf{m}^\top$, where $\mathbf{r}_1^\top$ denotes the bottom row vector in $\mathbf{R}_1$. Let key be the bottom-right bit of $\mathbf{K}$ in G_3. Since $\mathbf{m} \in \mathrm{Ker}(\mathbf{M})$, we have $\mathbf{m}^\top \widetilde{\mathbf{M}} = (\mathbf{M}\mathbf{m})^\top \mathbf{M} = \mathbf{0}^\top$. We have

$$\mathsf{key} = (\mathbf{r}_1^\top + \mu \cdot \mathbf{m}^\top)(\widetilde{\mathbf{M}}\mathbf{s}_k + \mathbf{e}_\lambda) = \mathbf{r}_1^\top(\widetilde{\mathbf{M}}\mathbf{s}_k + \mathbf{e}_\lambda) + \mu \cdot \underbrace{\mathbf{m}^\top \widetilde{\mathbf{M}}\mathbf{s}_k}_{=0} + \mu \cdot \underbrace{\mathbf{m}^\top \mathbf{e}_\lambda}_{=1}.$$

Since μ is a random bit and adv_λ obtains no information on μ except for key, key is a random bit in the view of adv_λ as well, completing this part of the proof. □

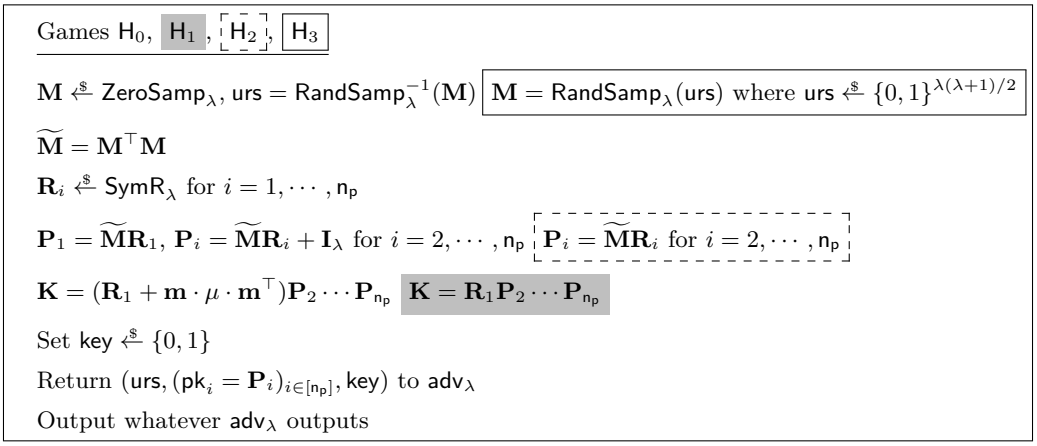

Figure 7: The challengers in H_0, H_1, H_2, and H_3 for the proof of Theorem 3.8. $\mathbf{m}$ is the non-zero vector in $\mathrm{Ker}(\mathbf{M})$ guaranteed by Lemma 3.4.

We now do all the previous steps in the reverse order as in Figure 7. Note that the view of the adversary in H_0 (respectively, H_3) is identical to its view in G_4 (respectively, the honest game where key is a random bit). By using the above arguments in a reverse order, we have the following lemma.

Lemma 3.13 $|\Pr[\mathsf{H}_3^{\mathsf{adv}_\lambda} \Rightarrow 1] - \Pr[\mathsf{H}_0^{\mathsf{adv}_\lambda} \Rightarrow 1]| \leq \mathsf{negl}(\lambda)$.

Putting all the above together, Theorem 3.8 immediately follows. □

The following corollary follows immediately from Theorem 3.8 by running the construction in Figure 5 $q(\lambda)$ times in parallel.

Corollary 3.14 *If* $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L}/\mathsf{poly}$ *holds, then for any constant* $\mathsf{n_p}$ *and any polynomial* $q = q(\lambda)$*, there exists an* $\mathsf{AC}^0[2]$-$\mathsf{n_p}$*-NIKE with* 0*-correctness and* NC^1-$\mathsf{negl}(\lambda)$*-security whose shared key has length* $q(\lambda)$*. Moreover, its public keys, secret keys, and URS have length* $\mathsf{poly}(\lambda)$*.*

Extension to IB-NIKE. We can also show that our construction satisfies 0-restricted extendability (see Definition 2.3), and thus can be converted into a restricted BPRF, which in turn implies a multi-party IB-NIKE in the bounded-parallel-time model. A similar argument can also be made for our constructions in the bounded-time model and the BSM, while those constructions further satisfy (full) extendability (see Definition 2.2) and can be converted into full-fledged BPRFs. We refer the reader to Section 6 for the full details.

4 Fine-Grained Multi-Party NIKE based on Zero-k-Clique

In this section, we propose a fine-grained multi-party NIKE based on Zero-k-Clique. All algorithms and adversaries in this section are modeled as word-RAMs with $O(\log \lambda)$-bit words. The resulting scheme is computable in $\tilde{O}(\lambda^{\mathsf{n_p}+k-1})$ and secure against adversaries running in time $\tilde{O}(\lambda^{\mathsf{n_p}+k-\Omega(1)})$ for any constant number $\mathsf{n_p}$ of parties.

4.1 Zero-k-Clique Problem and Its Properties

We now introduce the Zero-k-Clique problem, which is the set of k-partite graphs with weighted edges. Each graph is said to have a solution if there is a k-clique (i.e., a complete subgraph) where the sum of edge weights within the clique is 0. Below, for any two nodes v and u, $\mathsf{weight}(v,u)$ denotes the weight of the edge (v,u).

Definition 4.1 (Zero-k-Clique-R problem). Let $\lambda \in \mathbb{N}$ be the security parameter. For $k > 2$, a *Zero-k-Clique-R problem* $P_{k,R}$ is the set of all k-partite graphs with λ nodes in each partition and the weight of each edge being in $\mathbb{Z}_R$. Each graph with k partitions $\mathcal{P}_1, \cdots, \mathcal{P}_k$ is required to be complete, namely, for every $1 \le i < j \le k$, there is an edge between two nodes $v \in \mathcal{P}_i$ and $u \in \mathcal{P}_j$. A tuple $(v_1, \cdots, v_k) \in \mathcal{P}_1 \times \cdots \times \mathcal{P}_k$ is a solution of I if $\sum_{1 \le i < j \le k} \mathsf{weight}(v_i, v_j) \equiv 0 \pmod R$.

For any $P_{k,R}$, we define two distributions $\mathcal{D}_0[R]$ and $\mathcal{D}_1[R]$. $\mathcal{D}_0[R]$ (respectively, $\mathcal{D}_1[R]$) is the uniform distribution over the instances in $P_{k,R}$ having no solutions (respectively, having a single solution).

The Zero-k-Clique-R hypothesis says that it is easy to verify a solution of a random instance in the Zero-k-Clique problem and difficult to find a solution given a random instance.

Definition 4.2 (Strong Zero-k-Clique-R hypothesis and range $R = \lambda^{ck}$). For any constant $c > 1$, $\lambda \in \mathbb{N}$, and $R = \lambda^{ck}$, the *strong Zero-k-Clique-R hypothesis* holds if

- there exists an algorithm Verify with running time $O(\lambda^{k(1-\delta)})$ for some constant $\delta > 0$ such that $\mathsf{Verify}(I, \mathsf{w}) = 1$ iff w is a valid witness for I,
- for any algorithm $\mathcal{A} = \{\mathsf{adv}_\lambda\}$ with running time $O(\lambda^{k(1-\delta)})$ for some constant $\delta > 0$, $\Pr[\mathsf{adv}_\lambda(I) = \mathsf{w}] \le 1/100$ where $I \xleftarrow{\$} \mathcal{D}_1[R]$ and w denotes the unique valid witness of I.

Remark. Below when mentioning the strong Zero-k-Clique hypothesis, we are always referring to the strong Zero-k-Clique hypothesis with $k > 2$ and $R = \lambda^{ck}$.

Theorem 4.3 (Theorem 15 in [29]). *If the strong Zero-k-Clique-R hypothesis holds, then for every probabilistic word-RAM family $\{\mathsf{adv}_\lambda\}$ with running time $O((\ell(\lambda)T(\lambda))^{1-\delta})$ for some constant $\delta > 0$, we have $\Pr[\mathsf{adv}_\lambda((I_j)_{j=1}^{\ell(\lambda)}) = i] \le \epsilon$, where $I_i \xleftarrow{\$} \mathcal{D}_1[R]$ for $i \xleftarrow{\$} [\ell(\lambda)]$, $I_j \xleftarrow{\$} \mathcal{D}_0[R]$ for all $j \in [\ell(\lambda)] \backslash \{i\}$, $\ell(\lambda) = \lambda^{\Omega(1)}$, $\epsilon \le 1/200$, and $T(\lambda) = \lambda^k$.*

Next we recall a word-RAM family $\{\mathsf{Generate}_\lambda\}$ such that $\mathsf{Generate}_\lambda(R,k,b)$ produces an instance of the Zero-k-Clique-R problem drawn from a distribution within total variation distance at most $2\lambda^k/R$ of $\mathcal{D}_b[R]$ defined in [29] as in Figure 8.

$\underline{\mathsf{Generate}_\lambda(R,k,b)}$:

Generate $I \in P_{k,R}$ with partitions $\mathcal{P}_1, \cdots, \mathcal{P}_k$ such that $\mathsf{weight}(v_i, v_j) \xleftarrow{\$} \mathbb{Z}_R$ for all $1 \le i < j \le k$, all $v_i \in \mathcal{P}_i$, and all $v_j \in \mathcal{P}_j$

If $b = 0$, return I

Otherwise, sample $i, j \xleftarrow{\$} [k] \wedge 1 \le i < j \le k$, $v_1 \xleftarrow{\$} \mathcal{P}_1, \cdots, v_k \xleftarrow{\$} \mathcal{P}_k$ and set $\mathsf{weight}(v_i, v_j) = -\sum_{(i',j') \neq (i,j)} \mathsf{weight}(v_{i'}, v_{j'})$

Return I

Figure 8: Definition of $\mathsf{Generate}_\lambda$ [29].

Theorem 4.4 (Theorem 14 in [29]). *If the strong Zero-k-Clique-R hypothesis holds, then we have $\Delta(\mathsf{Generate}_\lambda(R,k,b), \mathcal{D}_b[R]) \le 2\lambda^k/R$, for all $b \in \{0,1\}$, where $\mathsf{Generate}_\lambda$ runs in time $O(\lambda^2)$.*

The following corollary follows immediately from Theorem 4.4.

Corollary 4.5 *If the strong Zero-k-Clique-R hypothesis holds, we have*

$$\Delta((I_i^j)_{j \in [\ell], i \in [\mathsf{n_p}]}, (\mathcal{D}_{b_i^j}[R])_{j \in [\ell], i \in [\mathsf{n_p}]}) \le \sum_{j \in [\ell], i \in [\mathsf{n_p}]} \Delta(I_i^j, \mathcal{D}_{b_i^j}[R]) \le 2\ell \mathsf{n_p} \lambda^k / R,$$

where $I_i^j \xleftarrow{\$} \mathsf{Generate}_\lambda(R,k,b_i^j)$, $b_i^j \in \{0,1\}$ for all $i \in [\mathsf{n_p}]$ and $j \in [\ell]$, $\ell = \lambda^{\Omega(1)}$, and $\mathsf{n_p}$ is any constant.

We now recall a theorem given in [29]. Roughly, it says that there exists a word-RAM family $\{\mathsf{Split}_\lambda\}$ such that for an instance I of the Zero-k-Clique-R problem with one solution, $\mathsf{Split}_\lambda(I)$ outputs two slightly smaller instances that both have solutions in $O(\lambda^{k(1-\delta)})$ time for some constant $\delta > 0$.

Theorem 4.6 (Theorem 16 in [29]). *If the strong Zero-k-Clique-R hypothesis holds, then there exists a word-RAM family $\{\mathsf{Split}_\lambda\}$ with running time $O(\lambda^{k(1-\delta)})$ for some constant $\delta > 0$ such that*

- $$\Delta((\hat{I}_1^i, \hat{I}_2^i), (\mathcal{D}_0[\sqrt{R}] \times \mathcal{D}_0[\sqrt{R}])) \leq \binom{k}{2} \cdot 4^{\binom{k}{2}} \cdot 3\lambda^k/\sqrt{R}$$
  for all $i \in [\binom{k}{2}]$ where $I \xleftarrow{\$} \mathcal{D}_0[R]$, $(\hat{I}_1^j, \hat{I}_2^j)_{j\in[\binom{k}{2}]} \xleftarrow{\$} \mathsf{Split}_\lambda(I)$.
- $$\Delta((\hat{I}_1^i, \hat{I}_2^i), (\mathcal{D}_1[\sqrt{R}] \times \mathcal{D}_1[\sqrt{R}])) \leq \binom{k}{2} \cdot 4^{\binom{k}{2}} \cdot 3\lambda^k/\sqrt{R}$$
  for some $i \in [\binom{k}{2}]$ where $I \xleftarrow{\$} \mathcal{D}_1[R]$, $(\hat{I}_1^j, \hat{I}_2^j)_{j\in[\binom{k}{2}]} \xleftarrow{\$} \mathsf{Split}_\lambda(I)$.

4.2 Construction with Weak Security

We now construct a multi-party NIKE that satisfies weak security. Before showing our construction, we first extend the binary splitting algorithm to an $\mathsf{n_p}$-party splitting algorithm.

Generalized splitting algorithm. For any constant $\mathsf{n_p}$, let $L = \lceil \log_2 \mathsf{n_p} \rceil$, $\bar{\mathsf{n}}_\mathsf{p} = 2^L$, and $c = \binom{k}{2}$. The algorithm applies Split_λ defined in Theorem 4.6 for L levels. Each call to Split_λ returns c pairs of instances, and the algorithm keeps all possible choices of one returned pair from each call. Therefore, after L levels, it outputs $m_\mathsf{split} = c^{\bar{\mathsf{n}}_\mathsf{p}-1}$ lists, each containing $\bar{\mathsf{n}}_\mathsf{p}$ instances. The output of $\mathsf{n_p}$-Split_λ is obtained by keeping the first $\mathsf{n_p}$ instances in each list. Let $\mathsf{n_p} \geq 2$ and $k > 2$ be any constants.

$\underline{\mathsf{n_p}\text{-}\mathsf{Split}_\lambda(I)}$:
Let $c = \binom{k}{2}$, $L = \lceil \log_2 \mathsf{n_p} \rceil$, and let $\mathcal{I}_0$ be the ordered list containing (I)
For $r \in \{0, \ldots, L-1\}$
  Let $\mathcal{I}_{r+1}$ be the empty ordered list
  For every $(I_1, \ldots, I_{2^r})$ in $\mathcal{I}_r$
    For every $i \in [2^r]$, let $(\hat{I}^a_{i,1}, \hat{I}^a_{i,2})_{a\in[c]} = \mathsf{Split}_\lambda(I_i)$
    For every $(a_1, \ldots, a_{2^r}) \in [c]^{2^r}$ append $(\hat{I}^{a_1}_{1,1}, \hat{I}^{a_1}_{1,2}, \ldots, \hat{I}^{a_{2^r}}_{2^r,1}, \hat{I}^{a_{2^r}}_{2^r,2})$ to $\mathcal{I}_{r+1}$
Return $(I_1^t, \ldots, I_{\mathsf{n_p}}^t)_{t\in[m_\mathsf{split}]}$ where $\mathcal{I}_L = (I_1^t, \ldots, I_{2^L}^t)_{t\in[m_\mathsf{split}]}$.

Figure 9: The construction of $\mathsf{n_p}$-Split_λ. Let $\{\mathsf{Split}_\lambda\}$ be the word-RAM family from Theorem 4.6.

Theorem 4.7 *If the strong Zero-k-Clique-R hypothesis holds, then, for $L = \lceil \log_2 \mathsf{n_p} \rceil$, $\bar{\mathsf{n}}_\mathsf{p} = 2^L$, and $m_\mathsf{split} = \binom{k}{2}^{\bar{\mathsf{n}}_\mathsf{p}-1}$, we have*

$$\Delta((\hat{I}_1^i, \cdots, \hat{I}_{\mathsf{n_p}}^i), \mathcal{D}_0[\sqrt[\bar{\mathsf{n}}_\mathsf{p}]{R}]^{\mathsf{n_p}}) \leq (2\bar{\mathsf{n}}_\mathsf{p} - 1) \cdot \binom{k}{2} \cdot 4^{\binom{k}{2}} \cdot 3\lambda^k / \sqrt[\bar{\mathsf{n}}_\mathsf{p}]{R}$$

for all $i \in [m_\mathsf{split}]$ where $(\hat{I}_1^j, \cdots, \hat{I}_{\mathsf{n_p}}^j)_{j\in[m_\mathsf{split}]} \xleftarrow{\$} \mathsf{n_p}\text{-}\mathsf{Split}_\lambda(I)$ for $I \xleftarrow{\$} \mathcal{D}_0[R]$, and

$$\Delta((\hat{I}_1^i, \cdots, \hat{I}_{\mathsf{n_p}}^i), \mathcal{D}_1[\sqrt[\bar{\mathsf{n}}_\mathsf{p}]{R}]^{\mathsf{n_p}}) \leq (2\bar{\mathsf{n}}_\mathsf{p} - 1) \cdot \binom{k}{2} \cdot 4^{\binom{k}{2}} \cdot 3\lambda^k / \sqrt[\bar{\mathsf{n}}_\mathsf{p}]{R}$$

for some $i \in [m_\mathsf{split}]$ where $(\hat{I}_1^j, \cdots, \hat{I}_{\mathsf{n_p}}^j)_{j\in[m_\mathsf{split}]} \xleftarrow{\$} \mathsf{n_p}\text{-}\mathsf{Split}_\lambda(I)$ for $I \xleftarrow{\$} \mathcal{D}_1[R]$.

Proof. Let $L = \lceil \log_2 \mathsf{n_p} \rceil$, $\bar{\mathsf{n}}_\mathsf{p} = 2^L$, and $c = \binom{k}{2}$. The total number of calls to Split_λ is $\sum_{r=0}^{L-1} 2^r c^{2^r-1}$, which is a constant depending only on $\mathsf{n_p}$ and k. Hence $\mathsf{n_p}$-Split_λ has the same asymptotic running time as Split_λ, namely $O(\lambda^{k(1-\delta)})$.

Consider first the case $I \xleftarrow{\$} \mathcal{D}_0[R]$. After r levels of the binary splitting recursion, each output list contains 2^r instances. By repeatedly applying Theorem 4.6 and taking a union bound over the 2^r split calls used to pass from level r to level $r+1$, for every output list after L levels we have

$$\Delta\left((\hat{I}_1,\ldots,\hat{I}_{\bar{\mathsf{n}}_\mathsf{p}}),\mathcal{D}_0[\sqrt[\bar{\mathsf{n}}_\mathsf{p}]{R}]^{\bar{\mathsf{n}}_\mathsf{p}}\right) \le \sum_{r=0}^{L-1} 2^r \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[2^{r+1}]{R}.$$

Since $2^{r+1} \le \bar{\mathsf{n}}_\mathsf{p}$ for every $r \le L-1$, we have $1/\sqrt[2^{r+1}]{R} \le 1/\sqrt[\bar{\mathsf{n}}_\mathsf{p}]{R}$. Therefore,

$$\Delta\left((\hat{I}_1,\ldots,\hat{I}_{\bar{\mathsf{n}}_\mathsf{p}}),\mathcal{D}_0[\sqrt[\bar{\mathsf{n}}_\mathsf{p}]{R}]^{\bar{\mathsf{n}}_\mathsf{p}}\right) \le (2\bar{\mathsf{n}}_\mathsf{p}-1)\binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[\bar{\mathsf{n}}_\mathsf{p}]{R}.$$

Let π be the deterministic projection defined by $\pi(I_1,\ldots,I_{\bar{\mathsf{n}}_\mathsf{p}}) = (I_1,\ldots,I_{\mathsf{n}_\mathsf{p}})$. Since applying the same deterministic function to two random variables cannot increase their total variation distance, applying π to the two $\bar{\mathsf{n}}_\mathsf{p}$-tuples preserves the above upper bound. This proves the claimed bound for $I \leftarrow \mathcal{D}_0[R]$.

We next consider $I \leftarrow \mathcal{D}_1[R]$. In the hybrid argument, when the input to the h-th call at level $r \in \{0,\ldots,L-1\}$ is distributed as $\mathcal{D}_1[\sqrt[2^r]{R}]$, Theorem 4.6 gives an index $j_{r,h} \in [c]$ such that the $j_{r,h}$-th returned pair is close to $\mathcal{D}_1[\sqrt[2^{r+1}]{R}] \times \mathcal{D}_1[\sqrt[2^{r+1}]{R}]$. Since $\mathsf{n}_\mathsf{p}\text{-}\mathsf{Split}_\lambda$ enumerates every $(j_{r,1},\ldots,j_{r,2^r}) \in [c]^{2^r}$ at every level r, one output list uses all these indices. For this list,

$$\begin{aligned}\Delta\left((\hat{I}_1,\ldots,\hat{I}_{\bar{\mathsf{n}}_\mathsf{p}}),\mathcal{D}_1[\sqrt[\bar{\mathsf{n}}_\mathsf{p}]{R}]^{\bar{\mathsf{n}}_\mathsf{p}}\right) &\le \sum_{r=0}^{L-1} 2^r \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[2^{r+1}]{R} \\ &\le (2\bar{\mathsf{n}}_\mathsf{p}-1)\binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[\bar{\mathsf{n}}_\mathsf{p}]{R}.\end{aligned}$$

Applying π gives the claimed bound for $I \leftarrow \mathcal{D}_1[R]$. □

Our construction. We now present our construction of fine-grained multi-party NIKE TNIKE based on the Zero-k-Clique hypothesis in Figure 10.

When an index $s \in [\ell]$ appears in an XOR, we identify it with its $\log \ell$-bit binary representation.

$\underline{\mathsf{Gen}_\lambda(\bot)}$:

Sample a random subset $\mathcal{S} \subset [\ell]$ such that $|\mathcal{S}| = q$

For all $j \in [\ell]$

  $I^j = \mathsf{Generate}_\lambda(R,k,1)$ if $j \in \mathcal{S}$ and $I^j = \mathsf{Generate}_\lambda(R,k,0)$ otherwise

Return $(\mathsf{pk} = (I^j)_{j\in[\ell]}, \mathsf{sk} = \mathcal{S})$

$\underline{\mathsf{Share}_\lambda(i, \mathsf{sk}_i, (\mathsf{pk}_j)_{j\in[\mathsf{n}_\mathsf{p}]})}$:

Parse $(\mathsf{pk}_t = (I_t^j)_{j\in[\ell]})_{t\in[\mathsf{n}_\mathsf{p}]}$, $\mathsf{sk}_i = \mathcal{S}_i$.

For every $t \in [\mathsf{n}_\mathsf{p}]\backslash\{i\}$

  Set $\mathcal{T} = \emptyset$

  For every $s \in \mathcal{S}_i$

    Check by brute forcing whether I_t^s has a solution

    If the check passes, set $\mathcal{T} = \mathcal{T} \cup \{s\}$

  $\mathcal{S}_i = \mathcal{S}_i \cap \mathcal{T}$

If $\mathcal{S}_i = \emptyset$, return $\bot$

Return $\mathsf{key} = \bigoplus_{j=1}^{|\mathcal{S}_i|} s_j$ for all $s_j \in \mathcal{S}_i$

Figure 10: Construction of $\mathsf{TNIKE} = \{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\}$.

Let $\mathcal{C}_1$ (respectively, $\mathcal{C}_2$) be the set of all word-RAM families such that for each $\mathcal{F} = \{f_\lambda\} \in \mathcal{C}_1$ and each $\lambda \in \mathbb{N}$, f_λ runs in time $\tilde{O}(\ell\lambda^2 + q\lambda^k)$ (respectively, $\tilde{O}((\ell\lambda^k)^{1-\delta})$ for some constant $\delta > 0$, where

$\ell = \lambda^{\Omega(1)}$, $\ell = \mathsf{poly}(\lambda)$, and $q < \ell$. Let $\bar{\mathsf{n}}_\mathsf{p} = 2^{\lceil \log_2 \mathsf{n}_\mathsf{p} \rceil}$, $\epsilon_1 = (1-(q/\ell)^{\mathsf{n}_\mathsf{p}-1})^q + 2\ell\mathsf{n}_\mathsf{p}\lambda^k/R$ and

$$\epsilon_2 = 201/200 + \ell(2\bar{\mathsf{n}}_\mathsf{p} - 1)\binom{k}{2}4^{\binom{k}{2}}3\lambda^k/R - (1-(q-1)/\ell)^{\mathsf{n}_\mathsf{p}} + \frac{1}{2}\sqrt{\frac{\ell^{\mathsf{n}_\mathsf{p}-1}}{q^{\mathsf{n}_\mathsf{p}}}} + 2\mathsf{n}_\mathsf{p}(q-1+\ell)\lambda^k/R.$$

We have the following theorem.

Theorem 4.8 *If the strong Zero-k-Clique-R hypothesis holds, then for any constant* n_p, TNIKE *is a* $\mathcal{C}_1$-n_p*-NIKE with* ϵ_1*-correctness and* $\mathcal{C}_2$-ϵ_2*-weak security.*

Proof. **Complexity.** Recall that $\mathsf{Generate}_\lambda$ runs in time $O(\lambda^2)$, and the check by brute forcing takes time $O(\lambda^k)$. Since all parties run $\mathsf{Generate}_\lambda$ for ℓ times and run the check q times, its total running time is $\tilde{O}(\ell\lambda^2 + q\lambda^k)$.

Correctness. We first prove that the probability that the protocol aborts is at most $(1-(q/\ell)^{\mathsf{n}_\mathsf{p}-1})^q$. Without loss of generality, we assume that the indices in $\mathcal{S}_{\mathsf{n}_\mathsf{p}}$ are sampled from $[\ell]$ without replacement. Define the indicator Y_x such that $Y_x = \begin{cases} 1 & x \in \cap_{i=1}^{\mathsf{n}_\mathsf{p}-1}\mathcal{S}_i \\ 0 & \text{otherwise} \end{cases}$. For any $x \xleftarrow{\$} [\ell]$ we have

$$\Pr[Y_x = 1] = \Pr[x \in \mathcal{S}_1 \cap \cdots \cap \mathcal{S}_{\mathsf{n}_\mathsf{p}-1}] = \Pr[x \in \mathcal{S}_1] \cdots \Pr[x \in \mathcal{S}_{\mathsf{n}_\mathsf{p}-1}] = (q/\ell)^{\mathsf{n}_\mathsf{p}-1}$$

i.e., $\Pr[Y_x = 0] = 1-(q/\ell)^{\mathsf{n}_\mathsf{p}-1}$. Fix $\mathcal{S}_{\mathsf{n}_\mathsf{p}} = \{x_1, \ldots, x_q\}$. Sampling without replacement gives $\Pr[Y_{x_j} = 0 \mid Y_{x_1} = \cdots = Y_{x_{j-1}} = 0] \le \Pr[Y_{x_j} = 0]$. Multiplying these conditional probabilities gives the bound below.

$$\Pr[\wedge_{i=1}^q Y_{x_i} = 0] \le \prod_{j=1}^q \Pr[Y_{x_j} = 0] = (1-(q/\ell)^{\mathsf{n}_\mathsf{p}-1})^q.$$

The correctness error occurs only if $\cap_{i\in[\mathsf{n}_\mathsf{p}]}\mathcal{S}_i = \emptyset$, or if some instance generated by $\mathsf{Generate}_\lambda(R,k,0)$ has a solution. Therefore the correctness error probability is at most

$$(1-(q/\ell)^{\mathsf{n}_\mathsf{p}-1})^q + \Delta((I_i^j)_{j\in[\ell],i\in[\mathsf{n}_\mathsf{p}]}, (\mathcal{D}_{b_i^j}[R])_{j\in[\ell],i\in[\mathsf{n}_\mathsf{p}]}),$$

where $I_i^j \xleftarrow{\$} \mathsf{Generate}_\lambda(R,k,b_i^j)$, $b_i^j = 1$ if $j \in \mathcal{S}_i$, and $b_i^j = 0$ otherwise. By Corollary 4.5, this is at most $(1-(q/\ell)^{\mathsf{n}_\mathsf{p}-1})^q + 2\ell\mathsf{n}_\mathsf{p}\lambda^k/R$. Thus $\epsilon_1 = (1-(q/\ell)^{\mathsf{n}_\mathsf{p}-1})^q + 2\ell\mathsf{n}_\mathsf{p}\lambda^k/R$.

$\mathcal{C}_2$-ϵ-weak security. Let λ be the security parameter, and $\mathcal{A} = \{\mathsf{adv}_\lambda\} \in \mathcal{C}_2$ be any adversary against the $\mathcal{C}_2$-ϵ-weak security of TNIKE. The proof proceeds via a sequence of hybrid games as in Figure 11.

G_0**.** This is the original security game where adv_λ receives $(\mathsf{pk}_i)_{i\in[\mathsf{n}_\mathsf{p}]}$ where $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\bot)$ for all $i \in [\mathsf{n}_\mathsf{p}]$, and outputs a session key key.

G_1**.** G_1 and G_0 only differ in the way that $(I_i^j)_{i\in[\mathsf{n}_\mathsf{p}],j\in[\ell]}$ are generated, namely, every instance I_i^j in pk_i is generated as $I_i^j \xleftarrow{\$} \mathcal{D}_0[R]$ (or $I_i^j \xleftarrow{\$} \mathcal{D}_1[R]$) rather than $I_i^j \xleftarrow{\$} \mathsf{Generate}_\lambda(R,k,0)$ (or $I_i^j \xleftarrow{\$} \mathsf{Generate}_\lambda(R,k,1)$) for all $i \in [\mathsf{n}_\mathsf{p}]$ and $j \in [\ell]$.

Lemma 4.9 $|\Pr[\mathsf{G}_1^{\mathsf{adv}_\lambda} \Rightarrow 1] - \Pr[\mathsf{G}_0^{\mathsf{adv}_\lambda} \Rightarrow 1]| \le 2\mathsf{n}_\mathsf{p}\ell\lambda^k/R$.

This lemma follows immediately from Corollary 4.5.

G_2**.** G_2 and G_1 differ in the way that $(\mathsf{sk}_i)_{i\in[\mathsf{n}_\mathsf{p}]}$ are generated, namely, for each $i \in [\mathsf{n}_\mathsf{p}]$, $\mathsf{sk}_i = \{i^*\} \cup \mathcal{S}_i$ where $i^* \xleftarrow{\$} [\ell]$, $\mathcal{S}_i \subset [\ell]$, and $|\mathcal{S}_i| = q-1$. The game aborts if $|\mathsf{sk}_i| < q$ rather than $\mathsf{sk}_i = \mathcal{S}_i$ where $\mathcal{S}_i \subset [\ell]$ and $|\mathcal{S}_i| = q$.

Lemma 4.10 $|\Pr[\mathsf{G}_2^{\mathsf{adv}_\lambda} \Rightarrow 1] - \Pr[\mathsf{G}_1^{\mathsf{adv}_\lambda} \Rightarrow 1]| \le 1 - (1-(q-1)/\ell)^{\mathsf{n}_\mathsf{p}} + \frac{1}{2}\sqrt{\frac{\ell^{\mathsf{n}_\mathsf{p}-1}}{q^{\mathsf{n}_\mathsf{p}}}}$.

Proof. We define Bad as the event that G_2 aborts. Let s_i^j be the jth variable in $\mathcal{S}_i$, for each $i \in [\mathsf{n}_\mathsf{p}]$ and $j \in [q-1]$, we have

$$\Pr[\mathsf{Bad}] = 1 - \prod_{i=1}^{\mathsf{n}_\mathsf{p}} \Pr[s_i^1 \neq i^* \wedge \cdots \wedge s_i^{q-1} \neq i^*] \le 1 - (1-(q-1)/\ell)^{\mathsf{n}_\mathsf{p}}.$$

Games G_0, $\boxed{\mathsf{G}_1}$, $\boxed{\mathsf{G}_2}$, G_3, G_4

Sample random $\{\mathcal{S}_i\}_{i\in[\mathsf{n_p}]} \subset [\ell]$, each $|\mathcal{S}_i| = q$

Sample random $\{\mathcal{S}_i\}_{i\in[\mathsf{n_p}]} \subset [\ell]$, each $|\mathcal{S}_i| = q-1$, and $i^* \xleftarrow{\$} [\ell]$

Let $(\mathsf{sk}_1, \cdots, \mathsf{sk}_{\mathsf{n_p}}) = (\mathcal{S}_1, \cdots, \mathcal{S}_{\mathsf{n_p}})$

For each $i \in [\mathsf{n_p}]$, let $\mathsf{sk}_i = \mathcal{S}_i \cup \{i^*\}$; if $|\mathsf{sk}_i| < q$, abort

For each $i \in [\mathsf{n_p}]$

For $s \in [\ell]$, If $s \in \mathsf{sk}_i$, $I_i^s = \mathsf{Generate}_\lambda(R, k, 1)$, else $I_i^s = \mathsf{Generate}_\lambda(R, k, 0)$

$\boxed{\text{For } s \in [\ell], \text{ if } s \in \mathsf{sk}_i,\ I_i^s \xleftarrow{\$} \mathcal{D}_1[R], \text{ else } I_i^s \xleftarrow{\$} \mathcal{D}_0[R]}$

For each $s \in [\ell]$, if $s = i^*$, $\hat{I}^s \xleftarrow{\$} \mathcal{D}_1[R^{\bar{\mathsf{n}}_\mathsf{p}}]$, else $\hat{I}^s \xleftarrow{\$} \mathcal{D}_0[R^{\bar{\mathsf{n}}_\mathsf{p}}]$
For each $s \in [\ell]$, $(I_1^s, \ldots, I_{\mathsf{n_p}}^s) = \mathsf{n_p}\text{-}\mathsf{Split}_\lambda(\hat{I}^s)$

For each $i \in [\mathsf{n_p}]$ and $s \in [\ell]$, If $s \in \mathcal{S}_i$, $I_i^s = \mathsf{Generate}_\lambda(R, k, 1)$

Let $(\mathsf{pk}_i = (I_i^j)_{j\in[\ell]})_{i\in[\mathsf{n_p}]}$ and $\mathcal{T} = \bigcap_{i=1}^{\mathsf{n_p}} \mathsf{sk}_i$

Return $(\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}$ to adv_λ and get key from adv_λ

Else if $\mathsf{key} = \bigoplus_{i=1}^{|\mathcal{T}|} t_i$ for all $t_i \in \mathcal{T}$ then output 1 and 0 otherwise

Figure 11: The challengers in G_0, G_1, G_2, G_3, and G_4.

Let G_2' denote G_2 conditioned on $\neg\mathsf{Bad}$, we have $|\Pr[\mathsf{G}_2'^{\mathsf{adv}_\lambda} \Rightarrow 1] - \Pr[\mathsf{G}_2^{\mathsf{adv}_\lambda} \Rightarrow 1]| \le \Pr[\mathsf{Bad}]$.

Let $\hat{\mathsf{sk}}_i$ be any subset in $[\ell]$ with size q and $\hat{S} = \bigcap_{i\in[\mathsf{n_p}]} \hat{\mathsf{sk}}_i$. In G_1, for every $\hat{\mathsf{sk}}_1, \cdots, \hat{\mathsf{sk}}_{\mathsf{n_p}}$, we have $\Pr[(\mathsf{sk}_i)_{i\in[\mathsf{n_p}]} = (\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}] = \binom{\ell}{q}^{-\mathsf{n_p}}$, where $\mathsf{sk}_1, \cdots, \mathsf{sk}_{\mathsf{n_p}}$ are independently sampled from the subset of $[\ell]$ with size q. In G_2', for every $\hat{\mathsf{sk}}_i$ in the subset of $[\ell]$ with size q, we have

$$\Pr[(\mathsf{sk}_i')_{i\in[\mathsf{n_p}]} = (\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}] = \sum_{s\in\hat{S}} \frac{1}{\ell} \cdot \binom{\ell-1}{q-1}^{-\mathsf{n_p}} = \frac{|\hat{S}|}{\ell}\binom{\ell-1}{q-1}^{-\mathsf{n_p}},$$

where $\mathsf{sk}_i' = \mathcal{S}_i' \cup \{s'\}$ for every $i \in [\mathsf{n_p}]$, $s' \xleftarrow{\$} [\ell]$, and $\mathcal{S}_1', \cdots, \mathcal{S}_{\mathsf{n_p}}'$ are sampled independently and uniformly among all subsets of $[\ell]\backslash\{s'\}$ with size $q-1$. Thus we have $\Pr[(\mathsf{sk}_i')_{i\in[\mathsf{n_p}]} = (\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}] = \frac{|\hat{S}|}{\mu}\Pr[(\mathsf{sk}_i)_{i\in[\mathsf{n_p}]} = (\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}]$, where $\mu = \frac{q^{\mathsf{n_p}}}{\ell^{\mathsf{n_p}-1}}$. Let $\mathcal{D}_1$ be the distribution of $(\mathsf{sk}_1, \cdots, \mathsf{sk}_{\mathsf{n_p}})$ and $\mathcal{D}_2$ be the distribution of $(\mathsf{sk}_1', \cdots, \mathsf{sk}_{\mathsf{n_p}}')$, we have

$$\begin{aligned}
\Delta(\mathcal{D}_1, \mathcal{D}_2) =& \frac{1}{2} \sum_{\hat{\mathsf{sk}}_1, \cdots, \hat{\mathsf{sk}}_{\mathsf{n_p}}} |\Pr[(\mathsf{sk}_i)_{i\in[\mathsf{n_p}]} = (\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}] - \Pr[(\mathsf{sk}_i')_{i\in[\mathsf{n_p}]} = (\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}]| \\
=& \frac{1}{2\mu} \sum_{\hat{\mathsf{sk}}_1, \cdots, \hat{\mathsf{sk}}_{\mathsf{n_p}}} \Pr[(\mathsf{sk}_i)_{i\in[\mathsf{n_p}]} = (\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}] \cdot ||\hat{S}| - \mu| \\
\le& \frac{1}{2\mu}\Big(\sum_{\hat{\mathsf{sk}}_1, \cdots, \hat{\mathsf{sk}}_{\mathsf{n_p}}} \Pr[(\mathsf{sk}_i)_{i\in[\mathsf{n_p}]} = (\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}] \cdot (||\hat{S}| - \mu|)^2\Big)^{1/2}.
\end{aligned}$$

Recall that $\hat{S} = \bigcap_{i\in[\mathsf{n_p}]} \hat{\mathsf{sk}}_i$ for any $\hat{\mathsf{sk}}_i$. Since $\mathsf{sk}_1, \cdots, \mathsf{sk}_{\mathsf{n_p}}$ are sampled independently, for any $(\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}$, we have

$$\sum_{\hat{\mathsf{sk}}_1, \cdots, \hat{\mathsf{sk}}_{\mathsf{n_p}}} \Pr[(\mathsf{sk}_i)_{i\in[\mathsf{n_p}]} = (\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}]|\hat{S}| = \sum_{s\in[\ell]} \Pr[s \in \bigcap_{i\in[\mathsf{n_p}]} \mathsf{sk}_i] = \ell(\frac{q}{\ell})^{\mathsf{n_p}} = \mu.$$

Moreover, for any fixed distinct $s,t\in[\ell]$, we have $\Pr[s,t\in\bigcap_{i\in[\mathsf{n_p}]}\mathsf{sk}_i]=\left(\frac{q(q-1)}{\ell(\ell-1)}\right)^{\mathsf{n_p}}\le\mu^2$. By the fact that $|\hat{S}|^2=|\hat{S}|+|\{(s,t)\in\hat{S}^2:s\neq t\}|$, we have

$$\sum_{\hat{\mathsf{sk}}_1,\cdots,\hat{\mathsf{sk}}_{\mathsf{n_p}}}\Pr[(\mathsf{sk}_i)_{i\in[\mathsf{n_p}]}=(\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}]|\hat{S}^2|=\sum_{s\in[\ell]}\Pr[s\in\bigcap_{i\in[\mathsf{n_p}]}\mathsf{sk}_i]+\sum_{s,t\in[\ell]\wedge s\neq t}\Pr[s,t\in\bigcap_{i\in[\mathsf{n_p}]}\mathsf{sk}_i]$$
$$=\ell(\frac{q}{\ell})^{\mathsf{n_p}}+\ell(\ell-1)(\frac{q(q-1)}{\ell(\ell-1)})^{\mathsf{n_p}}\le\mu+\mu^2.$$

Thus we have

$$\Delta(\mathcal{D}_1,\mathcal{D}_2)\le\frac{1}{2\mu}(\sum_{\hat{\mathsf{sk}}_1,\cdots,\hat{\mathsf{sk}}_{\mathsf{n_p}}}\Pr[(\mathsf{sk}_i)_{i\in[\mathsf{n_p}]}=(\hat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}]\cdot(||\hat{S}|-\mu|)^2)^{1/2}$$
$$\le\frac{1}{2\mu}((\mu+\mu^2)-2\mu^2+\mu^2)^{1/2}=\frac{1}{2\mu}\sqrt{\mu}=\frac{1}{2}\sqrt{\frac{\ell^{\mathsf{n_p}-1}}{q^{\mathsf{n_p}}}}.$$

Note that for every fixed tuple $(\widehat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}$, the conditional view of adv_λ in G_1 given $(\mathsf{sk}_i)_{i\in[\mathsf{n_p}]}=(\widehat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}$ is identical to its conditional view in G_2' given $(\mathsf{sk}'_i)_{i\in[\mathsf{n_p}]}=(\widehat{\mathsf{sk}}_i)_{i\in[\mathsf{n_p}]}$. Thus we have $|\Pr[\mathsf{G}_2'^{\mathsf{adv}_\lambda}\Rightarrow 1]-\Pr[\mathsf{G}_1^{\mathsf{adv}_\lambda}\Rightarrow 1]|\le\Delta(\mathcal{D}_1,\mathcal{D}_2)$.

By the triangle inequality, we have

$$|\Pr[\mathsf{G}_2^{\mathsf{adv}_\lambda}\Rightarrow 1]-\Pr[\mathsf{G}_1^{\mathsf{adv}_\lambda}\Rightarrow 1]|\le\Pr[\mathsf{Bad}]+\Delta(\mathcal{D}_1,\mathcal{D}_2)\le 1-(1-(q-1)/\ell)^{\mathsf{n_p}}+\frac{1}{2}\sqrt{\frac{\ell^{\mathsf{n_p}-1}}{q^{\mathsf{n_p}}}},$$

completing this part of the proof. □

G_3. G_3 and G_2 only differ in the way that $\{I_i^s\}_{i\in[\mathsf{n_p}],s\in\mathcal{S}_i}$ are generated, namely, for each $i\in[\mathsf{n_p}]$ and each $s\in\mathcal{S}_i$, $I_i^s\xleftarrow{\$}\mathsf{Generate}_\lambda(R,k,1)$ rather than $I_i^s\xleftarrow{\$}\mathcal{D}_1[R]$.

Lemma 4.11 $|\Pr[\mathsf{G}_3^{\mathsf{adv}_\lambda}\Rightarrow 1]-\Pr[\mathsf{G}_2^{\mathsf{adv}_\lambda}\Rightarrow 1]|\le 2\mathsf{n_p}(q-1)\lambda^k/R$.

This lemma follows immediately from Corollary 4.5.

G_4. In G_4, choose $\gamma^*\in[m_{\mathsf{split}}]$ guaranteed by Theorem 4.7 for $\hat{I}^{i^*}\xleftarrow{\$}\mathcal{D}_1[R^{\bar{\mathsf{n}}_\mathsf{p}}]$. For every $s\in[\ell]$, the selected output list is the γ^*-th output list of $\mathsf{n_p}\text{-}\mathsf{Split}_\lambda(\hat{I}^s)$. For $s\neq i^*$, the input is from $\mathcal{D}_0[R^{\bar{\mathsf{n}}_\mathsf{p}}]$, for which Theorem 4.7 holds for every output list.

Lemma 4.12
$$|\Pr[\mathsf{G}_4^{\mathsf{adv}_\lambda}\Rightarrow 1]-\Pr[\mathsf{G}_3^{\mathsf{adv}_\lambda}\Rightarrow 1]|\le\ell(2\bar{\mathsf{n}}_\mathsf{p}-1)\binom{k}{2}4^{\binom{k}{2}}3\lambda^k/R.$$

By Theorem 4.7 and a union bound, Lemma 4.12 immediately follows.

Lemma 4.13 $\Pr[\mathsf{G}_4^{\mathsf{adv}_\lambda}\Rightarrow 1]\le 1/200$.

Proof. We construct an adversary $\mathcal{B}=\{\mathsf{advb}_\lambda\}$ for Theorem 4.3. On input $(\hat{I}^s)_{s\in[\ell]}$, where one hidden index is sampled from $\mathcal{D}_1[R^{\bar{\mathsf{n}}_\mathsf{p}}]$ and the rest from $\mathcal{D}_0[R^{\bar{\mathsf{n}}_\mathsf{p}}]$, advb_λ samples the sets $\mathcal{S}_i$ as in G_4 and computes all output lists of $\mathsf{n_p}\text{-}\mathsf{Split}_\lambda(\hat{I}^s)$ for every s. For each $\gamma\in[m_{\mathsf{split}}]$, it forms public keys by taking the γ-th output list at every s and then resetting $I_i^s=\mathsf{Generate}_\lambda(R,k,1)$ for every $i\in[\mathsf{n_p}]$ and $s\in\mathcal{S}_i$. It runs adv_λ on these public keys, obtains key_γ, sets $\eta_\gamma=\mathsf{key}_\gamma\oplus\bigoplus_{t\in\bigcap_{i=1}^{\mathsf{n_p}}\mathcal{S}_i}t$, and checks by brute force whether $\eta_\gamma\in[\ell]$ and $\hat{I}^{\eta_\gamma}$ has a solution. If some check accepts, advb_λ outputs the corresponding η_γ; otherwise it outputs an arbitrary index.

Note that computing all output lists of $\mathsf{n_p}\text{-}\mathsf{Split}_\lambda(\hat{I}^s)$ for all $s\in[\ell]$ takes time $\tilde{O}(\ell\lambda^{k(1-\delta_0)})$ for some constant $\delta_0>0$, and the reduction runs adv_λ once for each $\gamma\in[m_{\mathsf{split}}]$. For each $\gamma\in[m_{\mathsf{split}}]$, it evaluates $\mathsf{Generate}_\lambda$ exactly $\sum_{i\in[\mathsf{n_p}]}|S_i|=\mathsf{n_p}(q-1)$ times. It also checks the m_{split} candidate indices by brute force. Since m_{split} and $\mathsf{n_p}$ are constants, $\ell=\lambda^{\Omega(1)}$, $\ell=\mathsf{poly}(\lambda)$, and $q<\ell$, the total running time is $\tilde{O}((\ell\lambda^k)^{1-\delta'})$ for some constant $\delta'>0$.

For $\gamma = \gamma^*$, conditioned on G_4 not aborting, the view given to adv_λ is identical to its view in G_4. If G_4 outputs 1, then no abort occurred and adv_λ returned the key $i^* \oplus \bigoplus_{t \in \bigcap_{i=1}^{\mathsf{n_p}} S_i} t$, so $\eta_{\gamma^*} = i^*$ and the final check accepts. Thus advb_λ succeeds with probability at least $\Pr[\mathsf{G}_4^{\mathsf{adv}_\lambda} \Rightarrow 1]$. By Theorem 4.3, its success probability is at most $1/200$. □

Putting all the above together, we have

$$\Pr[\mathsf{G}_0^{\mathsf{adv}_\lambda} \Rightarrow 1] \leq 201/200 + \epsilon_{\mathsf{split}} - (1-(q-1)/\ell)^{\mathsf{n_p}} + \frac{1}{2}\sqrt{\frac{\ell^{\mathsf{n_p}-1}}{q^{\mathsf{n_p}}}} + 2\mathsf{n_p}(q-1+\ell)\lambda^k/R,$$

where $\epsilon_{\mathsf{split}} = \ell(2\bar{\mathsf{n}}_\mathsf{p} - 1)\binom{k}{2}4^{\binom{k}{2}}3\lambda^k/R$, completing the proof of Theorem 4.8. □

Example with asymptotic complexity. Let $\ell = \lambda^{\mathsf{n_p}}$, $q = \lambda^{\mathsf{n_p}-1}\log\lambda$, and $R = \lambda^{\mathsf{n_p}k}$. We have

$$\epsilon_1 \leq (1-(q/\ell)^{\mathsf{n_p}-1})^q + 2\ell\mathsf{n_p}\lambda^k/R = \left(1-((\lambda^{\mathsf{n_p}-1}\log\lambda)/\lambda^{\mathsf{n_p}})^{\mathsf{n_p}-1}\right)^{\lambda^{\mathsf{n_p}-1}\log\lambda} + 2\mathsf{n_p}\lambda^{\mathsf{n_p}+k}/\lambda^{\mathsf{n_p}k},$$

which approaches

$$e^{-\log^{\mathsf{n_p}}\lambda} + O(1/\lambda^{k\mathsf{n_p}-k-\mathsf{n_p}}) = O(1/\lambda^{k\mathsf{n_p}-k-\mathsf{n_p}})$$

when λ is sufficiently large, and

$$\epsilon_2 = 1/200 + \frac{1}{2(\log\lambda)^{\mathsf{n_p}/2}} + \tilde{O}(\lambda^{-1}) + \tilde{O}(\lambda^{\mathsf{n_p}+k-k\mathsf{n_p}}).$$

In this case, the scheme is computable within time $\tilde{O}(\lambda^{\mathsf{n_p}+k-1})$ and secure against adversaries running in time $\tilde{O}(\lambda^{\mathsf{n_p}+k-\Omega(1)})$. When $\mathsf{n_p}$ is fixed, setting $k=3$ provides better bounds in our instantiation.

Concrete example. We now give a concrete example of the parameters. Let $\lambda = 2^{16}$, $\ell = 2^{48}$, $q = 2^{36}$, $R = 2^{144}$, $\mathsf{n_p} = 3$, and $k = 3$. Since $(1-(q/\ell)^{\mathsf{n_p}-1})^q < 10^{-16}$, $2\ell\mathsf{n_p}\lambda^k/R = 6/2^{48} < 2.14 \times 10^{-14}$, and $201/200 - (1-(q-1)/\ell)^{\mathsf{n_p}} < 1/200 + 1/1000 = 3/500$, we have $\epsilon_1 < 2.15 \times 10^{-14}$ and $\epsilon_2 \leq 3/500 + 1/128 + 1/500 < 1/63$.

4.3 Construction with Full Security

We now transform the weakly secure multi-party NIKE from Section 4.2 to a fully secure one. We first use the parallel repetition theorem of Bellare et al. [8] to amplify its weak security, reducing the adversary's recovery probability to an inverse-polynomial bound. After this amplification, we apply the Goldreich–Levin transformation to obtain full security.

Parallel repetition. Let V be the verifier of a three-message protocol. Namely, on common input x and random tape R, the verifier first receives the prover's first message M, sends a challenge C, then receives the prover's answer A, and outputs $V(x, M, C, A; R) \in \{0,1\}$. The first prover message M may be empty. For a prover P, let $\mathsf{Acc}(P, V, x)$ be the probability that V outputs 1 in the interaction with P on input x.

For $k \geq 1$, V^k denotes the k-fold parallel repetition of V. It uses independent random tapes $R_1, \ldots, R_k$. After receiving prover messages $M_1, \ldots, M_k$, it sends the challenges $C_1, \ldots, C_k$, where C_j is the challenge produced by V on (x, M_j) using random tape R_j. After receiving prover answers $A_1, \ldots, A_k$, it outputs 1 iff $V(x, M_j, C_j, A_j; R_j) = 1$ for all $j \in [k]$. Note that the prover interacting with V^k is not required to answer the k copies independently. Let $\mathsf{Comm}(A, B, x)$ denote the communication complexity of the interaction between A and B on input x. For an oracle algorithm S, write $S^{P^*,V}$ for S with oracle access to P^* and V.

Theorem 4.14 (Theorem 4.1 in [8]). *Let $0 < \epsilon, \delta < 1$, and let $k \geq 2$. Suppose $\epsilon > \frac{16}{\delta}e^{-\frac{\delta^2 k}{128}}$, then there exists an oracle algorithm S such that, for every prover P^*, verifier V, and input x, the following is true: If $\mathsf{Acc}(P^*, V^k, x) \geq 2\epsilon$, then $\mathsf{Acc}(S^{P^*,V}, V, x) \geq 1-\delta$. Moreover, $S^{P^*,V}$ runs in time polynomial in k, $|x|$, $1/\epsilon$, and $\mathsf{Comm}(P^*, V^k, x)$.*

In the following, $T(\lambda)$-$\epsilon(\lambda)$-weak security means $\mathcal{C}$-$\epsilon(\lambda)$-weak security for the class $\mathcal{C}$ of all word-RAM families running in time $\widetilde{O}(T(\lambda))$.

Let $t = \lceil d \log \lambda \rceil$, where $d > 0$ is a constant. Let T_{u}, T_1, T_2 be functions of λ. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the sets of all word-RAM families running in time $\widetilde{O}(T_{\mathsf{u}}(\lambda))$ and $\widetilde{O}(T_2(\lambda))$, respectively. Let $\Pi = \{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\}$ be a $\mathcal{C}_1$-$\mathsf{n_p}$-party NIKE with $\epsilon_{\mathsf{cor}}(\lambda)$-correctness and $T_1(\lambda)$-$\alpha(\lambda)$-weak security, where $\alpha(\lambda) \leq \alpha_0$ for all sufficiently large λ, for some constant $\alpha_0 < 1$. Let $\epsilon_0(\lambda) = \frac{32}{1-\alpha_0} e^{(-\frac{t(1-\alpha_0)^2}{512})}$ and let p be some polynomial such that $T_1(\lambda) \geq p(t, \lambda, 1/\epsilon_0(\lambda), T_2(\lambda), T_{\mathsf{u}}(\lambda))$.

Let $\Pi^{\|t} = \{\mathsf{Gen}_\lambda^{\|t}, \mathsf{Share}_\lambda^{\|t}\}$ be the scheme that runs t independent copies of Π and outputs the concatenation of the t shared keys.

Theorem 4.15 *$\Pi^{\|t}$ is a $\mathcal{C}_1$-$\mathsf{n_p}$-party NIKE with correctness error at most $t\epsilon_{\mathsf{cor}}(\lambda)$, and satisfies $\mathcal{C}_2$-λ^{-c}-weak security for some constant $c > 0$.*

Proof. **Complexity.** Note that $\Pi^{\|t}$ runs $t = O(\log \lambda)$ independent copies of Π. Since $\mathcal{C}_1$ consists of word-RAM families running in time $\widetilde{O}(T_{\mathsf{u}}(\lambda))$, the honest running time of $\Pi^{\|t}$ is still in $\mathcal{C}_1$.

Correctness. Let Bad_j be the event that the j-th copy does not give all parties the same shared key. By the correctness of Π, since $\Pr[\mathsf{Bad}_j] \leq \epsilon_{\mathsf{cor}}(\lambda)$ for every $j \in [t]$, we have

$$\Pr\left[\bigvee_{j=1}^{t} \mathsf{Bad}_j\right] \leq \sum_{j=1}^{t} \Pr[\mathsf{Bad}_j] \leq t\epsilon_{\mathsf{cor}}(\lambda).$$

Security. Define $V_{\Pi,\lambda}$ as follows. On input 1^λ, it samples urs as in the setup of Π and $\mathsf{urs} = \perp$ in the plain model. It samples $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for all $i \in [\mathsf{n_p}]$ and let $X = (\mathsf{urs}, \mathsf{pk}_1, \cdots, \mathsf{pk}_{\mathsf{n_p}})$ and $Y = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]})$. It then sends X to its prover, receives Y', and accepts iff $Y' = Y$.

Suppose that there exists an adversary $\mathcal{A} = \{\mathsf{adv}_\lambda\} \in \mathcal{C}_2$ such that

$$\Pr[\mathsf{adv}_\lambda(X_1, \cdots, X_t) = (Y_1, \cdots, Y_t)] > 2\epsilon_0(\lambda),$$

where urs_j is sampled as in the setup of Π and $\mathsf{urs}_j = \perp$ in the plain model, $(\mathsf{pk}_{i,j}, \mathsf{sk}_{i,j}) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs}_j)$ for all $i \in [\mathsf{n_p}]$, $X_j = (\mathsf{urs}_j, \mathsf{pk}_{1,j}, \cdots, \mathsf{pk}_{\mathsf{n_p},j})$, $Y_j = \mathsf{Share}_\lambda(1, \mathsf{sk}_{1,j}, (\mathsf{pk}_{i,j})_{i\in[\mathsf{n_p}]})$ for every $j \in [t]$.

Define a prover P^* for $V^t_{\Pi,\lambda}$ as follows. After receiving $(X_1, \ldots, X_t)$, it outputs $\mathsf{adv}_\lambda(X_1, \cdots, X_t)$. Let b_t be the output bit of $V^t_{\Pi,\lambda}$ in its interaction with P^* on input 1^λ. By the construction of $V^t_{\Pi,\lambda}$ and P^*, we have

$$b_t = \begin{cases} 1, & \mathsf{adv}_\lambda(X_1, \ldots, X_t) = (Y_1, \ldots, Y_t), \\ 0, & \text{otherwise.} \end{cases}$$

Let $\rho = \frac{1-\alpha_0}{2}$ and $\epsilon = \frac{1}{2}\Pr[\mathsf{adv}_\lambda(X_1, \cdots, X_t) = (Y_1, \cdots, Y_t)]$, we have $\epsilon > \epsilon_0(\lambda) = \frac{16}{\rho} e^{-\rho^2 t/128}$ and $\Pr[b_t = 1] = 2\epsilon$. By Theorem 4.14, there exists $S^{P^*, V_{\Pi,\lambda}}$ such that

$$\Pr[b_1 = 1] \geq 1 - \rho = \frac{1 + \alpha_0}{2} > \alpha_0.$$

We then construct $\mathcal{B} = \{\mathsf{advb}_\lambda\}$ against the weak security of Π as follows. On input $X = (\mathsf{urs}, \mathsf{pk}_1, \cdots, \mathsf{pk}_{\mathsf{n_p}})$, advb_λ runs $S^{P^*, V_{\Pi,\lambda}}$ in the interaction with $V_{\Pi,\lambda}$, uses X as the verifier message, and outputs the answer Y' returned by $S^{P^*, V_{\Pi,\lambda}}$.

Since $b_1 = 1$ iff $Y' = Y$, where $Y = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]})$ and $Y' = \mathsf{advb}_\lambda(X)$, we have

$$\Pr[\mathsf{advb}_\lambda(X) = Y] = \Pr[b_1 = 1] > \alpha_0 \geq \alpha(\lambda).$$

Note that $\mathcal{B}$ runs one execution of $S^{P^*, V_{\Pi,\lambda}}$. Each oracle call to P^* is answered by running adv_λ once, which takes time $\tilde{O}(T_2(\lambda))$. Each oracle call to $V_{\Pi,\lambda}$ is answered by sampling urs, $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for all $i \in [\mathsf{n_p}]$, computing $Y = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]})$, and checking whether the returned answer equals Y, which takes time $\tilde{O}(T_u(\lambda))$. Hence the running time of $\mathcal{B}$ is at most

$$p(t, \lambda, 1/\epsilon_0(\lambda), \tilde{O}(tT_u(\lambda))) \cdot (\tilde{O}(T_2(\lambda)) + \tilde{O}(T_u(\lambda))) \leq \tilde{O}(T_1(\lambda)).$$

This contradicts the $T_1(\lambda)$-$\alpha(\lambda)$-weak security of Π. Since $t = \lceil d \log \lambda \rceil$ and $\alpha_0 < 1$, there exists $c > 0$ such that

$$\Pr[\mathsf{adv}_\lambda(X_1, \ldots, X_t) = (Y_1, \cdots, Y_t)] \leq 2\epsilon_0(\lambda) = \frac{64}{1-\alpha_0} e^{-\frac{t(1-\alpha_0)^2}{512}} \leq \lambda^{-c}.$$

□

Full security. We next convert the weakly secure scheme obtained above into a fully secure one-bit scheme. Before the security analysis, we recall the following Chebyshev bound.

Theorem 4.16 (Chebyshev bound). *For any $\epsilon > 0$, $b \in \{0,1\}$, and $q \in \mathbb{N}$, let $\{x_i\}_{i\in[q]}$ be pairwise-independent, 0/1-random variables such that $\Pr[x_i = b] \geq 1/2+\epsilon$ for all $i \in [q]$. Let $x = 1$ if $\Sigma_{i=1}^{q} x_i \geq q/2$ and $x = 0$ otherwise. Then $\Pr[x \neq b] \leq 1/(4\epsilon^2 q)$.*

Let $\mathcal{C}_1$ (respectively, $\mathcal{C}_2$) be the set of all word-RAM families such that for each $\mathcal{F} = \{f_\lambda\} \in \mathcal{C}_1$ and each $\lambda \in \mathbb{N}$, f_λ runs in time $\tilde{O}(T_1(\lambda))$ (respectively, $\tilde{O}(T_2(\lambda))$) for any constant $\mathsf{n_p} \geq 2, k \geq 3$. Let ϵ_{cor} be a function of λ. For any constant $a > 0$, let $\epsilon_2 = 1/\log^a \lambda$, and let $\epsilon_1 = \epsilon_1(\lambda)$ be a function in λ satisfying $\epsilon_1 \leq \epsilon_2^3/32Q(\lambda)$. Let $\widehat{\mathsf{TNIKE}} = (\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda)$ be a $\mathcal{C}_1$-$\mathsf{n_p}$-NIKE that satisfies $\mathcal{C}_2$-ϵ_1-weak security with the shared key size $Q(\lambda) = \mathsf{poly}(\log(\lambda))$. We construct $\mathsf{TNIKE}^* = (\mathsf{Gen}^*_\lambda, \mathsf{Share}^*_\lambda)$ satisfying $\mathcal{C}_2$-ϵ_2-security as in Figure 12.

Theorem 4.17 *If $\widehat{\mathsf{TNIKE}}$ is a $\mathcal{C}_1$-$\mathsf{n_p}$-NIKE with $\mathcal{C}_2$-ϵ_1-weak security and ϵ_{cor}-correctness, then TNIKE^* is a $\mathcal{C}_1$-$\mathsf{n_p}$-NIKE with $\mathcal{C}_2$-ϵ_2-security and ϵ_{cor}-correctness.*

$\underline{\mathsf{Gen}^*_\lambda(\bot)}$:	$\underline{\mathsf{Share}^*_\lambda(i, \mathsf{sk}^*_i, (\mathsf{pk}^*_j)_{j\in[\mathsf{n_p}]})}$:
$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{Gen}_\lambda(\bot)$	Parse $\mathsf{pk}^*_j = (\mathsf{pk}_j, \mathbf{v}_j)$ for all $j \in [\mathsf{n_p}]$
$\mathbf{v} \xleftarrow{\$} \{0,1\}^{Q(\lambda)}$	$\mathsf{key} = \mathsf{Share}_\lambda(i, \mathsf{sk}^*_i, (\mathsf{pk}_j)_{j\in[\mathsf{n_p}]})$
Return $(\mathsf{pk}^* = (\mathsf{pk}, \mathbf{v}), \mathsf{sk}^* = \mathsf{sk})$	If $\mathsf{key} = \bot$, return $\bot$
	Return $\mathsf{key}^* = \mathsf{key} \cdot \mathbf{v}_{\mathsf{n_p}}$

Figure 12: Definition of $\mathsf{TNIKE}^* = \{\mathsf{Gen}^*_\lambda, \mathsf{Share}^*_\lambda\}$ with $\mathcal{C}_2$-$1/\lambda^{o(1)}$-security.

Proof. Correctness follows immediately from the ϵ_{cor}-correctness of $\widehat{\mathsf{TNIKE}}$. Indeed, whenever all parties in $\widehat{\mathsf{TNIKE}}$ output the same key $\mathsf{key} \neq \bot$, all parties in TNIKE^* apply the same deterministic post-processing to key and hence output the same session key. If $\widehat{\mathsf{TNIKE}}$ outputs $\bot$, then TNIKE^* also outputs $\bot$. Therefore, TNIKE^* has correctness error at most ϵ_{cor}.

Let m be the smallest power of two such that $m \geq 4Q(\lambda)/\epsilon_2^2$, $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\bot)$ for all $i \in [\mathsf{n_p}]$, $\mathsf{key} = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, \mathsf{pk}_{[\mathsf{n_p}]})$, $\mathbf{v}_{\mathsf{n_p}} \xleftarrow{\$} \{0,1\}^{Q(\lambda)}$. Let $D = \{D_\lambda\} \in \mathcal{C}_2$ be any adversary breaking the $\mathcal{C}_2$-ϵ_2-security of TNIKE^*. Since weak security gives $\Pr[\mathsf{key} = \bot] \leq \epsilon_1$, condition on $\mathsf{key} \neq \bot$ for this predictor calculation only. Let z be the public information except the challenge bit, $\beta \in \{0,1\}$ for the real challenge bit, and $u \xleftarrow{\$} \{0,1\}$. If $\Pr[D_\lambda(z,\beta) = 1] - \Pr[D_\lambda(z,u) = 1] < 0$, we replace D_λ by D'_λ, where $D'_\lambda(z,b) = 1 - D_\lambda(z,b)$. Thus we may assume that $\Pr[D_\lambda(z,\beta) = 1] - \Pr[D_\lambda(z,u) = 1] > \epsilon_2 - \epsilon_1$. We now convert this distinguisher into a predictor for the real challenge bit. Define $\mathcal{A} = \{\mathsf{adv}_\lambda\}$ as follows. On input z, it samples $r \xleftarrow{\$} \{0,1\}$, outputs r if $D_\lambda(z,r) = 1$ and outputs $1-r$ otherwise. On each call with input vector $\mathbf{v}$, set $\mathbf{v}_{\mathsf{n_p}} = \mathbf{v}$ and freshly sample $\mathbf{v}_1, \ldots, \mathbf{v}_{\mathsf{n_p}-1}$ independently and uniformly from $\{0,1\}^{Q(\lambda)}$. Since u is uniform over $\{0,1\}$, we have

$$\Pr[\mathsf{adv}_\lambda(z) = \beta] = \frac{1}{2} + \Pr[D_\lambda(z,\beta) = 1] - \Pr[D_\lambda(z,u) = 1] \geq \frac{1}{2} + \epsilon_2 - \epsilon_1.$$

We then construct an adversary $\mathcal{B} = \{b_\lambda\} \in \mathcal{C}_2$ breaking the $\mathcal{C}_2$-ϵ_1-weak security of $\widehat{\mathsf{TNIKE}}$.

Firstly, b_λ chooses $\log(m)$ pairs $(\sigma_1, r_1), \cdots, (\sigma_{\log(m)}, r_{\log(m)}) \xleftarrow{\$} \{0,1\} \times \{0,1\}^{Q(\lambda)}$. Then, for every nonempty subset $\mathcal{X} \subseteq [\log(m)]$, b_λ computes $r_\mathcal{X} = \oplus_{i\in\mathcal{X}} r_i$ and $\sigma_\mathcal{X} = \oplus_{i\in\mathcal{X}} \sigma_i$. For every $i \in [Q(\lambda)]$, b_λ runs

$$\mathsf{key}_i^\mathcal{X} = \sigma_\mathcal{X} \oplus \mathsf{adv}_\lambda(\{\mathsf{pk}_i\}_{i\in[\mathsf{n_p}]}, r_\mathcal{X} \oplus e^i),$$

where e^i denotes the $Q(\lambda)$-bit string with 1 in the i'th position and 0 everywhere else, and sets key_i^* as the majority of $\mathsf{key}_i^\mathcal{X}$. Finally, b_λ outputs $\mathsf{key}^* = \mathsf{key}_1^* || \cdots || \mathsf{key}_{Q(\lambda)}^*$. Here $\mathcal{S}$ depends on the public keys and excludes $\bot$; $\Pr[\mathsf{key} \in \mathcal{S}]$ is taken over the actual joint distribution of the public keys and key. Let

$$\mathcal{S} = \{\mathsf{key} \big| \Pr[\mathsf{adv}_\lambda(\{\mathsf{pk}_i\}_{i\in[\mathsf{n_p}]}, \mathbf{v}_{\mathsf{n_p}}) = \mathsf{key} \cdot \mathbf{v}_{\mathsf{n_p}}] > 1/2 + \epsilon_2/2\}.$$

A quick calculation yields $\Pr[\mathsf{key} \in \mathcal{S}] > \epsilon_2/2$. Since $\{\sigma_i\}_{i\in[\log(m)]}$ and $\{r_i\}_{i\in[\log(m)]}$ are sampled uniformly at random, we have $\Pr[(\sigma_i = r_i \cdot \mathsf{key})_{i\in[\log(m)]}] = 1/m$. Notice that the random variables $\{r_{\mathcal{X}}\}_{\mathcal{X}\subseteq[\log(m)]}$ are pairwise independent. Then the probability that the majority of bits is incorrect is

$$\Pr[(\mathsf{key}_i^* \neq \mathsf{key}_i) \mid (\sigma_j = \mathsf{key} \cdot r_j)_{j\in[\log(m)]} \wedge (\mathsf{key} \in S)] \leq 1/((m-1)\epsilon_2^2) \leq 2/(m\epsilon_2^2),$$

for each $i \in [Q(\lambda)]$ by the Chebyshev bound. Putting all these together we have

$$\begin{aligned}
\Pr[b_\lambda(\{\mathsf{pk}_i\}_{i\in[\mathsf{n_p}]}) = \mathsf{key}] \geq& \Pr[\mathsf{key} \in S] \cdot \Pr[b_\lambda(\{\mathsf{pk}_i\}_{i\in[\mathsf{n_p}]}) = \mathsf{key} \mid \mathsf{key} \in S] \\
\geq& \epsilon_2/(2m) \cdot \Pr[\forall i \in [Q(\lambda)], (\mathsf{key}_i^* = \mathsf{key}_i) \mid \mathsf{key} \in S, (\sigma_j = \mathsf{key} \cdot r_j)_{j\in[\log(m)]}] \\
=& \epsilon_2/(2m)(1 - \Pr[\exists i \in [Q(\lambda)] \text{ s.t. } \mathsf{key}_i^* \neq \mathsf{key}_i \mid \mathsf{key} \in S, (\sigma_j = \mathsf{key} \cdot r_j)_{j\in[\log(m)]}]) \\
\geq& \epsilon_2/(2m)(1 - \sum_{i=1}^{Q(\lambda)} \Pr[\mathsf{key}_i^* \neq \mathsf{key}_i \mid (\sigma_j = \mathsf{key} \cdot r_j)_{j\in[\log(m)]}, \mathsf{key} \in S]) \\
\geq& \epsilon_2/(2m)(1 - 2Q(\lambda)/(m\epsilon_2^2)) = \epsilon_2/(2m) - Q(\lambda)/(m^2\epsilon_2).
\end{aligned}$$

Since $8Q(\lambda)/\epsilon_2^2 > m \geq 4Q(\lambda)/\epsilon_2^2$, the probability that b_λ succeeds is greater than $\epsilon_2^3/(32Q(\lambda))$. Since $\epsilon_1 \leq \epsilon_2^3/(32Q(\lambda))$, this contradicts the $\mathcal{C}_2$-ϵ_1-weak security of $\widehat{\mathsf{TNIKE}}$. Note that adv_λ runs in time $\tilde{O}(T_2(\lambda))$ and $m = Q(\lambda)\lambda^{o(1)}$, $\{b_\lambda\}$ runs in time $\tilde{O}(Q(\lambda) \cdot m \cdot T_2(\lambda)) = \tilde{O}(T_2(\lambda))$, completing the proof of Theorem 4.17. □

Recall that in the example with asymptotic complexity, TNIKE satisfies $\mathcal{C}_2$-ϵ_{sec}-weak security, where

$$\epsilon_{\mathsf{sec}} = 1/200 + \frac{1}{2(\log\lambda)^{\mathsf{n_p}/2}} + \tilde{O}(\lambda^{-1}) + \tilde{O}(\lambda^{\mathsf{n_p}+k-k\mathsf{n_p}})$$

and $\mathcal{C}_2$ is the set of all word-RAM families such that for each $\{f_\lambda\} \in \mathcal{C}_2$, f_λ runs in time $\tilde{O}(\lambda^{\mathsf{n_p}+k-\Omega(1)})$. Let $\Pi = \mathsf{TNIKE}$. By Theorem 4.15, we can obtain $\Pi^{\|t}$ with $\mathcal{C}_2$-λ^{-c}-weak security. Let $\widehat{\mathsf{TNIKE}} = \Pi^{\|t}$. By Theorem 4.17, we can obtain TNIKE^* with $\mathcal{C}_2$-$1/\lambda^{o(1)}$-security.

Extension to IB-NIKE. Similar to our construction in the bounded-parallel-time model given in Section 3.2, we can show that our construction in this section satisfies extendability (see Definition 2.2), and thus can be converted into a BPRF and an IB-NIKE in the same fine-grained setting. We refer the reader to Section 6 for the full details.

5 Multi-Party Key Exchange in the Bounded Storage Model

In this section, we extend the two-party NIKE in the BSM [14] to a multi-party one and show the optimality of our construction up to constant factors within the natural class of schemes satisfying extendability and local key independence by giving a lower bound for the users' storage consumption. All algorithms in this section are considered as streaming word-RAMs except for the storage function applied by the adversary, which can be any unbounded function with limited output size.

5.1 Our Construction

Pairwise independence. We first recall the definitions of uniformly and approximately pairwise independence, (approximately) strongly 2-universal hash function family, and 2-universal hash function family, and recall their instantiations in Figure 13. [1]

Definition 5.1 (Uniform pairwise independence). A sequence of random variables $x_1, \cdots, x_q$ with alphabet $\mathcal{Y}$ is uniformly pairwise independent if for any $1 \leq i < j \leq q$ and any $\hat{y}_1, \hat{y}_2 \in \mathcal{Y}$, we have $\Pr[x_i = \hat{y}_1 \wedge x_j = \hat{y}_2] = 1/|\mathcal{Y}|^2$.

Definition 5.2 (Approximate pairwise independence [42, 14]). A sequence of random variables $x_1, \cdots, x_q$ with alphabet $\mathcal{Y}$ is approximately pairwise independent if for any $1 \leq i < j \leq q$ and any distinct $\hat{y}_1, \hat{y}_2 \in \mathcal{Y}$, we have $\Pr[x_i = \hat{y}_1 \wedge x_j = \hat{y}_2] = 1/(|\mathcal{Y}|(|\mathcal{Y}| - 1))$.

[1] The approximate pairwise independence and approximate 2-universal hash function family are defined in [42, 14] in an implicit way.

Definition 5.3 (Strongly 2-universal hash function family). A set $\mathcal{H}$ of functions with domain $\mathcal{X}$ and range $\mathcal{Y}$ is said to be a *strongly 2-universal hash function family* if for any distinct $x_1, x_2 \in \mathcal{X}$, $h(x_1)$ and $h(x_2)$ are uniformly pairwise independent for $h \xleftarrow{\$} \mathcal{H}$.

Definition 5.4 (Approximate strongly 2-universal hash function [42, 14]). The definition of *approximate strongly 2-universal hash function family* is exactly the same as Definition 5.3 except that we replace "uniformly pairwise independent" by "approximately pairwise independent".

Definition 5.5 (2-universal hash function family). A set $\mathcal{H}$ of functions with domain $\mathcal{X}$ and range $\mathcal{Y}$ is said to be a *2-universal hash function family* if for any distinct $x_1, x_2 \in \mathcal{X}$, we have $\Pr[g(x_1) = g(x_2)] \leq 1/|\mathcal{Y}|$, where $g \xleftarrow{\$} \mathcal{H}$.

$$\begin{array}{l} \mathcal{F}_n = \{f(x) = a_1 \cdot x + a_0 \mid a_0, a_1 \in GF(n)\} \\ \mathcal{H}_n = \{h(x) = a_1 \cdot x + a_0 \mid a_0, a_1 \in GF(n) \wedge a_1 \neq 0\} \\ \mathcal{G}_{\lambda,r} = \{g(x) = \mathsf{MSB}_r(a \cdot x) \mid a \in GF(2^\lambda)\} \end{array}$$

Figure 13: The definitions of $\mathcal{F}_n$, $\mathcal{H}_n$, and $\mathcal{G}_{\lambda,r}$. In this figure, all operations involved in f and h (respectively, g) are within the field $GF(n)$ (respectively, $GF(2^\lambda)$). One can see that the outputs of any function in $\mathcal{H}_n$ with distinct inputs must be all distinct.

Lemma 5.6 ([14, 42]). *For any finite field $GF(n)$ and any $\lambda, r \in \mathbb{N}$, $\mathcal{F}_n$ (respectively, $\mathcal{H}_n$) is a strongly 2-universal hash function family (respectively, approximate strongly 2-universal hash function family), and $\mathcal{G}_{\lambda,r}$ is a 2-universal hash function family.*

Parameters. Next we follow [14] to define several parameters. Let $\lambda \in \mathbb{N}$ be the security parameter and $\mathsf{n_p}$ be the number of parties. Let $\mathsf{m_a}, n$ be integers such that $\mathsf{m_a} < n$. We define q, ϵ_1, ϵ_2, r, and θ satisfying the following equations:

- $\delta = \min\{0.9453, (1/n)(n - \mathsf{m_a} - \log(1/\epsilon_1))\}$,
- ρ such that $H_b(\rho) + \rho \log(1/\delta) + 1/n = \delta$,
- $\lambda = (q/2) \cdot (q/n)^{\mathsf{n_p}-1} = \lfloor (\rho\epsilon_2^2 - \rho 2^{-\rho n \log(1/\delta)-2})^{-1} \rfloor$,
- $r = \lfloor \log\theta + \rho\lambda/2 - 1 \rfloor$,

where H_b is the *binary entropy function* defined as $H_b(\rho) = -\rho\log\rho - (1-\rho)\log(1-\rho)$. As noted in [14], when $n >> \mathsf{m_a}$, we have $\delta = 0.9453$, $\rho = 1/3$, $\lambda \approx 3/\epsilon_2^2$, and $r \approx \lambda/6$.

Before giving our construction, we recall the following two theorems. The first one is the main theorem in [14] for privacy amplification, and the second one is a direct combination of Chebyshev's inequality and Markov's inequality.

Theorem 5.7 ([14]). *Let $\lambda \in \mathbb{N}$ and $GF(n)$ be a finite field. Let $\mathcal{A}$ be any adversary with the memory bound $\mathsf{m_a} < n$, $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $\mathbf{z} = \mathcal{A}(\mathsf{urs})$, $g \xleftarrow{\$} \mathcal{G}_{\lambda,r}$, $\mathbf{s} = (s_i)_{i\in[\lambda]}$ be a sequence of approximately pairwise independent indices over $GF(n)$, and $\mathsf{key} = g(\mathsf{urs_s})$. There exists a security event E such that*

$$\exists \mathsf{E}: \ \Pr[\mathsf{E}] \geq 1 - \epsilon_1 - \epsilon_2 \text{ and } I(\mathsf{key}; g, \mathbf{z}, \mathbf{s} \mid \mathsf{E}) \leq \theta.^2$$

Theorem 5.8 *For any $n \in \mathbb{N}$, let $(t_i)_{i\in[n]}$ be a sequence of pairwise independent variables. Then for any $a > 0$, we have*

$$\Pr[|\sum_{i=1}^{n} t_i - \sum_{i=1}^{n} \mathbb{E}[t_i]| \geq a] \leq \frac{\sum_{i=1}^{n} \mathsf{Var}[t_i]}{a^2}.$$

Construction. Let $\mathcal{H}_n$ be the approximately strongly 2-universal hash function family, and $\mathcal{G}_{\lambda,r}$ be the 2-universal hash function family, as defined in Figure 13. Let $\mathcal{C}_1$ be the class of streaming word-RAM families with memory bounds $4\log n + \lambda + q$, where $q = n\sqrt[n]{2\lambda/n}$. Throughout this construction, the parameters satisfy $q \leq n$. Our construction of multi-party NIKE in the BSM is defined as in Figure 14.

Theorem 5.9 BSMKE *is a $\mathcal{C}_1$-$\mathsf{n_p}$-NIKE with $2/\lambda$-correctness and $(\mathsf{m_a}, \epsilon_1 + \epsilon_2 + 2/\lambda, \theta)$-security in the BSM.*

[2]The original theorem asserts that $I(\mathsf{key}; g \mid \mathbf{z}, \mathbf{s}, \mathsf{E}) \leq \theta$, while their proof clearly implies our version.

$\underline{\mathsf{Gen}_\lambda(\mathsf{urs} \in \{0,1\}^n)}$:

Set $\mathsf{pk}_i = (h_i, g_i)$ where $h_i \xleftarrow{\$} \mathcal{H}_n$ and $g_i \xleftarrow{\$} \mathcal{G}_{\lambda,r}$

Set $\mathsf{sk}_i = (h_i, \mathsf{urs}_{\mathcal{S}_i})$ where $\mathcal{S}_i$ is the set of indices in $(h_i(j))_{j\in[q]}$

Return $(\mathsf{pk}_i, \mathsf{sk}_i)$

$\underline{\mathsf{Share}_\lambda(i, \mathsf{sk}_i, (\mathsf{pk}_j)_{j\in[\mathsf{n_p}]})}$

Parse $\mathsf{sk}_i = (h_i, \mathsf{urs}_{\mathcal{S}_i})$

Remove bits in $\mathsf{urs}_{\mathcal{S}_i}$ with indices not in $\mathcal{S}_1 \cap \cdots \cap \mathcal{S}_{\mathsf{n_p}}$ by computing $(h_i(j))_{i\in[\mathsf{n_p}],j\in[q]}$ in a streaming manner

For each $j \in [q]$, let $u_j = h_{\mathsf{n_p}}(j)$ if $h_{\mathsf{n_p}}(j) \in \mathcal{S}_1 \cap \cdots \cap \mathcal{S}_{\mathsf{n_p}}$, and let $u_j = \perp$ otherwise

Let $\mathbf{s}$ be obtained from $(u_j)_{j\in[q]}$ by removing all $\perp$'s while preserving the order of j

Abort the whole protocol if $|\mathbf{s}| < \lambda$

Return $\mathsf{key} = g_{\mathsf{n_p}}(\mathsf{urs}_{\mathbf{s}_\lambda})$ where $\mathbf{s}_\lambda$ denotes the first λ entries of $\mathbf{s}$

Figure 14: The definition of BSMKE. Recall that for any set $\mathcal{S}$, $\mathsf{urs}_{\mathcal{S}}$ denotes the sequence of all bits in urs with indices (ordered lexicographically) in $\mathcal{S}$. For any sequence $\mathbf{s}$, $\mathsf{urs}_{\mathbf{s}}$ follows the order of $\mathbf{s}$.

Proof. **Complexity.** First we note that each party P_i, for $i \in [\mathsf{n_p}]$, consumes $4\log n + \lambda + q$ bits of memory to store q bits in urs, a 2-universal hash function in $\mathcal{G}_{\lambda,r}$, and at most two approximately strongly 2-universal hash functions in $\mathcal{H}_n$ for comparison at any point in the protocol.

Correctness. To prove correctness, it suffices to show that the protocol aborts with probability at most $2/\lambda$.

Let $\mathcal{S}_i = \{h_i(j) : j \in [q]\}$ for $i \in [\mathsf{n_p}]$, and define

$$Y_j = \begin{cases} 1 & \text{if } h_{\mathsf{n_p}}(j) \in \mathcal{S}_1 \cap \cdots \cap \mathcal{S}_{\mathsf{n_p}-1}, \\ 0 & \text{otherwise} \end{cases} \quad \text{and } Y = \sum_{j=1}^{q} Y_j.$$

Since $h_{\mathsf{n_p}} \xleftarrow{\$} \mathcal{H}_n$ is injective on the q queried points, Y is exactly the number of distinct common indices in $\mathcal{S}_1 \cap \cdots \cap \mathcal{S}_{\mathsf{n_p}}$. Thus the protocol aborts iff $Y < \lambda$.

For each fixed $a \in GF(n)$ and each $i \in [\mathsf{n_p} - 1]$, $\Pr[a \in \mathcal{S}_i] = q/n$. Since $\mathcal{S}_1, \ldots, \mathcal{S}_{\mathsf{n_p}-1}$ are independent and $h_{\mathsf{n_p}}(j)$ is uniform and independent of them,

$$\mathbb{E}[Y] = \sum_{j=1}^{q} \mathbb{E}[Y_j] = \sum_{j=1}^{q} \Pr[h_{\mathsf{n_p}}(j) \in \mathcal{S}_1 \cap \cdots \cap \mathcal{S}_{\mathsf{n_p}-1}] = \sum_{j=1}^{q} \left(\frac{q}{n}\right)^{\mathsf{n_p}-1} = q\left(\frac{q}{n}\right)^{\mathsf{n_p}-1} = 2\lambda.$$

We next bound the variance. For $j \neq j'$,

$$\begin{aligned}
&\mathbb{E}[Y_j Y_{j'}] = \Pr[Y_j = Y_{j'} = 1] \\
&= \sum_{\substack{a,b \in GF(n) \\ a \neq b}} \Pr[h_{\mathsf{n_p}}(j) = a, h_{\mathsf{n_p}}(j') = b] \Pr[a, b \in \mathcal{S}_1 \cap \cdots \cap \mathcal{S}_{\mathsf{n_p}-1}] \\
&= \sum_{\substack{a,b \in GF(n) \\ a \neq b}} \frac{1}{n(n-1)} \Pr[a, b \in \mathcal{S}_1 \cap \cdots \cap \mathcal{S}_{\mathsf{n_p}-1}] \ (\because \text{approximate pairwise independence of } h_{\mathsf{n_p}}) \\
&= \sum_{\substack{a,b \in GF(n) \\ a \neq b}} \frac{1}{n(n-1)} \prod_{i=1}^{\mathsf{n_p}-1} \Pr[a, b \in \mathcal{S}_i] \ (\because \text{independence of } (\mathcal{S}_i)_{i=1}^{\mathsf{n_p}-1}) \\
&= \sum_{\substack{a,b \in GF(n) \\ a \neq b}} \frac{1}{n(n-1)} \left(\frac{q(q-1)}{n(n-1)}\right)^{\mathsf{n_p}-1} \ (\because \text{injectivity and uniformity of } \{h_i\}_{i=1}^{\mathsf{n_p}-1}) \\
&= \left(\frac{q(q-1)}{n(n-1)}\right)^{\mathsf{n_p}-1} \leq \left(\frac{q}{n}\right)^{2(\mathsf{n_p}-1)} = \mathbb{E}[Y_j]\mathbb{E}[Y_{j'}].
\end{aligned}$$

As a result, we have

$$\mathsf{Var}[Y] = \sum_{j=1}^{q} \mathsf{Var}[Y_j] + 2 \sum_{1 \leq j < j' \leq q} (\mathbb{E}[Y_j Y_{j'}] - \mathbb{E}[Y_j]\mathbb{E}[Y_{j'}]) \leq \sum_{j=1}^{q} \mathsf{Var}[Y_j] \leq \sum_{j=1}^{q} \mathbb{E}[Y_j] = 2\lambda.$$

By Chebyshev's inequality,

$$\Pr[Y < \lambda] \leq \Pr[|Y - \mathbb{E}[Y]| \geq \lambda] \leq \frac{\mathsf{Var}[Y]}{\lambda^2} \leq \frac{2}{\lambda}.$$

Therefore, the protocol aborts with probability at most $2/\lambda$.

Security. Let $f : \{0,1\}^n \to \{0,1\}^{\mathsf{m_a}}$ be any storage function applied by the adversary. Let $(h_i, g_i)_{i \in [\mathsf{n_p}]}$ be the honestly generated public keys. Let $\mathbf{u} = (u_i)_{i \in [q]}$ be the sequence such that $u_i = h_{\mathsf{n_p}}(i)$ if there are indices $i_1, \cdots, i_{\mathsf{n_p}-1}$ satisfying $h_1(i_1) = \cdots = h_{\mathsf{n_p}-1}(i_{\mathsf{n_p}-1}) = h_{\mathsf{n_p}}(i)$ and $u_i = \perp$ otherwise, and let $\mathcal{S}$ be the set of all indices of u_i restricted to $u_i \neq \perp$, i.e., $\mathbf{s} = \mathbf{u}_{\mathcal{S}}$ the sequence $\mathbf{u}$ restricted to $u_i \neq \perp$. We have the following lemma. Let Good be the event that the protocol does not abort, i.e., $\mathsf{Good} := \{|\mathbf{s}| \geq \lambda\}$.

Lemma 5.10 $\mathbf{s}$ *is approximately pairwise independent.*

Proof. Let $\hat{x}_1, \hat{x}_2$ be any distinct elements in $GF(n)$. Due to approximately pairwise independence, we have $\Pr[h_{\mathsf{n_p}}(i) = \hat{x}_1 \wedge h_{\mathsf{n_p}}(j) = \hat{x}_2] = \frac{1}{n(n-1)}$ for all distinct $i, j \in [q]$, and $\Pr[\exists i_l, j_l : h_l(i_l) = \hat{x}_1 \wedge h_l(j_l) = \hat{x}_2] = \frac{q(q-1)}{n(n-1)}$ for all $l \in [\mathsf{n_p} - 1]$. Hence, for any distinct $i, j \in [q]$, we have

$$\begin{aligned}
\Pr[u_i = \hat{x}_1 \wedge u_j = \hat{x}_2] =& \Pr[\exists (i_l, j_l)_{l \in [\mathsf{n_p}-1]} : h_1(i_1) = \cdots = h_{\mathsf{n_p}-1}(i_{\mathsf{n_p}-1}) = \hat{x}_1 = h_{\mathsf{n_p}}(i) \wedge \\
& h_1(j_1) = \cdots = h_{\mathsf{n_p}-1}(j_{\mathsf{n_p}-1}) = \hat{x}_2 = h_{\mathsf{n_p}}(j)] \\
=& \Pr[h_{\mathsf{n_p}}(i) = \hat{x}_1 \wedge h_{\mathsf{n_p}}(j) = \hat{x}_2] \cdot \prod_{l=1}^{\mathsf{n_p}-1} \Pr[\exists i_l, j_l : h_l(i_l) = \hat{x}_1 \wedge h_l(j_l) = \hat{x}_2] \\
=& \frac{1}{n(n-1)} \cdot \left(\frac{q(q-1)}{n(n-1)}\right)^{\mathsf{n_p}-1},
\end{aligned}$$

and

$$\begin{aligned}
\Pr[u_i \neq \perp \wedge u_j \neq \perp] =& \Pr[\exists (i_l, j_l)_{l \in [\mathsf{n_p}-1]} : h_1(i_1) = \cdots = h_{\mathsf{n_p}-1}(i_{\mathsf{n_p}-1}) = h_{\mathsf{n_p}}(i) \wedge \\
& h_1(j_1) = \cdots = h_{\mathsf{n_p}-1}(j_{\mathsf{n_p}-1}) = h_{\mathsf{n_p}}(j)] \\
=& \prod_{l=1}^{\mathsf{n_p}-1} \Pr[\exists i_l, j_l : h_l(i_l) = h_{\mathsf{n_p}}(i) \wedge h_l(j_l) = h_{\mathsf{n_p}}(j)] = \left(\frac{q(q-1)}{n(n-1)}\right)^{\mathsf{n_p}-1},
\end{aligned}$$

i.e.,

$$\Pr[u_i = \hat{x}_1 \wedge u_j = \hat{x}_2 | u_i \neq \perp \wedge u_j \neq \perp] = \frac{\Pr[u_i = \hat{x}_1 \wedge u_j = \hat{x}_2]}{\Pr[u_i \neq \perp \wedge u_j \neq \perp]} = \frac{1}{n(n-1)}.$$

This completes the proof of Lemma 5.10. □

By the correctness analysis above, we have $\Pr[\mathsf{Good}] \geq 1 - 2/\lambda$. On Good, let $\mathbf{s}_\lambda$ denote the first λ entries of $\mathbf{s}$. By Lemma 5.10 and Theorem 5.7, there exists an event E_0 such that

$$\Pr[\mathsf{E}_0 \mid \mathsf{Good}] \geq 1 - \epsilon_1 - \epsilon_2 \text{ and } I(\mathsf{key}; g_{\mathsf{n_p}}, \mathbf{z}, \mathbf{s}_\lambda \mid \mathsf{E}_0, \mathsf{Good}) \leq \theta,$$

where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $\mathbf{z} = f(\mathsf{urs})$, and $\mathsf{key} = g_{\mathsf{n_p}}(\mathsf{urs}_{\mathbf{s}_\lambda})$. Let $\mathsf{E} = \mathsf{E}_0 \cap \mathsf{Good}$. Then we have

$$\Pr[\mathsf{E}] \geq 1 - \epsilon_1 - \epsilon_2 - 2/\lambda.$$

The event E_0 is chosen using only $(\mathsf{urs}, g_{\mathsf{n_p}}, \mathbf{z}, \mathbf{s}_\lambda)$ and not the remaining public-key components. Since $\mathbf{s}_\lambda$ is determined by $(h_i)_{i\in[\mathsf{n_p}]}$ and $\mathsf{key} = g_{\mathsf{n_p}}(\mathsf{urs}_{\mathbf{s}_\lambda})$, those remaining components give no additional information about key once $(g_{\mathsf{n_p}}, \mathbf{z}, \mathbf{s}_\lambda, \mathsf{E})$ is fixed. Hence

$$I(\mathsf{key}; (\mathbf{z}, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}) \mid \mathsf{E}) = I(\mathsf{key}; g_{\mathsf{n_p}}, \mathbf{z}, \mathbf{s}_\lambda \mid \mathsf{E}) \leq \theta.$$

Thus, BSMKE satisfies $(\mathsf{m_a}, \epsilon_1 + \epsilon_2 + 2/\lambda, \theta)$-security in the BSM.

Putting all the above together, Theorem 5.9 follows. □

Example with asymptotic complexity. Let $n = \lambda^{\mathsf{n_p}+1}$, $\mathsf{m_a} = \lambda^{\mathsf{n_p}+1}/2$, $\epsilon_1 = 1/\lambda^{\mathsf{n_p}}$, $\epsilon_2 = \Theta(1/\sqrt{\lambda})$, and $\theta = \mathsf{negl}(\lambda)$. We have $r = O(\lambda)$. In this case, each $\mathsf{P}_{i\in[\mathsf{n_p}]}$ consumes about $q = n\sqrt[\mathsf{n_p}]{2\lambda/n} = O(\lambda^{\mathsf{n_p}})$ bits of memory and the communication cost, i.e., the total size of the public keys, is $O(\lambda)$.

Concrete example. We now give a concrete example of the system based on the data given in [14]. Assume that all users have access to a 40 Gbit/s broadcast channel used for 2×10^5 seconds (about two days), we have $n = 8.6 \times 10^{15}$. Let $\mathsf{m_a} = 4.5 \times 10^{15}$. With $\theta = 10^{-20}$ and error probabilities $\epsilon_1 = 10^{-20}$ and $\epsilon_2 = 10^{-3}$, the parameters are $\delta = 0.476$, $\rho = 0.077$, $\lambda = 1.3 \times 10^7$, and $r = 5.0 \times 10^5$. Except with probability about 10^{-3}, Eve knows less than 10^{-20} bits about the 61 KByte session key. To obtain this r-bit session key in a 3-party protocol, each party consumes about 1.2×10^{13} bits of memory, and the communication consists of about 3.9×10^7 bits in total.

Extensions. Similar to our constructions in previous sections, our construction in the BSM can also be converted into a BPRF and a multi-party IB-NIKE. We refer the reader to Section 6 for the details.

5.2 Limitations of Multi-Party NIKEs in the BSM

In this section, we show that for any NIKE for a constant number of parties that satisfies (restricted) extendability and local key independence, the entropy of the shared key conditioned on the public keys is bounded by roughly $\mathsf{m_u}^{\mathsf{n_p}}/\mathsf{m_a}^{\mathsf{n_p}-1}$, where $\mathsf{m_u}$ and $\mathsf{m_a}$ are the memory bounds for users and adversaries respectively. This suggests that $\mathsf{m_u}$ must be at least about $\sqrt[\mathsf{n_p}]{\lambda \cdot \mathsf{m_a}^{\mathsf{n_p}-1}}$ for sharing λ bits, and thus our multi-party NIKE in the BSM, which satisfies local key independence as we show later, achieves optimality up to constant factors within this natural class.

Before giving our negative result, we recall several lemmata and theorems below.

Lemma 5.11 *For two random variables x, y, we have $H(x) \geq H(x \mid y)$.*

Theorem 5.12 (Data processing inequality). *For two random variables x, y and deterministic functions f, g, we have*

$$H(f(x)) \leq H(x) \text{ and } I(f(x); g(y)) \leq I(x; y).$$

The same inequalities hold conditioned on any additional random variable.

Lemma 5.13 (Lemma 1 in [19]). *Let y, z, $(z_i)_{i\in[n]}$ be a sequence of random variables such that conditioned on y, the variables $z, z_1, \cdots, z_n$ are independent and identically distributed, i.e.,*

$$\Pr[z = \widehat{z} \wedge z_1 = \widehat{z}_1 \wedge \cdots \wedge z_n = \widehat{z}_n \mid y = \widehat{y}] = \Pr[z = \widehat{z} \mid y = \widehat{y}] \prod_{i=1}^{n} \Pr[z = \widehat{z}_i \mid y = \widehat{y}]$$

for all $\widehat{y}, \widehat{z}, (\widehat{z}_i)_{i\in[n]}$. Then $I(y; z \mid z_1, \cdots, z_n) \leq H(y)/(n+1)$.

Theorem 5.14 (Theorem 1 in [32]). [3] *Let x, y, z, c, s, s' be random variables such that $H(c \mid y) = 0$, $H(s \mid x, c) = 0$, and $H(s' \mid y, c) = 0$. Then*

$$H(s) \le I(x; y \mid z) + H(s \mid s') + I(s; c, z).$$

Our negative result. Below we give our negative result showing limitations of multi-party NIKE in the BSM.

Theorem 5.15 *Let $\mathcal{C}_1$ be the class of streaming word-RAMs with memory bounds $\mathsf{m_u}$. Let $\mathsf{NIKE} = \{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\}$ be a $\mathcal{C}_1$-$\mathsf{n_p}$-NIKE for any constant $\mathsf{n_p} \ge 2$ in the BSM with URS length n and satisfying 0-restricted extendability and $((\mu_i)_{i\in[\mathsf{n_p}]}, \lfloor \mathsf{m_a}/\mathsf{m_u} \rfloor)$-local key independence for any $\mathsf{m_a} \ge \mathsf{m_u}$, where the associated combining and extracting algorithm families are $\{\mathsf{Combine}_\lambda, \mathsf{Extract}_\lambda\}$.* [4] *Then*

$$H(\mathsf{key} \mid (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}) \le \mathsf{m_u}^{\mathsf{n_p}}/\mathsf{m_a}^{\mathsf{n_p}-1} + 2\sum_{i=2}^{\mathsf{n_p}} (\mathsf{m_u}/\mathsf{m_a})^{\mathsf{n_p}-i}\mu_i,$$

where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ *for all* $i \in [\mathsf{n_p}]$, *and* $\mathsf{key} = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]})$.

Proof. Let $\mathsf{localkey}_{[i]} = \mathsf{Combine}_\lambda(\mathsf{localkey}_{[i-1]}, \mathsf{pk}_i)$ for all $i > 1$, where $\mathsf{localkey}_{[1]} = \mathsf{sk}_1$. Let $M_i = (\mathsf{pk}_i, \mathsf{sk}_i)$ and $\mathsf{localkey}'_{[i]} = \mathsf{Combine}_\lambda(\mathsf{sk}_i, (\mathsf{pk}_j)_{i>j\ge 1})$ for all $i \in [\mathsf{n_p}]$. Let $t = \lfloor \mathsf{m_a}/\mathsf{m_u} \rfloor$, sample $M'_r = (\mathsf{pk}'_r, \mathsf{sk}'_r) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for all $r \in [t]$, and set $M_E = (M'_r)_{r\in[t]}$.

Fix any $i > 1$. We apply Theorem 5.14 with $x = (\mathsf{localkey}_{[i-1]}, (\mathsf{pk}_j)_{j\in[i-1]})$, $y = (M_i, (\mathsf{pk}_j)_{j\in[i-1]})$, $z = (M_E, (\mathsf{pk}_j)_{j\in[i-1]})$, $c = \mathsf{pk}_i$, $s = \mathsf{localkey}_{[i]}$, and $s' = \mathsf{localkey}'_{[i]}$. By the definition of M_i, 0-restricted extendability, and determinism of $\mathsf{Combine}_\lambda$, we have

$$\begin{gathered} H(\mathsf{pk}_i \mid M_i, (\mathsf{pk}_j)_{j\in[i-1]}) = 0, \\ H(\mathsf{localkey}_{[i]} \mid \mathsf{localkey}_{[i-1]}, (\mathsf{pk}_j)_{j\in[i-1]}, \mathsf{pk}_i) = 0, \\ H(\mathsf{localkey}'_{[i]} \mid M_i, (\mathsf{pk}_j)_{j\in[i-1]}, \mathsf{pk}_i) = 0, \\ \Pr[\mathsf{localkey}_{[i]} \ne \mathsf{localkey}'_{[i]}] = 0. \end{gathered}$$

Therefore, by Theorem 5.14,

$$\begin{aligned} H(\mathsf{localkey}_{[i]}) &\le I((\mathsf{localkey}_{[i-1]}, (\mathsf{pk}_j)_{j\in[i-1]}); (M_i, (\mathsf{pk}_j)_{j\in[i-1]}) \mid M_E, (\mathsf{pk}_j)_{j\in[i-1]}) \\ &\quad + H(\mathsf{localkey}_{[i]} \mid \mathsf{localkey}'_{[i]}) + I(\mathsf{localkey}_{[i]}; \mathsf{pk}_i, M_E, (\mathsf{pk}_j)_{j\in[i-1]}) \\ &= I(\mathsf{localkey}_{[i-1]}; M_i \mid M_E, (\mathsf{pk}_j)_{j\in[i-1]}) + I(\mathsf{localkey}_{[i]}; \mathsf{pk}_i, M_E, (\mathsf{pk}_j)_{j\in[i-1]}). \end{aligned}$$

By the first condition in Definition 2.4,

$$I(\mathsf{localkey}_{[i]}; \mathsf{pk}_i, M_E, (\mathsf{pk}_j)_{j\in[i-1]}) \le \mu_i.$$

Thus,

$$H(\mathsf{localkey}_{[i]}) \le I(\mathsf{localkey}_{[i-1]}; M_i \mid M_E, (\mathsf{pk}_j)_{j\in[i-1]}) + \mu_i.$$

Next, by the second condition in Definition 2.4, we have

$$I(\mathsf{localkey}_{[i-1]}; M_i \mid M_E, (\mathsf{pk}_j)_{j\in[i-1]}) \le I(\mathsf{localkey}_{[i-1]}; M_i \mid M_E, (\mathsf{pk}_j)_{j\in[i-1]}, \mathsf{urs}) + \mu_i.$$

Conditioned on urs and $(\mathsf{pk}_j)_{j\in[i-1]}$, the variables $M_i, M'_1, \ldots, M'_t$ are independent copies generated in the same way. Hence, by Lemma 5.13, we have

$$I(\mathsf{localkey}_{[i-1]}; M_i \mid M_E, (\mathsf{pk}_j)_{j\in[i-1]}, \mathsf{urs}) \le \frac{H(\mathsf{localkey}_{[i-1]} \mid \mathsf{urs}, (\mathsf{pk}_j)_{j\in[i-1]})}{t+1} \le \frac{H(\mathsf{localkey}_{[i-1]})}{t+1}.$$

Since $t = \lfloor \mathsf{m_a}/\mathsf{m_u} \rfloor$, we have $1/(t+1) \le \mathsf{m_u}/\mathsf{m_a}$. Combining the above inequalities gives

$$H(\mathsf{localkey}_{[i]}) \le (\mathsf{m_u}/\mathsf{m_a}) H(\mathsf{localkey}_{[i-1]}) + 2\mu_i.$$

[3] This is the special case of Theorem 1 in [32] where the public transcript is the single message c.

[4] We use $t = \lfloor \mathsf{m_a}/\mathsf{m_u} \rfloor$. This only affects constant factors; for readability one may assume that $\mathsf{m_a}/\mathsf{m_u}$ is an integer.

We now iterate this recurrence. Since $H(\mathsf{localkey}_{[1]}) \le H(M_1) \le \mathsf{m_u}$, we have

$$H(\mathsf{localkey}_{[2]}) \le (\mathsf{m_u}/\mathsf{m_a})H(\mathsf{localkey}_{[1]}) + 2\mu_2 \le {\mathsf{m_u}}^2/\mathsf{m_a} + 2\mu_2.$$

Assume that, for some $i \ge 3$,

$$H(\mathsf{localkey}_{[i-1]}) \le {\mathsf{m_u}}^{i-1}/{\mathsf{m_a}}^{i-2} + 2\sum_{\ell=2}^{i-1}(\mathsf{m_u}/\mathsf{m_a})^{i-1-\ell}\mu_\ell.$$

Then

$$\begin{aligned} H(\mathsf{localkey}_{[i]}) &\le (\mathsf{m_u}/\mathsf{m_a})H(\mathsf{localkey}_{[i-1]}) + 2\mu_i \\ &\le {\mathsf{m_u}}^{i}/{\mathsf{m_a}}^{i-1} + 2\sum_{\ell=2}^{i}(\mathsf{m_u}/\mathsf{m_a})^{i-\ell}\mu_\ell. \end{aligned}$$

Thus,

$$H(\mathsf{localkey}_{[\mathsf{n_p}]}) \le {\mathsf{m_u}}^{\mathsf{n_p}}/{\mathsf{m_a}}^{\mathsf{n_p}-1} + 2\sum_{i=2}^{\mathsf{n_p}}(\mathsf{m_u}/\mathsf{m_a})^{\mathsf{n_p}-i}\mu_i.$$

Finally, conditioning on the public keys,

$$\begin{aligned} H(\mathsf{key} \mid (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}) &= H(\mathsf{Extract}_\lambda(\mathsf{localkey}_{[\mathsf{n_p}]}, \mathsf{pk}_{\mathsf{n_p}}) \mid (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}) \\ &\le H(\mathsf{localkey}_{[\mathsf{n_p}]} \mid (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}) \\ &\le H(\mathsf{localkey}_{[\mathsf{n_p}]}) \\ &\le {\mathsf{m_u}}^{\mathsf{n_p}}/{\mathsf{m_a}}^{\mathsf{n_p}-1} + 2\sum_{i=2}^{\mathsf{n_p}}(\mathsf{m_u}/\mathsf{m_a})^{\mathsf{n_p}-i}\mu_i. \end{aligned}$$

This completes the proof of Theorem 5.15. □

Optimality of BSMKE. Let $\mathsf{n_p}$, n, λ, and q be as in Section 5. By Section 6.4, BSMKE satisfies 0-extendability, while aborts are covered by the $2/\lambda$-correctness bound in Theorem 5.9. Thus, it remains to verify local key independence. The next lemma gives the required bounds and shows that our instantiation matches the lower bound up to constant factors.

Lemma 5.16 *For every $t \in \mathbb{N}$, BSMKE satisfies $((\mu_i)_{i\in[\mathsf{n_p}]}, t)$-local key independence in the BSM with*

$$\mu_i = O(t \cdot q^{i+1}/n^i + q^i/n^{i-1} + \log n) \text{ for all } i \in [\mathsf{n_p}].$$

Proof. For notational simplicity, let $\mathcal{S}_i$ be the index set stored by the i-th user, and let $\mathcal{S}_{[i]} = \mathcal{S}_1 \cap \cdots \cap \mathcal{S}_i$, so that $\mathsf{localkey}_{[i]} = \mathsf{urs}_{\mathcal{S}_{[i]}}$. We suppress the compact public hash descriptions defining these sets; their size is $O(\log n)$ and is absorbed into the $O(\log n)$ term.

Fix $i \in [\mathsf{n_p}]$, and let $\mathcal{S}'_j$ be the index set stored in sk'_j. The public keys reveal the length of the variable-length string $\mathsf{urs}_{\mathcal{S}_{[i]}}$, which costs at most $O(\log n)$ bits. Once the public keys fix the index sets, the only correlation between $\mathsf{urs}_{\mathcal{S}_{[i]}}$ and $\mathsf{urs}_{\mathcal{S}'_1\cup\cdots\cup\mathcal{S}'_t}$ comes from their overlapping positions. Hence, we have

$$\begin{aligned} & I\left(\mathsf{localkey}_{[i]}; (\mathsf{pk}_j)_{j\in[i]}, (\mathsf{pk}'_j, \mathsf{sk}'_j)_{j\in[t]}\right) \\ =& I\left(\mathsf{urs}_{\mathcal{S}_{[i]}}; (\mathsf{pk}_j)_{j\in[i]}, (\mathsf{pk}'_j)_{j\in[t]}\right) + I\left(\mathsf{urs}_{\mathcal{S}_{[i]}}; (\mathsf{sk}'_j)_{j\in[t]} \mid (\mathsf{pk}_j)_{j\in[i]}, (\mathsf{pk}'_j)_{j\in[t]}\right) \\ \le& O(\log n) + I\left(\mathsf{urs}_{\mathcal{S}_{[i]}}; \mathsf{urs}_{\mathcal{S}'_1\cup\cdots\cup\mathcal{S}'_t} \mid (\mathsf{pk}_j)_{j\in[i]}, (\mathsf{pk}'_j)_{j\in[t]}\right) \\ \le& O(\log n) + \mathbb{E}\left[\left|\mathcal{S}_{[i]} \cap (\mathcal{S}'_1 \cup \cdots \cup \mathcal{S}'_t)\right|\right] \\ \le& O(\log n) + \sum_{j=1}^{t}\mathbb{E}[|\mathcal{S}_{[i]} \cap \mathcal{S}'_j|] \\ =& O(\log n) + t\sum_{a\in GF(n)}\Pr[a \in \mathcal{S}_{[i]} \cap \mathcal{S}'_1] \\ =& O(\log n) + t \cdot n(q/n)^{i+1} \\ =& O(t \cdot q^{i+1}/n^i + \log n). \end{aligned}$$

This proves the first condition.

It remains to verify the second condition. For $i > 1$, conditioned on urs and $(\mathsf{pk}_j)_{j\in[i-1]}$, the value $\mathsf{localkey}_{[i-1]} = \mathsf{urs}_{\mathcal{S}_{[i-1]}}$ is fixed. Therefore,

$$I(\mathsf{localkey}_{[i-1]}; (\mathsf{pk}_i, \mathsf{sk}_i) \mid (\mathsf{pk}_j)_{j\in[i-1]}, (\mathsf{pk}'_j, \mathsf{sk}'_j)_{j\in[t]}, \mathsf{urs}) = 0.$$

Without conditioning on urs, the new key pair $(\mathsf{pk}_i, \mathsf{sk}_i)$ can reveal information about $\mathsf{localkey}_{[i-1]}$ only through the overlap $\mathcal{S}_{[i-1]} \cap \mathcal{S}_i = \mathcal{S}_{[i]}$. Hence

$$I(\mathsf{localkey}_{[i-1]}; (\mathsf{pk}_i, \mathsf{sk}_i) \mid (\mathsf{pk}_j)_{j\in[i-1]}, (\mathsf{pk}'_j, \mathsf{sk}'_j)_{j\in[t]}) \le \mathbb{E}[|\mathcal{S}_{[i]}|] = \sum_{a\in GF(n)} \Pr[a \in \mathcal{S}_{[i]}] = n(q/n)^i = q^i/n^{i-1}.$$

Thus we have

$$\begin{aligned} &I(\mathsf{localkey}_{[i-1]}; (\mathsf{pk}_i, \mathsf{sk}_i) \mid (\mathsf{pk}_j)_{j\in[i-1]}, (\mathsf{pk}'_j, \mathsf{sk}'_j)_{j\in[t]}) \\ \le &I(\mathsf{localkey}_{[i-1]}; (\mathsf{pk}_i, \mathsf{sk}_i) \mid (\mathsf{pk}_j)_{j\in[i-1]}, (\mathsf{pk}'_j, \mathsf{sk}'_j)_{j\in[t]}, \mathsf{urs}) + q^i/n^{i-1}. \end{aligned}$$

Since $\mu_i = O(t \cdot q^{i+1}/n^i + q^i/n^{i-1} + \log n)$, both conditions in Definition 2.4 are satisfied. □

Remark. For the usual parameter choice $n = \Theta(\mathsf{m_a})$, $\mathsf{m_u} = \Theta(q)$, and $t = \lfloor \mathsf{m_a}/\mathsf{m_u} \rfloor$, Lemma 5.16 gives

$$t \cdot q^{i+1}/n^i = O((\mathsf{m_a}/\mathsf{m_u}) \cdot {\mathsf{m_u}}^{i+1}/{\mathsf{m_a}}^i) = O({\mathsf{m_u}}^i/{\mathsf{m_a}}^{i-1}) \text{ and } q^i/n^{i-1} = O({\mathsf{m_u}}^i/{\mathsf{m_a}}^{i-1}).$$

Hence, we have $\mu_i = O({\mathsf{m_u}}^i/{\mathsf{m_a}}^{i-1} + \log n)$. Plugging this into Theorem 5.15 yields

$$H(\mathsf{key} \mid (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}) \le O({\mathsf{m_u}}^{\mathsf{n_p}}/{\mathsf{m_a}}^{\mathsf{n_p}-1} + \log n).$$

When $\log n = o(\lambda)$, any scheme in this class with $H(\mathsf{key} \mid (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}) = \Omega(\lambda)$ must satisfy

$$\mathsf{m_u} = \Omega\left(\sqrt[\mathsf{n_p}]{\lambda \cdot {\mathsf{m_a}}^{\mathsf{n_p}-1}}\right).$$

Thus, the BSM construction is optimal up to constant factors within this class.

6 Fine-Grained Multi-Party IB-NIKE and BPRF

In this section, we propose multi-party IB-NIKE schemes in the fine-grained setting. We first give the definitions of fine-grained multi-party IB-NIKE and BPRF, and then give a generic construction of multi-party IB-NIKE based on BPRF. Next we present a generic construction of BPRF from multi-party NIKE with an additional property we refer to as extendability, which is satisfied by all our multi-party NIKE schemes. Throughout this section, we focus on IB-NIKE for a constant number of parties with a larger, constant-size identity space, and BPRF with constant-size input space.

6.1 Definitions

We now give the definitions of IB-NIKE and BPRF in the fine-grained setting.

Definition 6.1 (Identity-based non-interactive key exchange). A $\mathcal{C}_1$-$\mathsf{n_p}$-*identity-based non-interactive key exchange (IB-NIKE)* consists of three algorithm families $\mathsf{IB\text{-}NIKE} = \{\mathsf{IBSetup}_\lambda, \mathsf{IBExtract}_\lambda, \mathsf{IBShare}_\lambda\} \in \mathcal{C}_1$ such that:

- $\mathsf{IBSetup}_\lambda$ on input $\mathsf{urs} \in \{0,1\}^n$ for some $n = n(\lambda)$ outputs a master secret key ibmsk.
- $\mathsf{IBExtract}_\lambda$ is a deterministic algorithm that on input a master secret key ibmsk and an identity $\mathsf{id} \in \mathcal{ID}_\lambda$, where $\mathcal{ID}_\lambda$ is the identity space with *constant size*, outputs a secret key $\mathsf{ibsk}_{\mathsf{id}}$ for id.
- $\mathsf{IBShare}_\lambda$ is a deterministic algorithm that on input a list $\mathcal{I} \subseteq \mathcal{ID}_\lambda$ consisting of $\mathsf{n_p}$ identities and a secret key $\mathsf{ibsk}_{\mathsf{id}}$ for $\mathsf{id} \in \mathcal{I}$ outputs $\mathsf{ibshk} \in \mathcal{K}_\lambda \cup \{\perp\}$ for $\mathcal{I}$, where $\mathcal{K}_\lambda$ is the shared key space. We assume that the identities in $\mathcal{I}$ are always lexicographically ordered.

ϵ-*correctness* is satisfied if for all $\lambda \in \mathbb{N}$, all lists of $\mathsf{n_p}$ distinct identities $\mathcal{I} \subseteq \mathcal{ID}_\lambda$, all $\mathsf{id}_i, \mathsf{id}_j \in \mathcal{I}$, we have

$$\Pr[\mathsf{IBShare}_\lambda(\mathsf{ibsk}_{\mathsf{id}_i}, \mathcal{I}) = \mathsf{IBShare}_\lambda(\mathsf{ibsk}_{\mathsf{id}_j}, \mathcal{I})] \geq 1 - \epsilon,$$

where $\mathsf{ibsk}_{\mathsf{id}_i} = \mathsf{IBExtract}_\lambda(\mathsf{ibmsk}, \mathsf{id}_i)$ and $\mathsf{ibsk}_{\mathsf{id}_j} = \mathsf{IBExtract}_\lambda(\mathsf{ibmsk}, \mathsf{id}_j)$ for $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$ and $\mathsf{ibmsk} \xleftarrow{\$} \mathsf{IBSetup}_\lambda(\mathsf{urs})$.

$\mathcal{C}_2$-ϵ-*security* is satisfied if for any adversary $\mathcal{A} = \{\mathsf{adv}_\lambda\} \in \mathcal{C}_2$, all sufficiently large $\lambda \in \mathbb{N}$, and any list of $\mathsf{n_p}$ distinct identities $\mathcal{I}^* \subseteq \mathcal{ID}_\lambda$, we have

$$\begin{aligned}|\Pr[1 \xleftarrow{\$} \mathsf{adv}_\lambda(\mathsf{urs}, \mathsf{ibshk}, (\mathsf{ibsk}_{\mathsf{id}})_{\mathsf{id} \in \mathcal{ID}_\lambda \setminus \mathcal{I}^*})] \\ - \Pr[1 \xleftarrow{\$} \mathsf{adv}_\lambda(\mathsf{urs}, \mathsf{ibshk}', (\mathsf{ibsk}_{\mathsf{id}})_{\mathsf{id} \in \mathcal{ID}_\lambda \setminus \mathcal{I}^*})]| \leq \epsilon,\end{aligned}$$

where $\mathsf{ibmsk} \xleftarrow{\$} \mathsf{IBSetup}_\lambda(\mathsf{urs})$, $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $\mathsf{ibshk} = \mathsf{IBShare}_\lambda(\mathsf{ibsk}_{\mathsf{id}^*}, \mathcal{I}^*)$, $\mathsf{ibshk}' \xleftarrow{\$} \mathcal{K}_\lambda$, $\mathsf{id}^* \in \mathcal{I}^*$, and $\mathsf{ibsk}_{\mathsf{id}} = \mathsf{IBExtract}_\lambda(\mathsf{ibmsk}, \mathsf{id})$ for any $\mathsf{id} \in \mathcal{ID}_\lambda$.

$(\mathsf{m_a}, \epsilon, \theta)$-*security in the BSM* is satisfied if for any function $f : \{0,1\}^n \to \{0,1\}^{\mathsf{m_a}}$ and any list of $\mathsf{n_p}$ distinct identities $\mathcal{I}^* \subseteq \mathcal{ID}_\lambda$, we have

$$\exists \mathsf{E} : \ \Pr[\mathsf{E}] \geq 1 - \epsilon \text{ and } I(\mathsf{ibshk}; (f(\mathsf{urs}), (\mathsf{ibsk}_{\mathsf{id}})_{\mathsf{id} \in \mathcal{ID}_\lambda \setminus \mathcal{I}^*}) \mid \mathsf{E}) \leq \theta,$$

where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $\mathsf{ibsk}_{\mathsf{id}} = \mathsf{IBExtract}_\lambda(\mathsf{ibmsk}, \mathsf{id})$ for $\mathsf{ibmsk} \xleftarrow{\$} \mathsf{IBSetup}_\lambda(\mathsf{urs})$, and $\mathsf{ibshk} = \mathsf{IBShare}_\lambda(\mathsf{ibsk}_{\mathsf{id}}, \mathcal{I}^*)$ for some $\mathsf{id} \in \mathcal{I}^*$.

We now define two types of sets

$$\mathcal{S}_{\hat{\mathsf{x}}, \ell_{\mathsf{inp}}} = \{\mathsf{x} = (\mathsf{x}_i)_{i \in [\ell_{\mathsf{inp}}]} \in \{0,1\}^{\ell_{\mathsf{inp}}} \mid \forall i : \mathsf{x}_i = \hat{\mathsf{x}}_i \vee \hat{\mathsf{x}}_i = \bot\}$$

for any $\hat{\mathsf{x}} = (\hat{\mathsf{x}}_i)_{i \in [\ell_{\mathsf{inp}}]} \in (\{0,1,\bot\})^{\ell_{\mathsf{inp}}}$, and

$$\mathcal{T}_{\mathsf{x}, \ell_{\mathsf{inp}}} = \{\hat{\mathsf{x}} \in \{0,1,\bot\}^{\ell_{\mathsf{inp}}} \mid \mathsf{x} \notin \mathcal{S}_{\hat{\mathsf{x}}, \ell_{\mathsf{inp}}}\}$$

for any $\mathsf{x} \in \{0,1\}^{\ell_{\mathsf{inp}}}$, and give the definition of BPRF and its restricted version, both of which can be converted into multi-party IB-NIKEs, as shown below. Here we note that BPRF in the fine-grained setting is of independent interest, since it also generically implies broadcast encryption with constant users in the fine-grained setting, using the technique in [10].

Definition 6.2 (Bit-fixing pseudorandom functions). A $\mathcal{C}_1$-ℓ_{inp}-*bit-fixing pseudorandom function (BPRF)* consists of four algorithm families $\mathsf{BPRF} = \{\mathsf{Setup}_\lambda, \mathsf{Eval}_\lambda, \mathsf{Constrain}_\lambda, \mathsf{CEval}_\lambda\} \in \mathcal{C}_1$ such that

- Setup_λ on input $\mathsf{urs} \in \{0,1\}^n$ for some $n = n(\lambda)$ outputs a master secret key msk.
- Eval_λ is a deterministic algorithm that on input a master secret key msk and a preimage $\mathsf{x} \in \{0,1\}^{\ell_{\mathsf{inp}}}$ outputs $\mathsf{y} \in \mathcal{Y} \cup \{\bot\}$, where $\bot \notin \mathcal{Y}$ denotes abort.
- $\mathsf{Constrain}_\lambda$ is a deterministic algorithm that on input a master secret key msk and a string $\hat{\mathsf{x}}$ outputs a constrained key $\mathsf{bfk}_{\hat{\mathsf{x}}} \in \mathcal{CK}_\lambda$, where $\mathcal{CK}_\lambda$ is the constrained key space.
- CEval_λ is a deterministic algorithm that on input a constrained key $\mathsf{bfk}_{\hat{\mathsf{x}}}$ and a preimage $\mathsf{x} \in \mathcal{S}_{\hat{\mathsf{x}}, \ell_{\mathsf{inp}}}$ outputs $\mathsf{y} \in \mathcal{Y} \cup \{\bot\}$.

ϵ-*correctness* is satisfied if for all $\hat{\mathsf{x}} \in (\{0,1,\bot\})^{\ell_{\mathsf{inp}}}$ and all x such that $\mathsf{x} \in \mathcal{S}_{\hat{\mathsf{x}}, \ell_{\mathsf{inp}}}$, we have

$$\Pr[\mathsf{CEval}_\lambda(\mathsf{bfk}_{\hat{\mathsf{x}}}, \mathsf{x}) = \mathsf{Eval}_\lambda(\mathsf{msk}, \mathsf{x})] \geq 1 - \epsilon,$$

where $\mathsf{bfk}_{\hat{\mathsf{x}}} = \mathsf{Constrain}_\lambda(\mathsf{msk}, \hat{\mathsf{x}})$ for $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$ and $\mathsf{msk} \xleftarrow{\$} \mathsf{Setup}_\lambda(\mathsf{urs})$.

$\mathcal{C}_2$-ϵ-*security* is satisfied if for any adversary $\mathcal{A} = \{\mathsf{adv}_\lambda\} \in \mathcal{C}_2$, all sufficiently large $\lambda \in \mathbb{N}$, and any $\mathsf{x} \in \{0,1\}^{\ell_{\mathsf{inp}}}$, we have

$$|\Pr[1 \xleftarrow{\$} \mathsf{adv}_\lambda(\mathsf{urs}, \mathsf{y}, \{\mathsf{bfk}_{\hat{\mathsf{x}}}\}_{\hat{\mathsf{x}} \in \mathcal{T}_{\mathsf{x}, \ell_{\mathsf{inp}}}})] - \Pr[1 \xleftarrow{\$} \mathsf{adv}_\lambda(\mathsf{urs}, \mathsf{y}', \{\mathsf{bfk}_{\hat{\mathsf{x}}}\}_{\hat{\mathsf{x}} \in \mathcal{T}_{\mathsf{x}, \ell_{\mathsf{inp}}}})]| \leq \epsilon,$$

where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $\mathsf{msk} \xleftarrow{\$} \mathsf{Setup}_\lambda(\mathsf{urs})$, $\mathsf{y} \xleftarrow{\$} \mathsf{Eval}_\lambda(\mathsf{msk}, \mathsf{x})$, $\mathsf{y}' \xleftarrow{\$} \mathcal{Y}$, and $\mathsf{bfk}_{\hat{\mathsf{x}}} = \mathsf{Constrain}_\lambda(\mathsf{msk}, \hat{\mathsf{x}})$ for all $\hat{\mathsf{x}} \in \mathcal{T}_{\mathsf{x}, \ell_{\mathsf{inp}}}$.

$(\mathsf{m_a}, \epsilon, \theta)$-*security* in the BSM is satisfied if for any function $f : \{0,1\}^n \to \{0,1\}^{\mathsf{m_a}}$ and any $\mathsf{x} \in \{0,1\}^{\ell_{\mathsf{inp}}}$, we have

$$\exists \mathsf{E} : \ \Pr[\mathsf{E}] \geq 1 - \epsilon \text{ and } I(\mathsf{y}; (f(\mathsf{urs}), \{\mathsf{bfk}_{\hat{\mathsf{x}}}\}_{\hat{\mathsf{x}} \in \mathcal{T}_{\mathsf{x}, \ell_{\mathsf{inp}}}}) \mid \mathsf{E}) \leq \theta,$$

where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $\mathsf{msk} \xleftarrow{\$} \mathsf{Setup}_\lambda(\mathsf{urs})$, $\mathsf{bfk}_{\hat{\mathsf{x}}} = \mathsf{Constrain}_\lambda(\mathsf{msk}, \hat{\mathsf{x}})$ for all $\hat{\mathsf{x}} \in \mathcal{T}_{\mathsf{x}, \ell_{\mathsf{inp}}}$, and $\mathsf{y} = \mathsf{Eval}_\lambda(\mathsf{msk}, \mathsf{x})$.

Definition 6.3 (Restricted bit-fixing pseudorandom functions). A $\mathcal{C}_1$-ℓ_{inp}*-restricted bit-fixing pseudorandom function (restricted BPRF)* is defined in exactly the same way as $\mathcal{C}_1$-ℓ_{inp}-BPRF except that constrained keys are only required for constraints in $\mathcal{R}_{\ell_{\mathsf{inp}}}$, where $\mathcal{R}_{\ell_{\mathsf{inp}}} = \bigcup_{1\le \mathsf{t}_1\le \mathsf{t}_2\le \ell_{\mathsf{inp}}} \{\bot\}^{\mathsf{t}_1-1}\times\{0,1\}^{\mathsf{t}_2-\mathsf{t}_1+1}\times\{\bot\}^{\ell_{\mathsf{inp}}-\mathsf{t}_2}$. Accordingly, in the correctness definition, $\hat{\mathsf{x}}$ ranges only over $\mathcal{R}_{\ell_{\mathsf{inp}}}$ and in the security and BSM-security definitions, the set $\mathcal{T}_{\mathsf{x},\ell_{\mathsf{inp}}}$ is replaced by $\mathcal{T}_{\mathsf{x},\ell_{\mathsf{inp}}} \cap \mathcal{R}_{\ell_{\mathsf{inp}}}$.

6.2 Constructions of Multi-Party IB-NIKE

In this section, we give our constructions of multi-party IB-NIKE in the bounded-parallel-time model, bounded-time model, and BSM, respectively. These constructions are similar to those in [27], with the distinction that the underlying BPRFs can be restricted BPRFs and are built in the fine-grained setting.

Construction in the bounded-parallel-time model. Let $\mathsf{NCBPRF} = \{\mathsf{Setup}_\lambda, \mathsf{Eval}_\lambda, \mathsf{Constrain}_\lambda, \mathsf{CEval}_\lambda\}$ be an $\mathsf{AC}^0[2]$-$\mathsf{n_p}$-restricted BPRF with URS length n and image space $\mathcal{Y}$ for some constant $\mathsf{n_p}$, and some $n = n(\lambda)$. Let $\mathsf{m_p}$ be any constant such that $\mathsf{n_p}/\mathsf{m_p}$ is an integer. Our $\mathsf{NCIB\text{-}NIKE} = \{\mathsf{IBSetup}_\lambda, \mathsf{IBExtract}_\lambda, \mathsf{IBShare}_\lambda\}$ with identity space $\{0,1\}^{\mathsf{n_p}/\mathsf{m_p}}$ and URS length n is defined as in Figure 15.

Theorem 6.4 *If* NCBPRF *satisfies* ϵ_1*-correctness and* NC^1-ϵ_2*-security, then* NCIB-NIKE *in Figure 15 is an* $\mathsf{AC}^0[2]$-$\mathsf{m_p}$*-IB-NIKE with* $2\cdot\epsilon_1$*-correctness and* NC^1-$(\epsilon_2+\epsilon_1)$*-security.*

$\underline{\mathsf{IBSetup}_\lambda(\mathsf{urs}):}$

Return $\mathsf{ibmsk} \xleftarrow{\$} \mathsf{Setup}_\lambda(\mathsf{urs})$

$\underline{\mathsf{IBExtract}_\lambda(\mathsf{ibmsk},\mathsf{id}):}$

For $i \in [\mathsf{m_p}]$

  $\mathsf{bfk}_i = \mathsf{Constrain}_\lambda(\mathsf{ibmsk}, \hat{\mathsf{x}}_i)$ where $\hat{\mathsf{x}}_i = (\bot^{(i-1)\mathsf{n_p}/\mathsf{m_p}}, \mathsf{id}, \bot^{(\mathsf{m_p}-i)\mathsf{n_p}/\mathsf{m_p}})$

Return $\mathsf{ibsk}_{\mathsf{id}} = (\mathsf{id}, (\mathsf{bfk}_i)_{i\in[\mathsf{m_p}]})$

$\underline{\mathsf{IBShare}_\lambda(\mathsf{ibsk}_{\mathsf{id}}, \mathcal{I}):}$

Parse $\mathcal{I} = \{\mathsf{id}_i\}_{i\in[\mathsf{m_p}]}$ and $\mathsf{ibsk}_{\mathsf{id}} = (\mathsf{id}, (\mathsf{bfk}_i)_{i\in[\mathsf{m_p}]})$

Return $\mathsf{ibshk}_{\mathcal{I}} = \mathsf{CEval}_\lambda(\mathsf{bfk}_{i^*}, (\mathsf{id}_i)_{i\in[\mathsf{m_p}]})$ where $i^* \in [\mathsf{m_p}]$ s.t. $\mathsf{id}_{i^*} = \mathsf{id}$

Figure 15: Construction of $\mathsf{NCIB\text{-}NIKE} = \{\mathsf{IBSetup}_\lambda, \mathsf{IBExtract}_\lambda, \mathsf{IBShare}_\lambda\}$.

Proof. **Complexity.** First note that $\{\mathsf{IBSetup}_\lambda, \mathsf{IBExtract}_\lambda, \mathsf{IBShare}_\lambda\}$ are computable in $\mathsf{AC}^0[2]$ since they run $\mathsf{Constrain}_\lambda$ and CEval_λ for a constant number of times.

Correctness. For any list of $\mathsf{m_p}$ distinct identities $\mathcal{I}^* \subseteq \{0,1\}^{\mathsf{n_p}/\mathsf{m_p}}$ and any $i, j \in [\mathsf{m_p}]$, by the ϵ_1-correctness of NCBPRF, we have

$$\Pr[\mathsf{CEval}_\lambda(\mathsf{bfk}_i, \mathcal{I}^*) = \mathsf{Eval}_\lambda(\mathsf{ibmsk}, \mathcal{I}^*)] \ge 1-\epsilon_1$$

$$\text{and} \quad \Pr[\mathsf{CEval}_\lambda(\mathsf{bfk}_j, \mathcal{I}^*) = \mathsf{Eval}_\lambda(\mathsf{ibmsk}, \mathcal{I}^*)] \ge 1-\epsilon_1,$$

where $\mathsf{ibmsk} \xleftarrow{\$} \mathsf{Setup}_\lambda(\mathsf{urs})$ for $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$ and $\mathsf{bfk}_i = \mathsf{Constrain}_\lambda(\mathsf{ibmsk}, \hat{\mathsf{x}}_i)$ for $\hat{\mathsf{x}}_i \in \{0,1,\bot\}^{\mathsf{n_p}}$ and any $i \in [\mathsf{m_p}]$. Therefore by a union bound, we have

$$\Pr[\mathsf{CEval}_\lambda(\mathsf{bfk}_i, \mathcal{I}^*) = \mathsf{CEval}_\lambda(\mathsf{bfk}_j, \mathcal{I}^*)] \ge 1-2\epsilon_1,$$

$$\text{i.e.,} \quad \Pr[\mathsf{IBShare}_\lambda(\mathsf{ibsk}_{\mathsf{id}_i}, \mathcal{I}^*) = \mathsf{IBShare}_\lambda(\mathsf{ibsk}_{\mathsf{id}_j}, \mathcal{I}^*)] \ge 1-2\epsilon_1,$$

where $\mathsf{ibsk}_{\mathsf{id}_i} = \mathsf{IBExtract}_\lambda(\mathsf{ibmsk}, \mathsf{id}_i)$ for $\mathsf{ibmsk} \xleftarrow{\$} \mathsf{IBSetup}_\lambda(\mathsf{urs})$ and any $i \in [\mathsf{m_p}]$.

Security. Let λ be the security parameter and $\mathcal{A} = \{\mathsf{adv}_\lambda\} \in \mathsf{NC}^1$ be any adversary against the NC^1-security of NCIB-NIKE. For any $\mathcal{I}^* = (\mathsf{id}_1^*, \cdots, \mathsf{id}_{\mathsf{m_p}}^*)$ and set $\mathsf{x} = (\mathsf{id}_1^*, \cdots, \mathsf{id}_{\mathsf{m_p}}^*) \in \{0,1\}^{\mathsf{n_p}}$, we construct $\mathcal{B} = \{\mathsf{advb}_\lambda\} \in \mathsf{NC}^1$ against the security of NCBPRF as follows.

On receiving $(\mathsf{urs}, \mathsf{y}, \{\mathsf{bfk}_{\hat{\mathsf{x}}}\}_{\hat{\mathsf{x}} \in \mathcal{T}_{\mathsf{x},\mathsf{n_p}} \cap \mathcal{R}_{\mathsf{n_p}}})$ where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $\mathsf{msk} \xleftarrow{\$} \mathsf{Setup}_\lambda(\mathsf{urs})$, $\mathsf{bfk}_{\hat{\mathsf{x}}} = \mathsf{Constrain}_\lambda(\mathsf{msk}, \hat{\mathsf{x}})$ for all $\hat{\mathsf{x}} \in \mathcal{T}_{\mathsf{x},\mathsf{n_p}} \cap \mathcal{R}_{\mathsf{n_p}}$, and either $\mathsf{y} = \mathsf{Eval}_\lambda(\mathsf{msk}, \mathsf{x})$ or $\mathsf{y} \xleftarrow{\$} \mathcal{Y}$, advb_λ constructs, for every $\mathsf{id} \in \{0,1\}^{\mathsf{n_p}/\mathsf{m_p}} \backslash \mathcal{I}^*$, the secret key

$$\mathsf{ibsk}_{\mathsf{id}} = (\mathsf{id}, (\mathsf{bfk}_{\hat{\mathsf{x}}_{\mathsf{id},i}})_{i \in [\mathsf{m_p}]}),$$

where

$$\hat{\mathsf{x}}_{\mathsf{id},i} = (\bot^{(i-1)\mathsf{n_p}/\mathsf{m_p}}, \mathsf{id}, \bot^{(\mathsf{m_p}-i)\mathsf{n_p}/\mathsf{m_p}}).$$

Since $\mathsf{id} \notin \mathcal{I}^*$, each $\hat{\mathsf{x}}_{\mathsf{id},i}$ is incompatible with x, and hence $\hat{\mathsf{x}}_{\mathsf{id},i} \in \mathcal{T}_{\mathsf{x},\mathsf{n_p}}$. Then advb_λ returns urs, $\mathsf{ibshk} = \mathsf{y}$, and $(\mathsf{ibsk}_{\mathsf{id}})_{\mathsf{id} \in \{0,1\}^{\mathsf{n_p}/\mathsf{m_p}} \backslash \mathcal{I}^*}$ to adv_λ. On receiving $b \in \{0,1\}$ from adv_λ, advb_λ forwards b to its challenger.

First we note that all operations in advb_λ are in NC^1, since advb_λ only packages the constrained keys above and runs adv_λ. Let Good be the event that the challenge value computed by $\mathsf{IBShare}_\lambda$ is equal to $\mathsf{Eval}_\lambda(\mathsf{msk}, x)$. By the ϵ_1-correctness of the underlying BPRF, $\Pr[\mathsf{Good}] \geq 1 - \epsilon_1$. Conditioned on Good, the simulated view is identical to the real IB-NIKE security game when $\mathsf{y} = \mathsf{Eval}_\lambda(\mathsf{msk}, x)$, and it is identical to the random IB-NIKE security game when $\mathsf{y} \xleftarrow{\$} \mathcal{Y}$. Hence the distinguishing advantage is at most $\epsilon_2 + \epsilon_1$, completing the proof of Theorem 6.4. □

Construction in the bounded-time model and BSM. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the sets of all word-RAM families such that for each $\mathcal{F}_1 = \{f_\lambda^1\} \in \mathcal{C}_1$ and $\mathcal{F}_2 = \{f_\lambda^2\} \in \mathcal{C}_2$ and each $\lambda \in \mathbb{N}$, f_λ^1 runs in time $\tilde{O}(T_1(\lambda))$ and f_λ^2 runs in time $\tilde{O}(T_2(\lambda))$. Let $\mathsf{m_p}$ be any constant such that $\mathsf{n_p}/\mathsf{m_p}$ is an integer. Our construction in the bounded-time model TIB-NIKE with URS length n is defined in exactly the same way as NCIB-NIKE (see Figure 15) except that the underlying NCBPRF is replaced by a $\mathcal{C}_1$-$\mathsf{n_p}$-BPRF with URS length n and image length ℓ for some constant $\mathsf{n_p}$, some ℓ, and some $n = n(\lambda)$. Notice that for our instantiation of $\mathcal{C}_1$-$\mathsf{n_p}$-BPRF given later in Section 6.3, we have $n = 0$, i.e., the IB-NIKE is in the plain model.

Moreover, let $\mathcal{C}_1'$ be the class of streaming word-RAM families with memory bounds $\mathsf{m_u}$. Let $\mathsf{m_p}'$ be any constant such that $\mathsf{n_p}/\mathsf{m_p}'$ is an integer. Our construction in the BSM BSIB-NIKE with URS length n' is defined in exactly the same way as NCIB-NIKE (see Figure 15) except that the underlying NCBPRF is replaced by a $\mathcal{C}_1'$-$\mathsf{n_p}$-BPRF with URS length n' and image length ℓ' for some constant $\mathsf{n_p}$, some ℓ', and some $n' = n'(\lambda)$. We have the following theorems.

Theorem 6.5 *If* TBPRF *is a* $\mathcal{C}_1$-$\mathsf{n_p}$*-BPRF with* ϵ_1*-correctness and* $\mathcal{C}_2$-ϵ_2*-security, then* TIB-NIKE *is a* $\mathcal{C}_1$-$\mathsf{m_p}$*-IB-NIKE with* $2 \cdot \epsilon_1$*-correctness and* $\mathcal{C}_2$-$(\epsilon_1 + \epsilon_2)$*-security.*

Theorem 6.6 *If* BSBPRF *is a* $\mathcal{C}_1'$-$\mathsf{n_p}$*-BPRF with* 0*-correctness and* $(\mathsf{m_a}, \epsilon', \epsilon_2')$*-security in the BSM, then* BSIB-NIKE *is a* $\mathcal{C}_1'$-$\mathsf{m_p}'$*-IB-NIKE with* 0*-correctness and* $(\mathsf{m_a}, \epsilon', \epsilon_2')$*-security in the BSM.*

The proofs of Theorems 6.5 and 6.6 are very similar to the proof of Theorem 6.4 and thus we omit the details.

6.3 BPRF from Multi-Party NIKE with Extendability

In this section, we instantiate (restricted) BPRF based on our multi-party NIKE with (restricted) extendability, and then show that all our multi-party NIKE schemes satisfy extendability.

Restricted BPRF in the bounded-parallel-time model. Let $\mathsf{NCNIKE} = \{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\}$ be an $\mathsf{AC}^0[2]$-$\mathsf{n_p}$-NIKE with shared key space $\mathcal{K}_\lambda$ and URS length n for some constant $\mathsf{n_p}$, and some $n = n(\lambda)$ satisfying restricted ϵ_1-extendability with associated algorithm families $\{\mathsf{Combine}_\lambda, \mathsf{Extract}_\lambda\}$ and NC^1-ϵ_2-security. Our $\mathsf{NCBPRF} = \{\mathsf{Setup}_\lambda, \mathsf{Eval}_\lambda, \mathsf{Constrain}_\lambda, \mathsf{CEval}_\lambda\}$ is defined as in Figure 16.

Theorem 6.7 NCBPRF *is an* $\mathsf{AC}^0[2]$-$(\mathsf{n_p} - 1)$*-restricted BPRF with* ϵ_1*-correctness and* NC^1-$(\epsilon_2 + 2^{\mathsf{n_p}}\epsilon_1)$*-security.*

Proof. **Complexity.** First we note that the constructions are computable in $\mathsf{AC}^0[2]$ since they only run Gen_λ and Share_λ for a constant number of times.

Correctness. Let $\mathsf{t}_1, \mathsf{t}_2 \in [\mathsf{n_p} - 1]$ and set $\mathcal{V} = [\mathsf{t}_1, \mathsf{t}_2]$. Fix $\hat{\mathsf{x}} = (\hat{\mathsf{x}}_i)_{i \in [\mathsf{n_p}-1]} \in \{\bot\}^{\mathsf{t}_1 - 1} \times \{0,1\}^{\mathsf{t}_2 - \mathsf{t}_1 + 1} \times \{\bot\}^{\mathsf{n_p} - 1 - \mathsf{t}_2}$ and $\mathsf{x} \in \mathcal{S}_{\hat{\mathsf{x}},\mathsf{n_p}-1}$. Consider the experiment where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $\mathsf{msk} \xleftarrow{\$} \mathsf{Setup}_\lambda(\mathsf{urs})$, and $\mathsf{bfk}_{\hat{\mathsf{x}}} = \mathsf{Constrain}_\lambda(\mathsf{msk}, \hat{\mathsf{x}})$. Let E be the event guaranteed by the ϵ_1-restricted extendability of NCNIKE for the selected key pairs $((\mathsf{pk}_{i,\mathsf{x}_i}, \mathsf{sk}_{i,\mathsf{x}_i})_{i \in [\mathsf{n_p}-1]}, (\mathsf{pk}_{\mathsf{n_p}}, \mathsf{sk}_{\mathsf{n_p}}))$. For any $i' \in \mathcal{V}$, let

$$\mathsf{localkey} = \mathsf{Combine}_\lambda(\mathsf{sk}_{i',\mathsf{x}_{i'}}, \{\mathsf{pk}_{i,\mathsf{x}_i}\}_{i \in \mathcal{V} \backslash \{i'\}})$$

$\underline{\mathsf{Setup}_\lambda(\mathsf{urs})}$:

$(\mathsf{pk}_{\mathsf{n_p}}, \mathsf{sk}_{\mathsf{n_p}}) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$

$(\mathsf{pk}_{i,\beta}, \mathsf{sk}_{i,\beta}) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ where $i \in [\mathsf{n_p} - 1]$, $\beta \in \{0,1\}$

Return $\mathsf{msk} = ((\mathsf{pk}_{i,\beta}, \mathsf{sk}_{i,\beta})_{i\in[\mathsf{n_p}-1],\beta\in\{0,1\}}, (\mathsf{pk}_{\mathsf{n_p}}, \mathsf{sk}_{\mathsf{n_p}}))$

$\underline{\mathsf{Eval}_\lambda(\mathsf{msk}, \mathsf{x})}$:

Parse $\mathsf{msk} = ((\mathsf{pk}_{i,\beta}, \mathsf{sk}_{i,\beta})_{i\in[\mathsf{n_p}-1],\beta\in\{0,1\}}, (\mathsf{pk}_{\mathsf{n_p}}, \mathsf{sk}_{\mathsf{n_p}}))$ and $\mathsf{x} = (\mathsf{x}_i)_{i\in[\mathsf{n_p}-1]}$

Return $\mathsf{y} = \mathsf{Share}_\lambda(1, \mathsf{sk}_{1,\mathsf{x}_1}, ((\mathsf{pk}_{i,\mathsf{x}_i})_{i\in[\mathsf{n_p}-1]}, \mathsf{pk}_{\mathsf{n_p}}))$

$\underline{\mathsf{Constrain}_\lambda(\mathsf{msk}, \hat{\mathsf{x}})}$:

Parse $\mathsf{msk} = ((\mathsf{pk}_{i,\beta}, \mathsf{sk}_{i,\beta})_{i\in[\mathsf{n_p}-1],\beta\in\{0,1\}}, (\mathsf{pk}_{\mathsf{n_p}}, \mathsf{sk}_{\mathsf{n_p}}))$ and $\hat{\mathsf{x}} = (\hat{\mathsf{x}}_i)_{i\in[\mathsf{n_p}-1]}$

Let $\mathcal{V} = \{i \in [\mathsf{n_p} - 1] \mid \hat{\mathsf{x}}_i \neq \bot\}$

If $\mathcal{V} = \emptyset$, return $\mathsf{bfk}_{\hat{\mathsf{x}}} = \mathsf{msk}$

Otherwise, let $j \in \mathcal{V}$

$\mathsf{localkey} = \mathsf{Combine}_\lambda(\mathsf{sk}_{j,\hat{\mathsf{x}}_j}, \{\mathsf{pk}_{i,\hat{\mathsf{x}}_i}\}_{i\in\mathcal{V}\setminus\{j\}})$

Return $\mathsf{bfk}_{\hat{\mathsf{x}}} = (\mathsf{localkey}, \mathcal{V}, \{\mathsf{pk}_{i,\beta}\}_{i\in[\mathsf{n_p}-1]\setminus\mathcal{V},\beta\in\{0,1\}}, \mathsf{pk}_{\mathsf{n_p}})$

$\underline{\mathsf{CEval}_\lambda(\mathsf{bfk}_{\hat{\mathsf{x}}}, \mathsf{x})}$:

If $\mathsf{bfk}_{\hat{\mathsf{x}}}$ is a master secret key, return $\mathsf{Eval}_\lambda(\mathsf{bfk}_{\hat{\mathsf{x}}}, \mathsf{x})$

Parse $\mathsf{bfk}_{\hat{\mathsf{x}}} = (\mathsf{localkey}, \mathcal{V}, \{\mathsf{pk}_{i,\beta}\}_{i\in[\mathsf{n_p}-1]\setminus\mathcal{V},\beta\in\{0,1\}}, \mathsf{pk}_{\mathsf{n_p}})$ and $\mathsf{x} = (\mathsf{x}_i)_{i\in[\mathsf{n_p}-1]}$

$\mathsf{localkey}^* = \mathsf{Combine}_\lambda(\mathsf{localkey}, \{\{\mathsf{pk}_{i,\mathsf{x}_i}\}_{i\in[\mathsf{n_p}-1]\setminus\mathcal{V}}, \mathsf{pk}_{\mathsf{n_p}}\})$

Return $\mathsf{y} = \mathsf{Extract}_\lambda(\mathsf{localkey}^*, \mathsf{pk}_{\mathsf{n_p}})$

Figure 16: Construction of $\mathsf{NCBPRF} = \{\mathsf{Setup}_\lambda, \mathsf{Eval}_\lambda, \mathsf{Constrain}_\lambda, \mathsf{CEval}_\lambda\}$.

and

$$\mathsf{localkey}^* = \mathsf{Combine}_\lambda(\mathsf{localkey}, \{\{\mathsf{pk}_{i,\mathsf{x}_i}\}_{i\in[\mathsf{n_p}-1]\setminus\mathcal{V}}, \mathsf{pk}_{\mathsf{n_p}}\}).$$

Then $\Pr[\mathsf{E}] \geq 1 - \epsilon_1$, and conditioned on E,

$$\mathsf{localkey}^* = \mathsf{Combine}_\lambda(\mathsf{sk}_{1,\mathsf{x}_1}, \{\{\mathsf{pk}_{i,\mathsf{x}_i}\}_{1<i\leq\mathsf{n_p}-1}, \mathsf{pk}_{\mathsf{n_p}}\})$$

Thus,

$$\begin{aligned}
&\Pr[\mathsf{CEval}_\lambda(\mathsf{bfk}_{\hat{\mathsf{x}}}, \mathsf{x}) = \mathsf{Eval}_\lambda(\mathsf{msk}, \mathsf{x})] \\
\geq &\Pr[\mathsf{CEval}_\lambda(\mathsf{bfk}_{\hat{\mathsf{x}}}, \mathsf{x}) = \mathsf{Eval}_\lambda(\mathsf{msk}, \mathsf{x}) \mid \mathsf{E}] \Pr[\mathsf{E}] \\
= &\Pr[\mathsf{Extract}_\lambda(\mathsf{localkey}^*, \mathsf{pk}_{\mathsf{n_p}}) = \mathsf{Share}_\lambda(1, \mathsf{sk}_{1,\mathsf{x}_1}, ((\mathsf{pk}_{i,\mathsf{x}_i})_{i\in[\mathsf{n_p}-1]}, \mathsf{pk}_{\mathsf{n_p}})) \mid \mathsf{E}] \Pr[\mathsf{E}] \\
= &\Pr[\mathsf{E}] \geq 1 - \epsilon_1
\end{aligned}$$

Security. Let λ be the security parameter, and $\mathcal{A} = \{\mathsf{adv}_\lambda\} \in \mathsf{NC}^1$ be any adversary against the NC^1-security of NCBPRF. For any $\mathsf{x} \in \{0,1\}^{\mathsf{n_p}-1}$, we construct an adversary $\mathcal{B} = \{\mathsf{advb}_\lambda\} \in \mathsf{NC}^1$ against the security of NCNIKE as follows.

On receiving $(\mathsf{urs}, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}, \mathsf{key})$ where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for $i \in [\mathsf{n_p}]$, and $\mathsf{key} = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]})$ or $\mathsf{key} \xleftarrow{\$} \mathcal{K}_\lambda$, advb_λ first samples $(\mathsf{pk}'_i, \mathsf{sk}'_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for every $i \in [\mathsf{n_p} - 1]$. Then advb_λ sets $\mathsf{pk}_{i,\beta} = \begin{cases} \mathsf{pk}_i \text{ if } \mathsf{x}_i = \beta \\ \mathsf{pk}'_i \text{ if } \mathsf{x}_i \neq \beta \end{cases}$ and $\mathsf{bfk}_{\hat{\mathsf{x}}} = (\mathsf{localkey}, \mathcal{V}, ((\mathsf{pk}_{i,\beta})_{i\in[\mathsf{n_p}-1]\setminus\mathcal{V},\beta\in\{0,1\}}, \mathsf{pk}_{\mathsf{n_p}}))$ for any $\hat{\mathsf{x}} \in \mathcal{T}_{\mathsf{x},\mathsf{n_p}-1} \cap \mathcal{R}_{\mathsf{n_p}-1}$ where $\mathsf{localkey} = \mathsf{Combine}_\lambda(\mathsf{sk}_{i',\hat{\mathsf{x}}_{i'}}, \{\mathsf{pk}_{i,\hat{\mathsf{x}}_i}\}_{i\in\mathcal{V}\setminus\{i'\}})$ for some $i' \in \mathcal{V}$ such that $\mathsf{sk}_{i',\hat{\mathsf{x}}_{i'}}$ is sampled by advb_λ. Then advb_λ returns $\mathsf{y} = \mathsf{key}$ and $\{\mathsf{bfk}_{\hat{\mathsf{x}}}\}_{\hat{\mathsf{x}}\in\mathcal{T}_{\mathsf{x},\mathsf{n_p}-1}\cap\mathcal{R}_{\mathsf{n_p}-1}}$ to adv_λ. When adv_λ finally returns $b \in \{0,1\}$, advb_λ forwards b to its challenger.

First we note that all operations in advb_λ are in NC^1, since advb_λ runs Gen_λ for a constant number of times. Moreover, when $\mathsf{y} = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]})$ (respectively, $\mathsf{y} \xleftarrow{\$} \mathcal{K}_\lambda$) where $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for $i \in [\mathsf{n_p}]$ and $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, the view of adv_λ is identical to its view in the security game for NCBPRF when y is honestly generated (respectively, generated at random), except on an event of probability at most $2^{\mathsf{n_p}-1}\epsilon_1$, by a union bound over the $2^{\mathsf{n_p}-1}$ complete key combinations. Hence, the advantage of advb_λ in breaking the security of NCNIKE is at least the advantage of adv_λ in breaking the security of BPRF minus $2^{\mathsf{n_p}}\epsilon_1$, completing the proof of Theorem 6.7. □

BPRF in the bounded-time model and BSM. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ represent the sets of all word-RAM families such that for each $\mathcal{F}_1 = \{f^1_\lambda\} \in \mathcal{C}_1$ and $\mathcal{F}_2 = \{f^2_\lambda\} \in \mathcal{C}_2$ and each $\lambda \in \mathbb{N}$, f^1_λ runs in time $\tilde{O}(T_1(\lambda))$ and f^2_λ runs in time $\tilde{O}(T_2(\lambda))$. Our construction in the bounded-time model TBPRF is defined in exactly the same way as in Figure 16 except that the underlying NCNIKE is replaced by TNIKE^*, which is a $\mathcal{C}_1$-$\mathsf{n_p}$-NIKE with shared key space $\mathcal{K}_\lambda$ and URS length n satisfying ϵ_1-extendability and $\mathcal{C}_2$-ϵ_2-security for some constant $\mathsf{n_p}$, and some $n = n(\lambda)$. When the underlying TBPRF is instantiated by our NIKE TNIKE^* (see Figure 12), we have $n = 0$.

Moreover, let $\mathcal{C}'_1$ represent the set of streaming word-RAM families with memory bounds $\mathsf{m_u}$. Our construction in the BSM BSBPRF is defined in exactly the same way as in Figure 16 except that the underlying NCNIKE is replaced by BSMKE, which is a $\mathcal{C}'_1$-$\mathsf{n_p}$-NIKE with shared key space $\mathcal{K}'_\lambda$ and URS length n' satisfying 0-extendability, ϵ'_1-correctness with $\epsilon'_1 = 2/\lambda$, and $(\mathsf{m_a}, \epsilon', \epsilon'_2)$-security in the BSM for some constant $\mathsf{n_p}$, some ℓ', and some $n' = n'(\lambda)$. In the full BPRF case, the all-$\perp$ constraint is handled by returning the master secret key and evaluating directly; all nontrivial constraints are handled by the combining and extracting procedures in Figure 16.

Theorem 6.8 TBPRF *is a* $\mathcal{C}_1$-$(\mathsf{n_p}-1)$-*BPRF with* ϵ_1-*correctness and* $\mathcal{C}_2$-$(\epsilon_2 + 2^{\mathsf{n_p}}\epsilon_1)$-*security.*

Theorem 6.9 BSBPRF *is a* $\mathcal{C}'_1$-$(\mathsf{n_p}-1)$-*BPRF with* 0-*correctness and* $(\mathsf{m_a} - (\mathsf{n_p}-1)q, \epsilon', \epsilon'_2)$-*security, provided* $\mathsf{m_a} \geq (\mathsf{n_p}-1)q$.

The proofs of Theorem 6.8 and Theorem 6.9 are very similar to the proof of Theorem 6.7 and thus we omit the details. The additional $(\mathsf{n_p}-1)q$ bits account for the URS bits in the unused secret keys needed to compute the constrained keys.

6.4 Extendability of Our Multi-Party NIKEs

We now argue that our multi-party NIKE in the bounded-parallel-time model satisfies restricted extendability, and our multi-party NIKEs in the bounded-time model and bounded-storage model satisfy extendability.

We first recall that NCNIKE $= \{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\}$ (see Figure 5) satisfies 0-correctness, $\mathsf{TNIKE}^* = \{\mathsf{Gen}^*_\lambda, \mathsf{Share}^*_\lambda\}$ (see Figure 12) satisfies $\tilde{O}((1-(q/\ell)^{\mathsf{n_p}-1})^q + 2\ell\mathsf{n_p}\lambda^k/R)$-correctness, and BSMKE $= \{\mathsf{Gen}'_\lambda, \mathsf{Share}'_\lambda\}$ (see Figure 14) satisfies $2/\lambda$-correctness. Then we have the following theorem.

Theorem 6.10 NCNIKE *(see Figure 5) satisfies* 0*-restricted extendability,* $\mathsf{TNIKE}^* = \{\mathsf{Gen}^*_\lambda, \mathsf{Share}^*_\lambda\}$ *(see Figure 12) satisfies* $\tilde{O}(\ell\mathsf{n_p}\lambda^k/R)$*-extendability, and* BSMKE *(see Figure 14) satisfies* 0*-extendability.*

We define the combining and extracting algorithm families $\{\mathsf{NCCombine}_\lambda, \mathsf{NCExtract}_\lambda\}$, $\{\mathsf{TCombine}_\lambda, \mathsf{TExtract}_\lambda\}$, $\{\mathsf{BSMCombine}_\lambda, \mathsf{BSMExtract}_\lambda\}$ for NCNIKE, TNIKE^*, and BSMKE respectively as in Figure 17. For NCNIKE, the required equalities follow from the matrix identities in the correctness proof of Theorem 3.8. These identities show that the same shared matrix is obtained no matter which party starts the computation. Since restricted extendability only asks for this property on consecutive sets of parties, NCNIKE satisfies 0-restricted extendability. For TNIKE^*, consider the NIKE with full security obtained by combining Theorems 4.8, 4.15 and 4.17. If no instance generated by $\mathsf{Generate}_\lambda(R, k, 0)$ has a solution, then $\mathsf{TCombine}_\lambda$ only takes intersections of the index sets with solutions, and $\mathsf{TExtract}_\lambda$ gives the same output as Share^*_λ. Thus extendability can fail only if some instance generated by $\mathsf{Generate}_\lambda(R, k, 0)$ has a solution. By the union bound, this happens with probability at most $\tilde{O}(\ell n_p \lambda^k/R)$ for the full security scheme. If the intersection is empty, both $\mathsf{TExtract}_\lambda$ and Share^*_λ output $\perp$, so this case does not violate extendability. For BSMKE, let local key be (D, urs_S) defined as in Figure 17. The algorithm $\mathsf{BSMCombine}_\lambda$ updates this pair in the same way as Share'_λ when more public keys are added, and $\mathsf{BSMExtract}_\lambda$ applies the same last step as Share'_λ. If there are too few common indices, both algorithms output $\perp$. Hence BSMKE satisfies 0-extendability.

$\underline{\mathsf{NCCombine}_\lambda(\mathsf{localkey}, \{\mathsf{pk}_i\}_{i\in\mathcal{V}}):}$

Parse $\mathsf{localkey} = \mathbf{R}$, $\{\mathsf{pk}_i\}_{i\in\mathcal{V}} = \{\{P_j\}_{j\in[\mathsf{t}_1,\mathsf{s}_1-1]} \cup \{P_j\}_{j\in[\mathsf{s}_2+1,\mathsf{t}_2]}\}$

Return $\mathsf{localkey}' = (\prod_{j=\mathsf{t}_1}^{\mathsf{s}_1-1} \mathbf{P}_j^\top)\mathbf{R}(\prod_{j=\mathsf{s}_2+1}^{\mathsf{t}_2} \mathbf{P}_j)$

$\underline{\mathsf{NCExtract}_\lambda(\mathsf{localkey}, \mathsf{pk}_{\mathsf{n_p}}):}$

Return key, which is the bottom-right bit of $\mathsf{localkey}$

$\underline{\mathsf{TCombine}_\lambda(\mathsf{localkey}, \{\mathsf{pk}_j\}_{j\in\mathcal{V}}):}$

Parse $\mathsf{localkey} = (\mathcal{S}^a)_{a\in[t]}$ and $\{\mathsf{pk}_j = (((I^s_{j,a})_{s\in[\ell]})_{a\in[t]}, \mathbf{v}_j)\}_{j\in\mathcal{V}}$

For $a \in [t]$, $j \in \mathcal{V}$, $\mathcal{T} = \emptyset$
  For every $s \in \mathcal{S}^a$, check by brute forcing whether $I^s_{j,a}$ has a solution
    If the check passes, $\mathcal{T} = \mathcal{T} \cup \{s\}$
  $\mathcal{S}^a = \mathcal{S}^a \cap \mathcal{T}$

Return $\mathsf{localkey} = (\mathcal{S}^a)_{a\in[t]}$

$\underline{\mathsf{TExtract}_\lambda(\mathsf{localkey}, \mathsf{pk}_{\mathsf{n_p}}):}$

Parse $\mathsf{localkey} = (\mathcal{S}^a)_{a\in[t]}$ and $\mathsf{pk}_{\mathsf{n_p}} = ((I^{j,s}_{\mathsf{n_p}})_{s\in[t],j\in[\ell]}, \mathbf{v}_{\mathsf{n_p}})$

If $|\mathcal{S}^a| < 1$ for some $a \in [t]$ abort

Else return $\mathsf{key} = ((\bigoplus_{s\in\mathcal{S}^1} s)|| \cdots ||(\bigoplus_{s\in\mathcal{S}^t} s)) \cdot \mathbf{v}_{\mathsf{n_p}}$

$\underline{\mathsf{BSMCombine}_\lambda(\mathsf{localkey}, \{\mathsf{pk}_j\}_{j\in\mathcal{V}}):}$

Parse $\mathsf{localkey} = (D, \mathsf{urs}_\mathcal{S})$

Parse $\{\mathsf{pk}_j = (h_j, g_j)\}_{j\in\mathcal{V}}$

Let $D' = D \cup \{h_j : j \in \mathcal{V}\}$

Let $\mathcal{S}' = \mathcal{S} \cap \bigcap_{j\in\mathcal{V}} \mathcal{S}_j$, where $\mathcal{S}_j = \{h_j(a) : a \in [q]\}$

Recompute $\mathcal{S}$ and $\mathcal{S}'$ from D and $(h_j)_{j\in\mathcal{V}}$ in a streaming manner, and remove from $\mathsf{urs}_\mathcal{S}$ all bits whose indices are not in $\mathcal{S}'$

Return $\mathsf{localkey}' = (D', \mathsf{urs}_{\mathcal{S}'})$

$\underline{\mathsf{BSMExtract}_\lambda(\mathsf{localkey}, \mathsf{pk}_{\mathsf{n_p}}):}$

Parse $\mathsf{localkey} = (D, \mathsf{urs}_\mathcal{S})$, recomputing $\mathcal{S}$ from D, and parse $\mathsf{pk}_{\mathsf{n_p}} = (h_{\mathsf{n_p}}, g_{\mathsf{n_p}})$

For each $j \in [q]$, let $u_j = h_{\mathsf{n_p}}(j)$ if $h_{\mathsf{n_p}}(j) \in \mathcal{S}$, and let $u_j = \perp$ otherwise

Let $\mathbf{s}$ be obtained from $(u_j)_{j\in[q]}$ by removing all $\perp$'s while preserving the order of j

Abort the whole protocol if $|\mathbf{s}| < \lambda$

Let $\mathbf{s}_\lambda$ be the first λ entries of $\mathbf{s}$

Return $\mathsf{key} = g_{\mathsf{n_p}}(\mathsf{urs}_{\mathbf{s}_\lambda})$

Figure 17: Constructions of $\{\mathsf{NCCombine}_\lambda, \mathsf{NCExtract}_\lambda\}$, $\{\mathsf{TCombine}_\lambda, \mathsf{TExtract}_\lambda\}$, and $\{\mathsf{BSMCombine}_\lambda, \mathsf{BSMExtract}_\lambda\}$. Recall that for any set $\mathcal{S}$, $\mathsf{urs}_\mathcal{S}$ denotes the sequence of all bits in urs with indices (ordered lexicographically) in $\mathcal{S}$; for any sequence $\mathbf{s}$, $\mathsf{urs}_\mathbf{s}$ follows the order of $\mathbf{s}$. Recall $t = \lceil d \log \lambda \rceil$ for some constant d.

Acknowledgements

Part of this work was supported by the National Natural Science Foundation of China under Grant Numbers 62472073 and 62272091, and by the National Cryptologic Science Fund of China under Grant Number 2026NCSF02059.

References

[1] Abboud, A., Williams, V.V., Weimann, O.: Consequences of faster alignment of sequences. In: Esparza, J., Fraigniaud, P., Husfeldt, T., Koutsoupias, E. (eds.) ICALP 2014, Part I. LNCS, vol. 8572, pp. 39–51. Springer, Berlin, Heidelberg (Jul 2014) (Cited on page 2.)

[2] Afshar, A., Couteau, G., Mahmoody, M., Sadeghi, E.: Fine-grained non-interactive key-exchange: Constructions and lower bounds. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part I. LNCS, vol. 14004, pp. 55–85. Springer, Cham (Apr 2023) (Cited on page 1, 3, 4, 7.)

[3] Alman, J., Huang, Y., Yeo, K.: Fine-grained complexity in a world without cryptography. In: Fehr, S., Fouque, P.A. (eds.) EUROCRYPT 2025, Part VII. LNCS, vol. 15607, pp. 375–405. Springer, Cham (May 2025) (Cited on page 2, 3.)

[4] Backurs, A., Tzamos, C.: Improving viterbi is hard: better runtimes imply faster clique algorithms. In: Proceedings of the 34th International Conference on Machine Learning - Volume 70. p. 311–321. ICML'17, JMLR.org (2017) (Cited on page 2.)

[5] Ball, M., Dachman-Soled, D., Kulkarni, M.: New techniques for zero-knowledge: Leveraging inefficient provers to reduce assumptions, interaction, and trust. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 674–703. Springer, Cham (Aug 2020) (Cited on page 2, 5.)

[6] Barrington, D.A.M.: Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. In: 18th ACM STOC. pp. 1–5. ACM Press (May 1986) (Cited on page 2.)

[7] Bauer, B., Couteau, G., Sadeghi, E.: Fine-grained non-interactive key exchange, revisited. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part II. LNCS, vol. 14921, pp. 286–312. Springer, Cham (Aug 2024) (Cited on page 2, 3, 4.)

[8] Bellare, M., Impagliazzo, R., Naor, M.: Does parallel repetition lower the error in computationally sound protocols? In: 38th FOCS. pp. 374–383. IEEE Computer Society Press (Oct 1997) (Cited on page 6, 22.)

[9] Biham, E., Goren, Y.J., Ishai, Y.: Basing weak public-key cryptography on strong one-way functions. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 55–72. Springer, Berlin, Heidelberg (Mar 2008) (Cited on page 2.)

[10] Boneh, D., Waters, B.: Constrained pseudorandom functions and their applications. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 280–300. Springer, Berlin, Heidelberg (Dec 2013) (Cited on page 7, 33.)

[11] Boneh, D., Zhandry, M.: Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 480–499. Springer, Berlin, Heidelberg (Aug 2014) (Cited on page 1.)

[12] Bringmann, K., Gawrychowski, P., Mozes, S., Weimann, O.: Tree edit distance cannot be computed in strongly subcubic time (unless APSP can). In: Czumaj, A. (ed.) 29th SODA. pp. 1190–1206. ACM-SIAM (Jan 2018) (Cited on page 2.)

[13] Brzuska, C., Couteau, G.: On building fine-grained one-way functions from strong average-case hardness. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 584–613. Springer, Cham (May / Jun 2022) (Cited on page 2.)

[14] Cachin, C., Maurer, U.M.: Unconditional security against memory-bounded adversaries. In: Kaliski, Jr., B.S. (ed.) CRYPTO'97. LNCS, vol. 1294, pp. 292–306. Springer, Berlin, Heidelberg (Aug 1997) (Cited on page 2, 3, 7, 9, 25, 26, 29.)

[15] Campanelli, M., Gennaro, R.: Fine-grained secure computation. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, Part II. LNCS, vol. 11240, pp. 66–97. Springer, Cham (Nov 2018) (Cited on page 2, 5.)

[16] Degwekar, A., Vaikuntanathan, V., Vasudevan, P.N.: Fine-grained cryptography. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part III. LNCS, vol. 9816, pp. 533–562. Springer, Berlin, Heidelberg (Aug 2016) (Cited on page 2, 5, 11.)

[17] Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. Inf. Theory 22(6), 644–654 (1976) (Cited on page 1.)

[18] Dodis, Y., Quach, W., Wichs, D.: Speak much, remember little: Cryptography in the bounded storage model, revisited. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part I. LNCS, vol. 14004, pp. 86–116. Springer, Cham (Apr 2023) (Cited on page 2, 9.)

[19] Dziembowski, S., Maurer, U.M.: On generating the initial key in the bounded-storage model. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 126–137. Springer, Berlin, Heidelberg (May 2004) (Cited on page 4, 7, 8, 29.)

[20] Egashira, S., Wang, Y., Tanaka, K.: Fine-grained cryptography revisited. Journal of Cryptology 34(3), 23 (Jul 2021) (Cited on page 2, 5, 11.)

[21] Freire, E.S.V., Hofheinz, D., Kiltz, E., Paterson, K.G.: Non-interactive key exchange. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 254–271. Springer, Berlin, Heidelberg (Feb / Mar 2013) (Cited on page 1.)

[22] Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 33–62. Springer, Cham (Aug 2018) (Cited on page 1.)

[23] Goldreich, O., Levin, L.A.: A hard-core predicate for all one-way functions. In: STOC. pp. 25–32. ACM (1989) (Cited on page 6.)

[24] Guo, S., Kamath, P., Rosen, A., Sotiraki, K.: Limits on the efficiency of (ring) LWE-based non-interactive key exchange. J. Cryptol. 35(1), 1 (2022) (Cited on page 1.)

[25] Håstad, J.: One-way permutations in NC0. Inf. Process. Lett. 26(3), 153–155 (1987) (Cited on page 2.)

[26] Hesse, J., Hofheinz, D., Kohl, L.: On tightly secure non-interactive key exchange. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 65–94. Springer, Cham (Aug 2018) (Cited on page 5.)

[27] Hofheinz, D.: Fully secure constrained pseudorandom functions using random oracles. Cryptology ePrint Archive, Paper 2014/372 (2014), https://eprint.iacr.org/2014/372 (Cited on page 4, 7, 34.)

[28] Joux, A.: A one-round protocol for tripartite Diffie-Hellman. J. Cryptol. 17(4), 263–276 (2004) (Cited on page 1, 4.)

[29] LaVigne, R., Lincoln, A., Williams, V.V.: Public-key cryptography in the fine-grained setting. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 605–635. Springer, Cham (Aug 2019) (Cited on page 2, 4, 6, 16, 17.)

[30] Lincoln, A., Williams, V.V., Williams, R.R.: Tight hardness for shortest cycles and paths in sparse graphs. In: Czumaj, A. (ed.) 29th SODA. pp. 1236–1252. ACM-SIAM (Jan 2018) (Cited on page 2.)

[31] Maurer, U.M.: Conditionally-perfect secrecy and a provably-secure randomized cipher. J. Cryptol. 5(1), 53–66 (1992) (Cited on page 2.)

[32] Maurer, U.M.: Secret key agreement by public discussion from common information. IEEE Trans. Inf. Theory 39(3), 733–742 (1993), https://doi.org/10.1109/18.256484 (Cited on page 8, 30.)

[33] Maurer, U.M.: Abstract models of computation in cryptography (invited paper). In: Smart, N.P. (ed.) 10th IMA International Conference on Cryptography and Coding. LNCS, vol. 3796, pp. 1–12. Springer, Berlin, Heidelberg (Dec 2005) (Cited on page 1.)

[34] Merkle, R.C.: Secure communications over insecure channels. Commun. ACM 21(4), 294–299 (1978) (Cited on page 2.)

[35] Razborov, A.A.: Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. Mathematical notes of the Academy of Sciences of the USSR 41(4) (Apr 1987) (Cited on page 2, 10.)

[36] Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) EUROCRYPT'97. LNCS, vol. 1233, pp. 256–266. Springer, Berlin, Heidelberg (May 1997) (Cited on page 1.)

[37] Smolensky, R.: Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In: Aho, A. (ed.) 19th ACM STOC. pp. 77–82. ACM Press (May 1987) (Cited on page 2, 10.)

[38] Wang, Y., Pan, J.: Non-interactive zero-knowledge proofs with fine-grained security. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 305–335. Springer, Cham (May / Jun 2022) (Cited on page 2, 5, 6, 11.)

[39] Wang, Y., Pan, J., Chen, Y.: Fine-grained secure attribute-based encryption. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part IV. LNCS, vol. 12828, pp. 179–207. Springer, Cham, Virtual Event (Aug 2021) (Cited on page 2, 5, 11.)

[40] Wang, Y., Su, C., Pan, J.: Fine-grained non-interactive key-exchange without idealized assumptions. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part II. LNCS, vol. 14921, pp. 251–285. Springer, Cham (Aug 2024) (Cited on page 1, 8.)

[41] Wang, Y., Su, C., Pan, J., Xu, C.: Fine-grained non-interactive key exchange without idealized assumptions, and lower bounds. Journal of Cryptology (2026), https://doi.org/10.1007/s00145-026-09589-2 (Cited on page 1.)

[42] Zuckerman, D.: Simulating BPP using a general weak random source. In: 32nd FOCS. pp. 79–89. IEEE Computer Society Press (Oct 1991) (Cited on page 25, 26.)