# Fine-Grained Non-Interactive Key-Exchange without Idealized Assumptions

Yuyu Wang [1](✉) Chuanjie Su [1] Jiaxin Pan [2]

[1] University of Electronic Science and Technology of China, Chengdu, China
wangyuyu@uestc.edu.cn, chuanjie.su@hotmail.com
[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 computation models, namely, the bounded parallel-time, bounded time, and bounded storage models. Their security is based on a very mild assumption (e.g., $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L}/\mathsf{poly}$) or even without any complexity assumption. 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 NIKE to multi-party identity-based NIKEs in the fine-grained settings.

**Keywords:** Multi-party non-interactive key exchange, fine-grained cryptography, complexity assumptions, identity-based cryptography.

## 1 Introduction

### 1.1 Background

Constructing non-interactive key exchange (NIKE) stands as one of the central focuses in cryptography. NIKE serves as a fundamental building block that enables 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 with 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 [14], Learning With Errors [20], and assumptions related to the factoring [17], and more exotic ones such as the existence of generic groups [31, 27] and algebraic groups [18]. The only known candidate of three-party NIKE is proposed by Joux based on the bilinear Diffie-Hellman assumption over pairings [24]. 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 [9]. This rises 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 $\mathsf{n_p}$-party NIKE schemes for any constant $\mathsf{n_p}$ in the 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 [13]*, 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 have security based on very mild assumptions or even no 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 the terms of running time (e.g., [28, 7]), parallel-running time (i.e., circuit depths) (e.g., [21, 13]), or storage (e.g., [29, 15]). 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 secure against adversaries running in time $o(\lambda^{\mathsf{n_p}})$ in the random oracle model (respectively, $o(\lambda^2)$ in the 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 multi-linear generic group model) when dealing with a large number of parties. This raises another question whether we can have multi-party NIKE even if we are in, say, the Pessiland, where one-way function does not exist.

An independent and concurrent recent work by Couteau et al. [6] 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 exponential secure injective pseudorandom generators (PRGs) and the sub-exponential hardness of computational Diffie-Hellman problem (CDH). Their results can also be extended to a 6-party NIKE under PRGs 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 PRGs 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 exponential secure injective pseudorandom generators.

## 1.2 Our Contributions

We revisit multi-party NIKE in the fine-grained settings and propose constructions in the bounded parallel-time, bounded time, and bounded storage models based on very mild fine-grained complexity assumptions or even no assumption. 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 $\mathsf{NC}^1$ (i.e., circuits with logarithmic-depth and polynomial-size). Clearly, the honest users consume less resources than adversaries, since $\mathsf{AC}^0[2] \subsetneq \mathsf{NC}^1$ [30, 32]. Similar to prior works in the same setting (e.g., [13, 12]), 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 $\mathsf{NC}^1$-fine-grained schemes are based on $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$ [13, 12, 4, 34, 16, 33]. According to Barrington, $\oplus\mathsf{L/poly}$ is the class of languages with polynomial-sized branching programs and $\mathsf{NC}^1$ languages have polynomial-sized branching programs of constant width [5]. 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 with running time $\tilde{O}(\lambda^{\mathsf{n_p}+k})$. The underlying assumption is the average-case Zero-$k$-Clique hypothesis for any constant $k > 2$, which is exactly the one used in [25] for constructing fine-grained two-party NIKE. It is also wildly used in the fine-grained complexity to derive conditional lower bounds for other problems (e.g., [1, 10, 26, 3]). Note that no known approach for this assumption is faster than essentially $\lambda^k$, and the worst-case Zero-$k$-Clique hypothesis for any constant $k$ does not appear to imply $\mathsf{P} \neq \mathsf{NP}$ [25].
- In the bounded storage model (BSM) [11], we extend the two-party NIKE in [11] 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 a crucial evaluation criterion.

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 uniformly random string (URS) model, where there exists a URS accessible

Multi-Party IB-NIKE $\overset{\text{Sec. A.2}}{\Longleftarrow}$ (Restricted) BPRF $\overset{\text{Sec. A.3}}{\Longleftarrow}$ Extendable Multi-Party NIKE

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

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, the URS model is strictly implied by the random oracle model, as one can use a hash function modeled as a random oracle to generate the URS.

**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 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 bounded storage model, making the number of users super-constant does not spoil the security proof since it is based on no assumption. 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 $\mathsf{P} \neq \mathsf{NP}$ [25].

Our transformation contains several intermediate steps, as described figuratively in Figure 1. Specifically, we follow the technique in [23] to show that in the fine-grained settings, 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 nice and natural property we call extendability, which is satisfied by all our constructions (and indeed also ones in previous works [24, 2]) except for ones in the 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. We refer the reader to Appendix A for the full details.

**Practical implications.** Our constructions do not only rely on weak assumptions and have low computational complexity, but also have the following practical implications.

Our constructions in the bounded (parallel-)time setting are well-suited for systems where attacks are only meaningful if they can succeed quickly. For instance, users can use the shared key to protect messages valuable for a short period and can be either published or deleted later. Another application is the generation of message authentication codes (MACs) using the shared key. It prevents adversaries from forging MACs within the time it takes for an honest user to compute the MAC. This ensures security by allowing the system to reject users who have timed out while attempting to generate MACs. More specifically, let $n = \log \lambda$. In the bounded parallel time model, the parallel running-time of each party is about $3n$, while any adversary with parallel running-time $O(n)$ cannot break it. Then we can let the parties re-share the session key with period, say $10n$, if we use it as the key of a MAC. Also, we can use it to transmit messages (e.g., verification codes) that can be published after a time of $10n$. In the bounded time model, assuming that the number of parties is 3, the running-time of each party can be about $\lambda^5 \log \lambda$, while any adversary with running-time $O(\lambda^6)$ cannot break it. The applications are similar to the above. The construction in BSM also finds direct applications in scenarios where a high-speed broadcast channel generates URSs that are too large for an adversary to store. Moreover, as mentioned in [13], combining fine-grained primitives with standard ones immediately yield hybrids secure against restricted adversaries under weak assumptions (or no assumption) and secure 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 $\mathsf{NC}^1$. Similar to previous works [13, 12, 4, 34, 16, 33], 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 [16] to construct an NIKE via the technique proposed in [22]. 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 a bunch of 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 $\mathsf{P}_1$, $\mathsf{P}_2$, and $\mathsf{P}_3$ 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 $\mathsf{P}_1$ and $\mathsf{P}_2$ with their secret matrices, while it is not computable by $\mathsf{P}_3$ 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 on $\mathbf{R}_2$, 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 $\mathsf{SymR}_\lambda$ is the uniform distribution over symmetric matrices in $\{0,1\}^{\lambda\times\lambda}$. The shared key is in the form of $\mathbf{R}_1\widetilde{\mathbf{M}}\mathbf{R}_2\cdots\widetilde{\mathbf{M}}\mathbf{R}_n$. 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 able 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 nice property of the above distribution is that the last column and row 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 the kernel of $\widetilde{\mathbf{M}} \in \mathsf{ZeroSamp}_\lambda$ 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 right-bottom entry of the shared matrix is uniformly random in the view of an $\mathsf{NC}^1$ adversary. By running the scheme in 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 [33] 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 "half-half" probability. 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 [25]. In the construction in [25], two parties exchange lists containing $\ell$ 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 searching among the $\sqrt{\ell}$ instances. The Zero-$k$-Clique hypothesis says that finding the correct solution among the $\ell$ 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 tricky manner. Each indicator, denoted by index $i$, equals 1 if the $i$th random index chosen

by the last party is also chosen by all other parties, and 0 otherwise. Without loss of generality, we assume that the indices chosen by the last party are genuinely independent, while those chosen by each of the other parties are independent but conditioned on that they are all distinct. The former guarantees the independence of these indicators, and the latter ensures that each indicator equals 1 with a high probability. By computing 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 $\ell$ 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 we can proceed the proof without security loss potentially caused by the generalized splitting algorithm. When employing the Goldreich-Levin extractor [19] for privacy amplification, letting each party compress the shared indices by summing them up, as mentioned earlier, ensures that the running time of the Goldreich-Levin reduction 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 the details and refer the reader to Section 4 for details.

**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 [11]. 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 intersection of multiple sets of pairwise independent indices are also pairwise independent. While this is a somewhat natural generalization, it has not been rigorously treated before, as far as we know. To ensure correctness, we need to guarantee that this intersection is sufficiently large so that all parties can share a session key of a certain size, which reflects one of the main bulk of our contribution in this part. Indeed, even in the two-party setting as in [11], only the expected size of the secret key was shown. We introduce a set of indicators in a similar way to our Zero-$k$-Clique based construction. The main difference is that we only prove that these indicators are pairwise independent rather than completely independent and provide a lower bound of their sum. This is done by simultaneously exploiting the independence between all sets and the pairwise independence of elements within each set, and applying a theorem implied by Chebyshev's Inequality and Markov's Inequality. Here notice that this argument holds only when the set of indices chosen by one of the parties are perfectly pairwise independent and may repeat. Fortunately, this only makes the sum smaller and hence does not affect our result.

We would like to highlight that in [2], 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, termed as 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 combining it with 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 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 [8], which constructs BPRFs using multilinear maps, we obtain a generic construction of BPRFs from multi-party NIKEs with extendability. Ultilizing existing generic techniques [8, 23], we subsequently achieve fine-grained multi-party IB-NIKEs from BPRFs.

# 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 the process of sampling an element $x$ from a set or distribution $\mathcal{S}$ uniformly at random. By $\mathsf{negl}$ 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 $i$th 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 $H(x|y)$ we denote the *conditional entropy* of $x$ conditioned on a random variable $y$ with alphabet $\mathcal{Y}$ defined as

$$H(x|y) = \sum_{\hat{y} \in \mathcal{Y}} H(x|y = \hat{y}).$$

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

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

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 as $\{f_\lambda\}_{\lambda\in\mathbb{N}}$. 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

- $\mathsf{Gen}_\lambda$ on input $\mathsf{urs} \in \{0,1\}^n$ for some $n = n(\lambda)$ outputs a public key $\mathsf{pk}$ and a secret key $\mathsf{sk}$.
- $\mathsf{Share}_\lambda$ 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 $\mathsf{sk}_i$ outputs $\mathsf{key} \in \mathcal{K}_\lambda$, where $i \in [\mathsf{n_p}]$ and $\mathcal{K}_\lambda$ is the session key space.

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

$$\Pr[\mathsf{Share}_\lambda(i, \mathsf{sk}_i, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}) = \mathsf{Share}_\lambda(j, \mathsf{sk}_j, (\mathsf{pk}_i)_{i\in[\mathsf{n_p}]})] \geq 1 - \epsilon$$

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

$\mathcal{C}_2$-$\epsilon$*-security* (respectively, $\mathcal{C}_2$-$\epsilon$*-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

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

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

where $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ 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 $\mathsf{key}' \xleftarrow{\$} \mathcal{K}_\lambda$.

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

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

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

**Remark on the BSM.** Here we note that the security in the BSM captures the situation where all parties have access to $\mathsf{urs}$ 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 $\mathsf{urs}$. Afterwards, $\mathsf{urs}$ 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 [11] rather the streaming BSM proposed by Dodis, Quach, and Wichs [15]. 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 termed as $\mathsf{Combine}_\lambda$, 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 $\epsilon$*-extendability* if there exists two deterministic algorithm families $\{\mathsf{Combine}_\lambda, \mathsf{Extract}_\lambda\} \in \mathcal{C}_1$ such that there exists an event $\mathsf{E}$ such that $\Pr[\mathsf{E}] \geq 1 - \epsilon$, and conditioned on $\mathsf{E}$, for any $\mathcal{V} \subseteq [\mathsf{n_p}]$, $\mathcal{V}' \subseteq \mathcal{V}$, $j \in \mathcal{V}$, and $i' \in \mathcal{V}'$, we have

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

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

where $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda$ for every $i \in [\mathsf{n_p}]$, $\mathsf{localkey}_{\mathcal{V}'} = \mathsf{Combine}_\lambda(\mathsf{sk}_{i'}, \{\mathsf{pk}_i\}_{i\in\mathcal{V}'\setminus\{i'\}})$ 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). *$\epsilon$-restricted extendability* is defined in exactly the same way as $\epsilon$-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 $\mathsf{t}_2 \geq \mathsf{s}_2 \geq \mathsf{s}_1 \geq \mathsf{t}_1$.

# 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 $\mathsf{NC}^1$ 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 $\mathsf{NC}^1$-Fine-Grained Setting

We recall and define sampling procedures in $\mathsf{NC}^1$, along with several lemmata.

**Function families.** We now recall the definitions of $\mathsf{NC}^1$ circuits, $\mathsf{AC}^0[2]$ circuits, and $\oplus\mathsf{L/poly}$ circuits. Note that $\mathsf{AC}^0[2] \subsetneq \mathsf{NC}^1$ [30, 32].

**Definition 3.1** ($\mathsf{NC}^1$). The class of (non-uniform) $\mathsf{NC}^1$ *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 $\lambda$, $f_\lambda$ can be computed by a (randomized) circuit of size $p(\lambda)$, depth $c\log(\lambda)$, and fan-in 2 using $\mathsf{AND}$, $\mathsf{OR}$, and $\mathsf{NOT}$ 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 $\lambda$, $f_\lambda$ can be computed by a (randomized) circuit of size $p(\lambda)$, depth $c$, and unbounded fan-in using $\mathsf{AND}$, $\mathsf{OR}$, $\mathsf{NOT}$, and $\mathsf{PARITY}$ 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 $\lambda$, there is a non-deterministic Turing machine $\mathcal{M}_\lambda$ such that for each input $x$ with length $\lambda$, $\mathcal{M}_\lambda(x)$ 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 [13].

$\underline{\mathsf{LSamp}_\lambda}$:
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 $\mathsf{LSamp}_\lambda$, $\mathsf{RSamp}_\lambda$, $\mathsf{ZeroSamp}_\lambda$, 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 [16]). *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 [13]). *If* $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$, *for any* $\{\mathsf{adv}_\lambda\} \in \mathsf{NC}^1$ *and all sufficiently large* $\lambda$, *we have*

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

Here, we follow [16, 34, 33] 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 [13] 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 $\lambda$, then our scheme, given later, satisfies infinitely-often security.

Next, we recall a sampling algorithm $\mathsf{RandSamp}_\lambda$ that is implicitly defined in [33] and presented in Figure 3. Additionally, an inverse algorithm $\mathsf{RandSamp}_\lambda^{-1}$ is introduced. Roughly, as proven in [33], 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 $\mathsf{NC}^1$ 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.

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

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

**Lemma 3.6** (Lemma 6 in [33]). *If* $\mathsf{NC}^1 \subsetneq \oplus\mathsf{L/poly}$, *for any* $\{\mathsf{adv}_\lambda\} \in \mathsf{NC}^1$, *we have*

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

*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 $\mathsf{SymR}_\lambda$ outputting uniformly random symmetric matrices, and then propose a core lemma upon which we will heavily rely later.

**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* $\lambda$, *we have:*

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

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

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

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 definitions of $\mathsf{SymR}_\lambda$.

*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 $\mathsf{NC}^1$, according to Lemma 3.5, we immediately have $|\Pr[\mathsf{adv}_\lambda(\widetilde{\mathbf{M}}) = 1] - \Pr[\mathsf{adv}_\lambda(\widetilde{\mathbf{M}}') = 1]| \le \mathsf{negl}(\lambda)$, i.e.,

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

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

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

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 $i \in [\mathsf{n_p}]$, 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:

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

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

**Construction.** We now give our construction of multi-party NIKE $\mathsf{NCNIKE}$ in the bounded parallel-time model in Figure 5.

| $\underline{\mathsf{Gen}_\lambda(\mathsf{urs} \in \{0,1\}^{\lambda(\lambda+1)/2})}$: | $\underline{\mathsf{Share}_\lambda(i, \mathsf{sk}_i, (\mathsf{pk}_j)_{j\in[\mathsf{n_p}]})}$: |
|---|---|
| $\mathbf{M} = \mathsf{RandSamp}_\lambda(\mathsf{urs})$ | 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 $\mathsf{key}$ be the bottom-right bit of $\mathbf{K}$ |
| Return $(\mathsf{pk}, \mathsf{sk})$ | return $\mathsf{key}$ |

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

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

*Proof.* **Correctness.** 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. Let $(\mathbf{P}_i = \mathbf{M}\mathbf{R}_i, \mathbf{R}_i)_{i\in[\mathsf{n_p}]}$ be $\mathsf{n_p}$ public/secret key pairs generated by $\mathsf{Gen}_\lambda$. 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 $i, i' \in [\mathsf{n_p}]$, we have

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

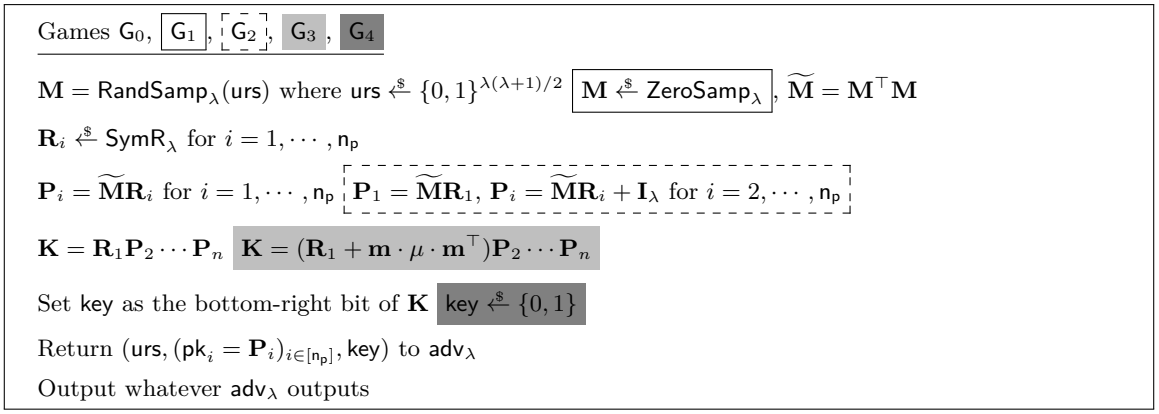


Figure 6: The challengers in $\mathsf{G}_0$, $\mathsf{G}_1$, $\mathsf{G}_2$, $\mathsf{G}_3$, and $\mathsf{G}_4$ for the proof of Theorem 3.8. $\mathbf{m}$ is the non-zero vector in $\mathrm{Ker}(\widetilde{\mathbf{M}})$.

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

**Security.** Let $\lambda$ 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.

$\mathsf{G}_0$**.** This is the original security game where $\mathsf{adv}_\lambda$ receives $\mathsf{urs} \xleftarrow{\$} \{0,1\}^{\lambda(\lambda+1)/2}$, $(\mathsf{pk}_i)_{i\in[k]}$ where $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for all $i \in [\mathsf{n_p}]$, and $\mathsf{key} = \mathsf{Share}_\lambda(1, \mathsf{sk}_1, (\mathsf{pk})_{i\in[\mathsf{n_p}]})$, and outputs a bit $\beta$.

$\mathsf{G}_1$**.** $\mathsf{G}_1$ and $\mathsf{G}_0$ 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 $\mathsf{G}_1$.

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

*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 $\mathbf{M} \xleftarrow{\$} \mathsf{ZeroSamp}_\lambda$, $\mathsf{advb}_\lambda^0$ sets $\mathsf{urs} = \mathsf{RandSamp}_\lambda^{-1}(\mathbf{M})$, samples $\mathbf{R}_1 \xleftarrow{\$} \mathsf{SymR}_\lambda$, sets $\mathbf{P}_1 = \widetilde{\mathbf{M}}\mathbf{R}_1$, computes $\mathbf{K} = \mathbf{R}_1 \prod_{i>1} \mathbf{P}_i$, and sets $\mathsf{key}$ as the bottom-right bit of $\mathbf{K}$. Next it sends $(\mathsf{urs}, (\mathbf{P}_i)_{i\in[\mathsf{n_p}]}, \mathsf{key})$ to $\mathsf{adv}_\lambda$. When $\mathsf{adv}_\lambda$ returns $\beta \in \{0,1\}$, $\mathsf{advb}_\lambda^0$ returns $\beta$ as well.

First we note that all operations in $\mathsf{advb}_\lambda^0$ are $\mathsf{NC}^1$, since $\mathsf{advb}_\lambda^0$ only samples random bits, computes multiplication of a constant number of matrices, and runs $\mathsf{adv}_\lambda$. 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 $\mathsf{adv}_\lambda$ is identical to its view in $\mathsf{G}_0$ (respectively, $\mathsf{G}_1$). Hence, the advantage of $\mathsf{advb}_\lambda^0$ 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 proof. □

$\mathsf{G}_2$**.** $\mathsf{G}_2$ and $\mathsf{G}_1$ only differ in the way that $\mathbf{P}_2, \cdots, \mathbf{P}_n$ 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 $i = 2, \cdots, n$ in $\mathsf{G}_2$.

**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 $\mathcal{B}^1 = \{\mathsf{advb}_\lambda^1\} \in \mathsf{NC}^1$ in the game described in Lemma 3.7 as follows.

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

First we note that all operations in $\mathsf{advb}_\lambda^1$ are $\mathsf{NC}^1$, since $\mathsf{advb}_\lambda^1$ only samples random bits, computes multiplication of a constant number of matrices, and runs $\mathsf{adv}_\lambda$. 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 $\mathsf{adv}_\lambda$ is identical to its view in $\mathsf{G}_1$ (respectively, $\mathsf{G}_2$). Hence, the advantage of $\mathsf{advb}_\lambda^1$ 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 proof. □

**$\mathsf{G}_3$.** $\mathsf{G}_3$ is exactly the same as $\mathsf{G}_2$ 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 $\mathrm{Ker}(\widetilde{\mathbf{M}})$, 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 $\mathsf{SymR}_\lambda$ 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 that the view of $\mathsf{adv}_\lambda$ in $\mathsf{G}_3$ is identical to its view in $\mathsf{G}_2$. □

**$\mathsf{G}_4$.** $\mathsf{G}_4$ is exactly the same as $\mathsf{G}_3$ except that the challenger sets the session key $\mathsf{key}$ 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 $\mathsf{G}_3$ and $\mathsf{G}_4$. Assuming that $\prod_{j=2}^{i} \mathbf{P}_j = \widetilde{\mathbf{M}}\mathbf{S}_i + \mathbf{I}_\lambda$ for some $2 \le i < \mathsf{n_p}$ 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 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 vector in $\mathbf{R}_1$. Let $\mathsf{key}$ be the bottom-right bit of $\mathbf{K}$ in $\mathsf{G}_2$. 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 $\mu$ is a random bit and $\mathsf{adv}_\lambda$ obtains no information on $\mu$ except for $\mathsf{key}$, $\mathsf{key}$ is a random bit in the view of $\mathsf{adv}_\lambda$ as well, completing this part of proof. □

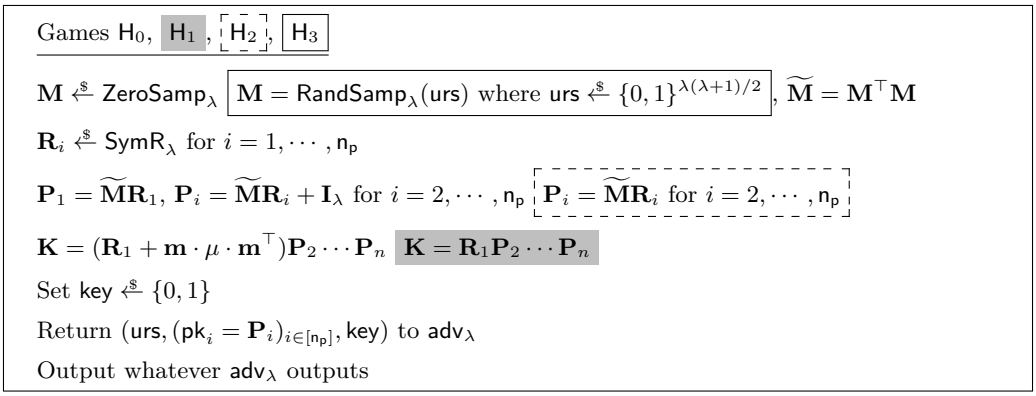


Figure 7: The challengers in $\mathsf{H}_0$, $\mathsf{H}_1$, $\mathsf{H}_2$, and $\mathsf{H}_3$ for the proof of Theorem 3.8. $\mathbf{m}$ is the non-zero vector in $\mathrm{Ker}(\widetilde{\mathbf{M}}^\top)$.

We now do all the previous steps in the reverse order as in Figure 7. Note that the view of the adversary in $\mathsf{H}_0$ (respectively, $\mathsf{H}_3$) is identical to its view in $\mathsf{G}_4$ (respectively, the honest game where $\mathsf{key}$ 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. □

**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. 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 Appendix A 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 $\tilde{O}(\lambda^{\mathsf{n_p}+k})$ 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 by $\mathsf{weight}(v, u)$ any two nodes $v$ and $u$ in a graph we denote the weight between $(v, u)$.

**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 $\lambda$ 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, there is an edge between two nodes $v \in \mathcal{P}_i$ and $u \in \mathcal{P}_j$ if and only if $i \neq j$. $(v_1, \cdots, v_k) \in \mathcal{P}_1 \times \cdots \times \mathcal{P}_k$ is said to be a solution of $I$ if for each $v_i, v_j \wedge i \neq j$, we have $\sum_{i\in[k]} \sum_{j\in[k]\wedge i\neq j} \mathsf{weight}(v_i, v_j) \equiv 0 \mod R$.

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 $\mathsf{Verify}$ with running time $O(\lambda^{k(1-\delta)})$ for some constant $\delta > 0$ such that $\mathsf{Verify}(I, \mathsf{w}) = 1$ if $I$ has a solution for an arbitrary witness $\mathsf{w}$ of an instance $I$ with one solution,
- for any algorithm $\mathcal{A}$ with running time $O(\lambda^{k(1-\delta)})$ for some constant $\delta > 0$, $\Pr[\mathsf{adv}_\lambda(I) = \mathsf{w}] \leq \frac{1}{100}$ where $I \xleftarrow{\$} \mathcal{D}_1[R]$ and $\mathsf{w}$ is an arbitrary witness of the instance $I$ with one solution.

**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 [25]).** *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*

$$\Pr[\mathsf{adv}_\lambda((I_j)_{j=1}^{\ell(\lambda)}) = i] \leq \epsilon,$$

*where $I_i \xleftarrow{\$} \mathcal{D}_1[R]$ for $i \xleftarrow{\$} [\ell(\lambda)]$, $I_j \xleftarrow{\$} \mathcal{D}_0[R]$ for all $j \in [\ell(\lambda)]\backslash\{i\}$, $\ell(\lambda) = \lambda^{\Omega(1)}$, $\epsilon \leq 1/200$, 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 of total variance distance at most $2\lambda^k/R$ from $\mathcal{D}_b[R]$ defined in [25] as in Figure 8.

**Theorem 4.4 (Theorem 14 in [25]).** *If the strong Zero-$k$-Clique-$R$ hypothesis holds, then we have $\Delta(\mathsf{Generate}_\lambda(R, k, b), \mathcal{D}_b[R]) \leq 2\lambda^k/R$, for all $b \in \{0, 1\}$, where $\mathsf{Generate}_\lambda$ runs in time $O(\lambda^2)$.*

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

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 $i \neq j$, all $v_i \in \mathcal{P}_i$, and all $v_j \in \mathcal{P}_j$

If $b = 0$, return $I$

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

Return $I$

Figure 8: Definition of $\mathsf{Generate}_\lambda$ [25].

The following corollary follows immediately from Theorem 4.4.

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

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

*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 any constant* $\mathsf{n_p}$.

We now recall a theorem given in [25]. 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 [25]).** *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

**Generalized splitting algorithm.** Before giving our construction, we propose generalized splitting algorithms $\{\mathsf{n_p}\text{-}\mathsf{Split}_\lambda\}$. Each splitting algorithm takes an instance $I$ and an integer $L = \log \mathsf{n_p}$ as input and generates $\binom{k}{2}^{\mathsf{n_p}}$ lists of instances in time $O(\lambda^{k(1-\delta)})$ for some constant $\delta > 0$, where the number of the solutions for each instance remains unchanged.

$\underline{\mathsf{n_p}\text{-}\mathsf{Split}_\lambda(I,L)}$:

Return $\mathsf{Split}_\lambda(I)$ if $L = 1$

Set $L = L - 1$ and run $(I_1^t, \cdots, I_{2^L}^t)_{t\in[m]} \xleftarrow{\$} \mathsf{n_p}\text{-}\mathsf{Split}_\lambda(I,L)$

For each $t \in [m]$ and each $i \in [2^L]$, run $(\hat{I}_{2i-1}^{j\times t}, \hat{I}_{2i}^{j\times t})_{j\in[\binom{k}{2}]} \xleftarrow{\$} \mathsf{Split}_\lambda(I_i^t)$

Return $(\hat{I}_1^j, \cdots, \hat{I}_{2^{L+1}}^j)_{j\in[m\cdot\binom{k}{2}]}$

Figure 9: The construction of $\mathsf{n_p}\text{-}\mathsf{Split}_\lambda$. By $m$ we denote the number of lists, which is initialized with $\binom{k}{2}$. 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, we have*

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

*for all* $i \in [\binom{k}{2}^{\mathsf{n_p}}]$ *where* $(\hat{I}_1^j, \cdots, \hat{I}_{\mathsf{n_p}}^j)_{j\in[m]} \xleftarrow{\$} \mathsf{n_p}\text{-}\mathsf{Split}_\lambda(I)$ *for* $I \xleftarrow{\$} \mathcal{D}_0[R]$*, and*

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

*for some* $i \in [\binom{k}{2}^{\mathsf{n_p}}]$ *where* $(\hat{I}_1^j, \cdots, \hat{I}_{\mathsf{n_p}}^j)_{j\in[m]} \xleftarrow{\$} \mathsf{n_p}\text{-}\mathsf{Split}_\lambda(I)$ *for* $I \xleftarrow{\$} \mathcal{D}_1[R]$*.*

*Proof.* We first note that since $\mathsf{n_p}\text{-}\mathsf{Split}_\lambda$ runs $\mathsf{Split}_\lambda$ for only a constant number of times, its running time is asymptoticly the same as $\mathsf{Split}_\lambda$, i.e., $O(\lambda^{k(1-\delta)})$. Let $I \xleftarrow{\$} \mathcal{D}_0[R]$. For all $j \in [\binom{k}{2}]$, we have

$$\Delta((\hat{I}_1^j, \hat{I}_2^j), (\mathcal{D}_0[\sqrt{R}] \times \mathcal{D}_0[\sqrt{R}])) \le \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt{R},$$

where $(\hat{I}_1^j, \hat{I}_2^j)_{j\in[\binom{k}{2}]} \xleftarrow{\$} \mathsf{n_p}\text{-}\mathsf{Split}_\lambda(I, 1)$, according to Theorem 4.6. For $L > 1$, let $I_i^t \xleftarrow{\$} \mathcal{D}_0[\sqrt[2^{L-1}]{R}]$ for each $t \in [m]$ and $i \in [2^L]$. By Theorem 4.6, we have

$$\Delta((\hat{I}_{2i-1}^{j\times t}, \hat{I}_{2i}^{j\times t}), (\mathcal{D}_0[\sqrt[2^L]{R}] \times \mathcal{D}_0[\sqrt[2^L]{R}])) \le \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[2^L]{R},$$

where $(\hat{I}_{2i-1}^{j\times t}, \hat{I}_{2i}^{j\times t})_{j\in[\binom{k}{2}]} \xleftarrow{\$} \mathsf{Split}_\lambda(I_i^t)$ for all $j \in [\binom{k}{2}]$. Since the sequence $I_1^t, \cdots, I_{2^L}^t$ are independent for each $t \in [m]$, and $\mathsf{n_p}\text{-}\mathsf{Split}_\lambda$ runs $\mathsf{Split}_\lambda$ for $2^{L-1}$ times, by a union bound, we have

$$\Delta((\hat{I}_1^j, \cdots, \hat{I}_{2^{L+1}}^j), \mathcal{D}_0[\sqrt[2^L]{R}]^{2^{L+1}}) \le 2^{L-1} \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[2^L]{R}$$

for each $j \in [m\binom{k}{2}]$. Therefore, for each $i \in \binom{k}{2}^{\mathsf{n_p}}$, we have

$$\begin{aligned}
\Delta((\hat{I}_1^i, \cdots, \hat{I}_{\mathsf{n_p}}^i), \mathcal{D}_0[\sqrt[\mathsf{n_p}]{R}]^{\mathsf{n_p}}) \le & \Delta((\hat{I}_1^i, \cdots, \hat{I}_{2^{\log \mathsf{n_p}}}^i), \mathcal{D}_0[\sqrt[2^{\log \mathsf{n_p}}]{R}]^{2^{\log \mathsf{n_p}}}) \\
\le & \sum_{i=0}^{2^{\log \mathsf{n_p} - 1}} 2^i \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[2^{i+1}]{R} \\
\le & (2\mathsf{n_p} - 1) \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[\mathsf{n_p}]{R}.
\end{aligned}$$

Let $I \xleftarrow{\$} \mathcal{D}_1[R]$. For some $j \in [\binom{k}{2}]$, we have

$$\Delta((\hat{I}_1^j, \hat{I}_2^j), (\mathcal{D}_1[\sqrt{R}] \times \mathcal{D}_1[\sqrt{R}])) \le \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt{R},$$

where $(\hat{I}_1^j, \hat{I}_2^j)_{j\in[\binom{k}{2}]} \xleftarrow{\$} \mathsf{n_p}\text{-}\mathsf{Split}_\lambda(I, 1)$, according to Theorem 4.6. For $L > 1$, let $I_i^t \xleftarrow{\$} \mathcal{D}_1[\sqrt[2^{L-1}]{R}]$ for all $t \in [m]$ and all $i \in [2^L]$. By Theorem 4.6, for some $j \in [\binom{k}{2}]$, we have

$$\Delta((\hat{I}_{2i-1}^{j\times t}, \hat{I}_{2i}^{j\times t}), (\mathcal{D}_1[\sqrt[2^L]{R}] \times \mathcal{D}_1[\sqrt[2^L]{R}])) \le \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[2^L]{R},$$

where $(\hat{I}_{2i-1}^{j\times t}, \hat{I}_{2i}^{j\times t})_{j\in[\binom{k}{2}]} \xleftarrow{\$} \mathsf{Split}_\lambda(I_i^t)$. Since all $I_1^t, \cdots, I_{2^L}^t$ are independent for all $t \in [m]$, and $\mathsf{n_p}\text{-}\mathsf{Split}_\lambda$ runs $\mathsf{Split}_\lambda$ $2^{L-1}$ times, by a union bound, we have

$$\Delta((\hat{I}_1^j, \cdots, \hat{I}_{2^{L+1}}^j), \mathcal{D}_1[\sqrt[2^L]{R}]^{2^{L+1}}) \le 2^{L-1} \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[2^L]{R}$$

for some $j \in [m\binom{k}{2}]$. Therefore, for some $i \in \binom{k}{2}^{\mathsf{n_p}}$, we have

$$\begin{aligned}
\Delta((\hat{I}_1^i, \cdots, \hat{I}_{\mathsf{n_p}}^i), \mathcal{D}_1[\sqrt[\mathsf{n_p}]{R}]^{\mathsf{n_p}}) \le & \Delta((\hat{I}_1^i, \cdots, \hat{I}_{2^{\log \mathsf{n_p}}}^i), \mathcal{D}_1[\sqrt[2^{\log \mathsf{n_p}}]{R}]^{2^{\log \mathsf{n_p}}}) \\
\le & \sum_{i=0}^{2^{\log \mathsf{n_p} - 1}} 2^i \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[2^{i+1}]{R} \\
\le & (2\mathsf{n_p} - 1) \binom{k}{2} 4^{\binom{k}{2}} 3\lambda^k / \sqrt[\mathsf{n_p}]{R}.
\end{aligned}$$

□

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

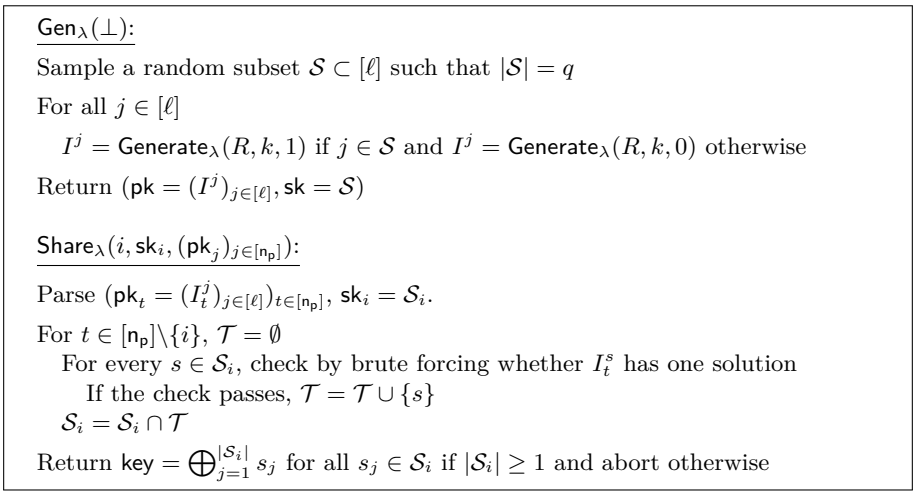


Figure 10: Construction of time $\mathsf{TNIKE} = \{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\}$.

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 $\lambda \in \mathbb{N}$, $f_\lambda$ runs in time $\tilde{O}(\ell\lambda^2 + q\lambda^k)$ (respectively, $\tilde{O}(\ell\lambda^k)$), where $\ell = \lambda^{\Omega(1)}$ and $q < \ell$. Let $\epsilon_1 = (1 - (\frac{q}{\ell})^{\mathsf{n_p}-1})^q + 2\ell\mathsf{n_p}\lambda^k/R$ and

$$\epsilon_2 \leq 201/200 + \ell(2\mathsf{n_p} - 1)\binom{k}{2}4^{\binom{k}{2}}3\lambda^k/R - (1 - \frac{q-1}{\ell})^{\mathsf{n_p}} + 2\mathsf{n_p}(q - 1 + \ell)\lambda^k/R.$$

We have the following theorem.

**Theorem 4.8** *If the strong Zero-k-Clique-R hypothesis holds, then for any constant* $\mathsf{n_p}$, $\mathsf{TNIKE}$ *is a* $\mathcal{C}_1$-$\mathsf{n_p}$*-NIKE with* $\epsilon_1$*-correctness and* $\mathcal{C}_2$-$\epsilon_2$*-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 $\ell$ times and run check for $q$ 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 $(1 - (\frac{q}{\ell})^{\mathsf{n_p}-1})^q$. Without loss of generality, we assume that the variables in $\mathcal{S}_{\mathsf{n_p}}$ are independently sampled from $[\ell]$ with replacement. Defining the indictor $Y_x$ such that $Y_x = \begin{cases} 1 & x \in \cap_{i=1}^{\mathsf{n_p}-1}\mathcal{S}_i \\ 0 & \text{otherwise} \end{cases}$. For any $x \xleftarrow{\$} [\ell]$ we have

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

i.e., $\Pr[Y_x = 0] = 1 - (q/\ell)^{\mathsf{n_p}-1}$. Moreover, for $x_1, \cdots, x_q \xleftarrow{\$} [\ell]$ for any $q$, we have

$$\begin{aligned}
\Pr[Y_{x_1} = 1 \wedge \cdots \wedge Y_{x_q} = 1] &= \Pr[x_1, \cdots, x_q \in \cap_{i=1}^{\mathsf{n_p}-1}\mathcal{S}_i] \\
&= \prod_{i=1}^{\mathsf{n_p}-1} \Pr[x_1, \ldots, x_q \in \mathcal{S}_i] \ (\because \text{independence of } \mathcal{S}_1, \cdots \mathcal{S}_{\mathsf{n_p}-1}) \\
&= \prod_{i=1}^{\mathsf{n_p}-1} (\Pr[x_1 \in \mathcal{S}_i] \cdots \Pr[x_q \in \mathcal{S}_i]) \ (\because \text{independence of } \{x_j\}_{j\in[q]}) \\
&= \Pr[x_1 \in \cap_{i=1}^{\mathsf{n_p}-1}\mathcal{S}_i] \cdots \Pr[x_q \in \cap_{i=1}^{\mathsf{n_p}-1}\mathcal{S}_i] = \Pr[Y_{x_1} = 1] \cdots \Pr[Y_{x_q} = 1].
\end{aligned}$$

Hence, $\{Y_{x_j}\}_{j\in[q]}$ are independent.

Since the intersection of $\mathcal{S}_1, \cdots, \mathcal{S}_{\mathsf{n_p}}$ only becomes larger when variables in $\mathcal{S}_{\mathsf{n_p}}$ are sampled with non-replacement, the probability that $\cap_{i=1}^{\mathsf{n_p}} \mathcal{S}_i = \emptyset$ (i.e., which is the probability that some user aborts) is no larger than

$$\Pr[\wedge_{i=1}^{q} Y_{x_i} = 0] = \prod_{j=1}^{q} \Pr[Y_{x_j} = 0] = (1 - (\frac{q}{\ell})^{\mathsf{n_p}-1})^q.$$

Moreover, the $\mathsf{n_p}$ parties agree on the same key without aborting occurs if and only if $|\cap_{i=1}^{\mathsf{n_p}} \mathcal{S}_i| > 0$ and no instance generated by $\mathsf{Generate}_\lambda(R, k, 0)$ has a solution, which happens with probability

$$\Pr[\wedge_{i=1}^{q} Y_{x_i} = 0] + \Delta((I_i^j)_{j\in[\ell],i\in[\mathsf{n_p}]}, (\mathcal{D}_{b_i^j}[R])_{j\in[\ell],i\in[\mathsf{n_p}]}),$$

where $I_i^j \xleftarrow{\$} \mathsf{Generate}_\lambda(R, k, b_i^j)$ and $b_i^j$ is the $i$'th index in $\mathcal{S}_j$. Since $\Pr[\wedge_{i=1}^{q} Y_{x_i} = 0] = (1 - (\frac{q}{\ell})^{\mathsf{n_p}-1})^q$, and

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

according to Corollary 4.5, the total error probability on correctness is $\epsilon_1 = (1 - (\frac{q}{\ell})^{\mathsf{n_p}-1})^q + 2\ell\mathsf{n_p}\lambda^k/R$.

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

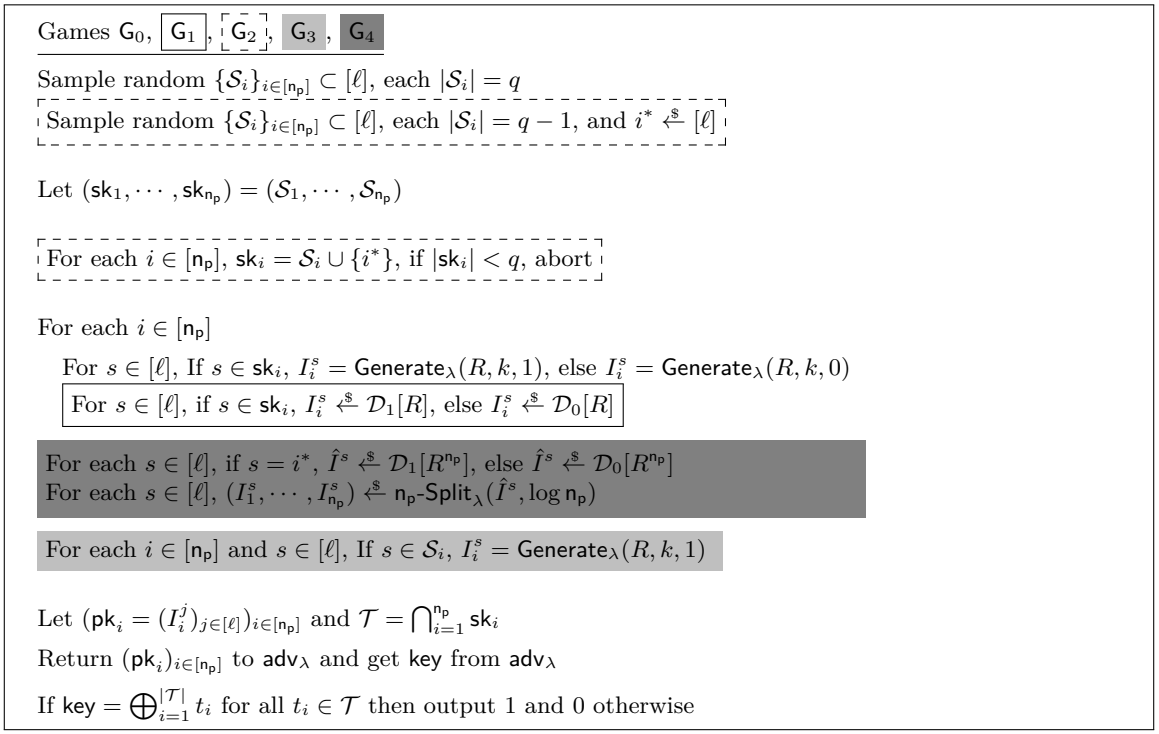


Figure 11: The challengers in $\mathsf{G}_0$, $\mathsf{G}_1$, $\mathsf{G}_2$, $\mathsf{G}_3$, and $\mathsf{G}_4$.

$\mathsf{G}_0$**.** This is the original security game where $\mathsf{adv}_\lambda$ receives $(\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}$ where $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\perp)$ for all $i \in [\mathsf{n_p}]$, and outputs a session key $\mathsf{key}$.

$\mathsf{G}_1$**.** $\mathsf{G}_1$ and $\mathsf{G}_0$ only differ in the way that $(I_i^j)_{i\in[\mathsf{n_p}],j\in[\ell]}$ are generated, namely, every instance $I_i^j$ in $\mathsf{pk}_i$ 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 $i \in [\mathsf{n_p}]$ and $j \in [\ell]$.

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

This lemma follows immediately from Corollary 4.5.

$\mathsf{G}_2$. $\mathsf{G}_2$ and $\mathsf{G}_1$ differ in the way that $(\mathsf{sk}_i)_{i\in[\mathsf{n_p}]}$ are generated, namely, for each $i \in [\mathsf{n_p}]$, $\mathsf{sk}_i = \{i^*\} \cup \mathcal{S}_i$ where $i^* \xleftarrow{\$} [\ell]$, $\mathcal{S}_i \subset [\ell]$, and $|\mathcal{S}_i| = q-1$, and 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** $|\Pr[\mathsf{G}_2^{\mathsf{adv}_\lambda} \Rightarrow 1] - \Pr[\mathsf{G}_1^{\mathsf{adv}_\lambda} \Rightarrow 1]| \leq 1 - (1-(q-1)/\ell)^{\mathsf{n_p}}$.

*Proof.* We define $\mathsf{Bad}$ as the event that $\mathsf{G}_2$ aborts. Note that $\mathsf{Bad}$ does not occur as long as we avoided the index with one solution when planting $q-1$ indices in each list. Let $s_i^j$ be the $j$th variable in $\mathcal{S}_i$, for each $i \in [\mathsf{n_p}]$ and $j \in [q-1]$, we have $\Pr[\mathsf{Bad}] = 1 - \prod_{i=1}^{\mathsf{n_p}} \Pr[s_i^1 \neq i^* \wedge \cdots \wedge s_i^{q-1} \neq i^*] \leq 1 - (1-(q-1)/\ell)^{\mathsf{n_p}}$.

Since the view of $\mathsf{adv}_\lambda$ is identical to its view in $\mathsf{G}_1$ when $\mathsf{Bad}$ does not occur in $\mathsf{G}_2$. Hence, we have

$$|\Pr[\mathsf{G}_2^{\mathsf{adv}_\lambda} \Rightarrow 1] - \Pr[\mathsf{G}_1^{\mathsf{adv}_\lambda} \Rightarrow 1]| \leq \Pr[\mathsf{Bad}] \leq 1 - (1-(q-1)/\ell)^{\mathsf{n_p}},$$

completing this part of proof. □

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

This lemma follows immediately from Corollary 4.5.

$\mathsf{G}_4$. In $\mathsf{G}_4$, $(I_i^j)_{i\in[\mathsf{n_p}]}$ are generated by $\mathsf{n_p}\text{-}\mathsf{Split}_\lambda(\hat{I}^j, \log \mathsf{n_p})$ for all $j$ instead.

Here, note that $\mathsf{n_p}\text{-}\mathsf{Split}_\lambda$ returns $m = \binom{k}{2}^{\mathsf{n_p}}$ instance lists every time, and we can check the correctness of each instance in the list by brute forcing. When $\ell$ is polynomial in $\lambda$, the time required to check by brute forcing a single instance is polynomially smaller than the time needed to solve the entire list instance. We will need to check $m$ of these instances by brute forcing, one for each of the $m$ produced $\mathsf{n_p}$ lists. When $\ell/m = \lambda^{-\Omega(1)}$, the total time required for all the checks by brute forcing is polynomially smaller than the time needed to solve a single list instance. Therefore, we only focus on the instance list which is verified by this check.

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

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

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

*Proof.* We construct an adversary $\mathcal{B} = \{\mathsf{advb}_\lambda\}$ in the game described in Theorem 4.3 as follows.

$\mathsf{advb}_\lambda$ runs in exactly the same way as the challengers in $\mathsf{G}_4$ except that $(\hat{I}^1, \cdots, \hat{I}^\ell)$ is generated by its own challenger as $\hat{I}^{i^*} \xleftarrow{\$} \mathcal{D}_1[R^{\mathsf{n_p}}]$ for $i^* \xleftarrow{\$} [\ell]$ and $\hat{I}^j \xleftarrow{\$} \mathcal{D}_0[R^{\mathsf{n_p}}]$ for all $j \in [\ell]\backslash\{i^*\}$. When $\mathsf{adv}_\lambda$ returns $\mathsf{key}$, $\mathsf{advb}_\lambda$ computes $\mathcal{T} = \bigcap_{i=1}^{\mathsf{n_p}} \mathcal{S}_i$ and returns $i^* = \mathsf{key} \oplus \bigoplus_{i=1}^{|\mathcal{T}|} t_i$.

Recall that $\mathsf{n_p}\text{-}\mathsf{Split}_\lambda$ runs in time time $\tilde{O}(\lambda^k)$. Since $\mathsf{advb}_\lambda$ runs $\mathsf{n_p}\text{-}\mathsf{Split}_\lambda$ for $\ell$ times, $\mathsf{Generate}_\lambda$ for $\mathsf{n_p} \cdot q$ times, and check for $\binom{k}{2}^{\mathsf{n_p}}$ times, its total running time is $\tilde{O}(\ell\lambda^k + \mathsf{n_p}q\lambda^2 + \binom{k}{2}^{\mathsf{n_p}}\lambda^k) = \tilde{O}(\ell\lambda^k)$.

Since the view of $\mathsf{adv}_\lambda$ is identical to its view in $\mathsf{G}_4$, the advantage of $\mathsf{advb}_\lambda$ is exactly $\Pr[\mathsf{G}_4^{\mathsf{adv}_\lambda} \Rightarrow 1]$, which is no more than $1/200$, according to Theorem 4.3. □

Putting all the above together, we have

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

where $\epsilon_{\mathsf{split}} = \ell(2\mathsf{n_p}-1)\binom{k}{2}4^{\binom{k}{2}}3\lambda^k/R$, 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

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

which approaches

$$e^{-\log^{2\mathsf{n_p}-1}\lambda} + O(1/\lambda^{k\mathsf{n_p}-k-\mathsf{n_p}}) = O(1/\lambda^{k\mathsf{n_p}-k-\mathsf{n_p}})$$

when $\lambda$ is sufficiently large, and

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

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

**Concrete examples.** We now give a concrete example of the parameters. Let $\ell = 2^{21}$, $\mathsf{n_p} = 3$, and $k = 3$, since $0 < 1-(1-\frac{q-1}{\ell})^{\mathsf{n_p}} - \frac{1}{\lambda^3} < 16/100$ and $\epsilon_{\mathsf{split}} + 2\mathsf{n_p}(q-1+\ell)\lambda^k/R < 1/200$, we have $\epsilon_1 < 2.87 \times 10^{-6}$ and $\epsilon_2 \leq \frac{1}{200} + \frac{16}{100} + \frac{1}{200} < \frac{1}{5}$.

**Full security.** While $\mathsf{TNIKE}$ only satisfy weak security, we can extend it to one with $\mathcal{C}_2$-$4 \cdot \epsilon_2$-security via privacy amplification by the Goldreich-Levin extractor [19]. Before giving our fully secure construction, we recall the Chebyshev bound as below.

**Theorem 4.14 (Chebyshev bound).** *For any $\epsilon > 0$, $b \in \{0,1\}$ and $q \in \mathbb{N}$, and let $\{x_i\}_{i\in[q]}$ be pairwise-independent, 0/1-random variables such that*

$$\Pr[x_i = b] \geq \frac{1}{2} + \epsilon$$

*for all $i \in [q]$. Let $x = \begin{cases} 1 & \text{if } \Sigma_{i=1}^q x_i \geq \frac{q}{2} \\ 0 & \text{if } \Sigma_{i=1}^q x_i < \frac{q}{2} \end{cases}$. Then*

$$\Pr[x \neq b] \leq \frac{1}{4\epsilon^2 q}.$$

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 $\lambda \in \mathbb{N}$, $f_\lambda$ 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 $\epsilon = 1/\lambda^{o(1)}$ and $\mathsf{TNIKE} = (\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda)$ be a $\mathcal{C}_1$-$\mathsf{n_p}$-NIKE that satisfies $\mathcal{C}_2$-$\frac{\epsilon}{4}$-weak security with the secret key size $\log(\lambda^{\Omega(1)})$, we construct $\mathsf{TNIKE}^* = (\mathsf{Gen}^*_\lambda, \mathsf{Share}^*_\lambda)$ satisfying $\mathcal{C}_2$-$\epsilon$-security as in Figure 12.

**Theorem 4.15** *If $\mathsf{TNIKE}$ is a $\mathcal{C}_1$-$\mathsf{n_p}$-NIKE with $\mathcal{C}_2$-$\frac{\epsilon}{4}$-weak security and $|\mathsf{sk}| = \log(\lambda^{\Omega(1)})$, then $\mathsf{TNIKE}^*$ is a $\mathcal{C}_1$-$\mathsf{n_p}$-NIKE with $\mathcal{C}_2$-$\epsilon$-security.*

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

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

*Proof.* Let $Q(\lambda) = |\mathsf{sk}|$, $m = 2Q(\lambda)/\epsilon$, and $\mathcal{A} = \{\mathsf{adv}_\lambda\} \in \mathcal{C}_2$ be any adversary breaking the $\mathcal{C}_2$-$\epsilon$-security of $\mathsf{TNIKE}^*$. We construct an adversary $\mathcal{B} = \{b_\lambda\} \in \mathcal{C}_2$ breaking the $\mathcal{C}_2$-$\frac{\epsilon}{4}$-weak security of $\mathsf{TNIKE}$.

Firstly, $b_\lambda$ chooses $\log(m)$ pairs $(\sigma_1, \mathbf{r}_1), \cdots, (\sigma_{\log(m)}, \mathbf{r}_{\log(m)}) \xleftarrow{\$} \{0,1\} \times \{0,1\}^{|\mathsf{sk}|}$. Then, for every nonempty subset $\mathcal{X} \subseteq [\log(m)]$, $b_\lambda$ computes $\mathbf{r}_\mathcal{X} = \oplus_{i\in\mathcal{X}} \mathbf{r}_i$ and $\sigma_\mathcal{X} = \oplus_{i\in\mathcal{X}} \sigma_i$. For every $i \in [Q(\lambda)]$, $b_\lambda$ runs

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

where $e^i$ denotes the $Q(\lambda)$-bit string with 1 in the $i$'th position and 0 everywhere else, and sets $\mathsf{key}^*_i$ as the majority of $\mathsf{key}_i^\mathcal{X}$. Finally, $b_\lambda$ outputs $\mathsf{key}^* = \mathsf{key}^*_1 || \cdots || \mathsf{key}^*_m$. Let

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

A quick calculation yields $|\mathcal{S}| > 2^{Q(\lambda)-1}\epsilon$. Since $\{\sigma_i\}_{i\in[\log(m)]}$ are sampled uniformly at random, we have

$$\Pr[(\sigma_i = \mathbf{r}_i \cdot \mathsf{key})_{i\in[\log(m)]}] = 1/m.$$

Notice that the set $\{\mathbf{r}_{\mathcal{X}}\}_{\mathcal{X}\subseteq[\log(m)]}$ are pairwise independent, the probability that the majority of bits is incorrect

$$\Pr[(\mathsf{key}_i^* \neq \mathsf{key}_i)\big|(\sigma_j = \mathsf{key}\cdot\mathbf{r}_j)_{j\in[\log(m)]} \wedge (\mathsf{key}\in S)] \leq \frac{4}{m\epsilon^2},$$

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

$$\begin{aligned}
\Pr[b_\lambda(\{\mathsf{pk}_i\}_{i\in[\mathsf{n_p}]}) = \mathsf{key}] \geq& \Pr[\mathsf{key}\in S]\cdot\Pr[b_\lambda(\{\mathsf{pk}_i\}_{i\in[\mathsf{n_p}]}) = \mathsf{key}\big|\mathsf{key}\in S]\\
=&\frac{\epsilon}{2}\Pr[\forall i\in[Q(\lambda)], (\mathsf{key}_i^* = \mathsf{key}_i)\big|\mathsf{key}\in S]\\
=&\frac{\epsilon}{2}(1-\Pr[\exists i\in[Q(\lambda)] \text{ s.t. } \mathsf{key}_i^*\neq\mathsf{key}_i\big|\mathsf{key}\in S])\\
\geq&\frac{\epsilon}{2}(1-\frac{Q(\lambda)}{m}\Pr[\mathsf{key}_i^*\neq\mathsf{key}_i\big|(\sigma_j=\mathsf{key}\cdot r_j)_{j\in[\log(m)]}\wedge\mathsf{key}\in S])\\
\geq&\frac{\epsilon}{2}(1-\frac{4Q(\lambda)}{(m\epsilon)^2}) = \frac{\epsilon}{2}-\frac{2Q(\lambda)}{m^2\epsilon}.
\end{aligned}$$

Assuming $Q(\lambda) > 2$, the probability that $b_\lambda$ succeeds is at least $\frac{\epsilon}{4}$. Since $\mathsf{adv}_\lambda$ runs in time $\tilde{O}(T_2(\lambda))$ and $m = \log\lambda^{\Omega(1)}\lambda^{o(1)}$, $\{b_\lambda\}$ runs in time $\tilde{O}(\log(\lambda^{\Omega(1)})\cdot m\cdot T(\lambda)) = \tilde{O}(T_2(\lambda))$, completing the proof of Theorem 4.15. □

**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 Appendix A 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 [11] to a multi-party one. 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 $\mathcal{Y}$ are 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 [35, 11]). A sequence of random variables $x_1,\cdots,x_q$ with alphabet $\mathcal{Y}$ are approximate 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 $\Pr[x_i=\hat{y}_1\wedge x_j=\hat{y}_2] = 1/|\mathcal{Y}|(|\mathcal{Y}|-1)$.

**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 [35, 11]). 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 "approximate 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 $g\xleftarrow{\$}\mathcal{H}$.

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

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** ([11, 35]). *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.*

In the rest part of this whole section, we say that a sequence of variables $(u_i)_{i\in\mathbb{Z}_q}$ is generated by $\mathcal{H}_n$ if it is generated as $u_i = h(x_i)$ for all $i \in \mathbb{Z}_q$ and some distinct $x_1, \cdots, x_n \in GF(n)$, where $h \xleftarrow{\$} \mathcal{H}_n$.

**Parameters.** Next we follow [11] 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 $q$, $\epsilon_1$, $\epsilon_2$, and $\theta$ satisfying the following equations:

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

where $H_b$ is the *binary entropy function* defined as $H_b(\rho) = -\rho\log\rho - (1-\rho)\log(1-\rho)$. As noted in [11], when $n >> \mathsf{m_a}$, we have $\delta = 0.9453$, $\rho = 1/3$, $\lambda \approx 3/\epsilon_2$, and $r \approx \lambda/6$.

Before giving our construction, we recall following two theorems. The first one is the main theorem in [11] for privacy amplification, and the second one is a direct combination from Chebyshev's Inequality and Markov's Inequality.

**Theorem 5.7 ([11]).** *Let $\lambda \in \mathbb{N}$ and $GF(n)$ be a finite field. Let $\mathcal{A}$ be any adversary with the memory bound $\mathsf{m_a} < n$, $\mathbf{s} = (s_i)_{i\in[\lambda]}$ be a sequence of approximately pairwise independent variables generated by $\mathcal{H}_n$, $\mathsf{urs}$, $\mathbf{z}$, and $g$ be random variables such that $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $\mathbf{z} = \mathcal{A}(\mathsf{urs})$, $g \xleftarrow{\$} \mathcal{G}_{\lambda,r}$, and $\mathsf{key} = g(\mathsf{urs_s})$. There exists a security event $\mathsf{E}$ such that*

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

**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 $q = n\sqrt[\mathsf{n_p}]{\frac{2\lambda}{n}}$. Our construction of multi-party NIKE in the BSM is defined as in Figure 14.

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

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

**Correctness.** To prove correctness, it is sufficient to prove that the probability that the protocol aborts is at most $\frac{2}{\lambda}$.

[1] The approximate pairwise independence and approximate 2-universal hash function family are defined in [35, 11] in an implicit way.

[2] The original theorem asserts that $I(\mathsf{key}; g|\mathbf{z}, \mathbf{s}, \mathsf{E}) \leq \theta$, 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 $\mathsf{sk}_i = \mathsf{urs}_{\mathcal{S}_i}$ where $\mathcal{S}_i$ is the set of all bits in $\mathsf{urs}$ with indices in $(h_i(j))_{j\in[q]}$

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

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

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

Abort the whole protocol if $|\mathsf{sk}_i| < \lambda$

Return $\mathsf{key} = g_{\mathsf{n_p}}(\mathsf{sk}_i)$

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

Without loss of generality, we assume that $h_{\mathsf{n_p}}$ is randomly sampled from the strongly 2-universal hash function family $\mathcal{F}_n$ instead (see Figure 13). Defining the indictor $Y_i$ as $Y_i = \begin{cases} 1 & x_i = h_{\mathsf{n_p}}(i) \in \cap_{l=1}^{\mathsf{n_p}-1}\mathcal{S}_l \\ 0 & \text{otherwise} \end{cases}$, we have

$$\Pr[Y_i = 1] = \Pr[x_i \in \cap_{l=1}^{\mathsf{n_p}-1}\mathcal{S}_l] = \prod_{l=1}^{\mathsf{n_p}-1} \Pr[x_i \in \mathcal{S}_l] = (\frac{q}{n})^{\mathsf{n_p}-1},$$

and for any $i \neq j$, we have

$$\begin{aligned}
\Pr[Y_i = 1 \wedge Y_j = 1] &= \Pr[x_i, x_j \in \cap_{l=1}^{\mathsf{n_p}-1}\mathcal{S}_l] \\
&= \prod_{l=1}^{\mathsf{n_p}-1} \Pr[x_i, x_j \in \mathcal{S}_l] \ (\because \text{independence of } \mathcal{S}_1, \cdots, \mathcal{S}_{\mathsf{n_p}-1}) \\
&= \prod_{l=1}^{\mathsf{n_p}-1} (\Pr[x_i \in \mathcal{S}_l] \cdot \Pr[x_j \in \mathcal{S}_l]) \ (\because \text{pairwise independence of } x_i \text{ and } x_j) \\
&= \Pr[x_i \in \cap_{l=1}^{\mathsf{n_p}-1}\mathcal{S}_l] \cdot \Pr[x_j \in \cap_{l=1}^{\mathsf{n_p}-1}\mathcal{S}_l] = \Pr[Y_i = 1] \cdot \Pr[Y_j = 1],
\end{aligned}$$

i.e., $Y_i$ and $Y_j$ are pairwise independent.

Moreover, we have

$$\mathbb{E}[Y_i] = \Pr[Y_i = 1] \cdot 1 + \Pr[Y_i = 0] \cdot 0 = \Pr[Y_i = 1] = (\frac{q}{n})^{\mathsf{n_p}-1},$$

$$\mathbb{E}[Y_i^2] = \Pr[Y_i^2 = 1] \cdot 1 + \Pr[Y_i^2 = 0] \cdot 0 = \Pr[Y_i^2 = 1] = \mathbb{E}[Y_i],$$

$$\mathsf{Var}[Y_i] = \mathbb{E}[Y_i^2] - \mathbb{E}[Y_i]^2 \geq \mathbb{E}[Y_i] = (\frac{q}{n})^{\mathsf{n_p}-1}.$$

According to Theorem 5.8, we have

$$\Pr\left[|Y_i \cdot q - \mathbb{E}[Y_i] \cdot q| \geq \frac{\mathbb{E}[Y_i] \cdot q}{2}\right] \leq \frac{4}{q\mathbb{E}[Y_i]},$$

$$\text{i.e., } \Pr\left[Y_i \cdot q \leq \frac{\mathbb{E}[Y_i] \cdot q}{2} = \lambda\right] \leq \frac{4}{q\mathbb{E}[Y_i]} = \frac{2}{\lambda}.$$

Since the number of indices shared by the parties when $h_{\mathsf{n_p}} \xleftarrow{\$} \mathcal{H}_n$ must be more than that when $h_{\mathsf{n_p}} \xleftarrow{\$} \mathcal{F}_n$, the probability the protocol aborts is at most $\frac{2}{\lambda}$, completing this part of proof.

**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., $\mathbf{s} = \mathbf{u}_{\mathcal{S}}$ be the sequence $\mathbf{u}$ restricted to $u_i \neq \perp$. We have the following lemma.

**Lemma 5.10** $\mathbf{s}$ *is generated by* $\mathcal{H}_n$.

*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 $i, j \in [q]$, 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 $i, j \in \mathbb{Z}_q$, we have

$$\begin{aligned}\Pr[u_i = \hat{x}_1 \wedge u_j = \hat{x}_2] =& \Pr[\exists (i_l, j_l)_{l\in[\mathsf{n_p}-1]} : h_1(i_1) = \cdots = h_{\mathsf{n_p}-1}(i_{\mathsf{n_p}-1}) = \hat{x}_1 = h_{\mathsf{n_p}}(i) \wedge \\ & h_1(j_1) = \cdots = h_{\mathsf{n_p}-1}(j_{\mathsf{n_p}-1}) = \hat{x}_2 = h_{\mathsf{n_p}}(j)] \\ =& \Pr[h_{\mathsf{n_p}}(i) = \hat{x}_1 \wedge h_1(j) = x_2] \cdot \prod_{l=1}^{\mathsf{n_p}} \Pr[\exists i_l, j_l : h_l(i_1) = \hat{x}_1 \wedge h_l(j_l) = \hat{x}_2] \\ =& \frac{1}{n(n-1)} \cdot \Big(\frac{q(q-1)}{n(n-1)}\Big)^{\mathsf{n_p}-1},\end{aligned}$$

and

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

i.e.,

$$\Pr[u_i = x_1 \wedge u_j = x_2 | u_i \neq \perp \wedge u_j \neq \perp] = \frac{\Pr[u_i = \hat{x}_1 \wedge u_j = \hat{x}_2]}{\Pr[u_i \neq \perp \wedge u_j \neq \perp]} = \frac{1}{n(n-1)}.$$

Hence, the distribution of $(u_i, u_j)$ is identical to that of $(h_{\mathsf{n_p}}(i), h_{\mathsf{n_p}}(j))$, conditioned on neither of them being $\perp$. Moreover, for each $l \in \mathcal{S}\backslash\{i, j\}$ we have $u_l = h_{\mathsf{n_p}}(l)$ where $h_{\mathsf{n_p}}$ can be determined by $u_i$ and $u_j$. Hence, the distribution of $(u_i)_{i\in\mathcal{S}}$ is identical to that of $(h_{\mathsf{n_p}}(i))_{i\in\mathcal{S}}$, completing the proof of Lemma 5.10. □

Combining the above lemma with Theorem 5.7, there exists an event $\mathsf{E}$ such that $\Pr[\mathsf{E}] \geq 1 - \epsilon_1 - \epsilon_2$ and $I(\mathsf{key}; g_{\mathsf{n_p}}, \mathbf{z}, \mathbf{s}|\mathsf{E})) \leq \theta$, where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $\mathbf{z} = f(\mathsf{urs})$, and $\mathsf{key} = g_{\mathsf{n_p}}(\mathsf{urs_s})$. Moreover, since $\mathbf{s}$ is determined by $\{h_i\}_{i\in[\mathsf{n_p}]}$ and the public keys $(\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}$ between all parties consist only of $\{h_i, g_i\}_{i\in[\mathsf{n_p}]}$, which contains no more information on $\mathsf{key}$ than $\mathbf{s}$, we have

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

Putting all the above together, Theorem 5.9 immediately follows. □

**Example with asymptotic complexity.** Let $n = \lambda^{\mathsf{n_p}+1}$, $\mathsf{m_a} = \lambda^{\mathsf{n_p}+1}/2$, $\epsilon_1 = 1/\lambda^{\mathsf{n_p}}$, $\epsilon_2 = 1/\sqrt{\lambda}$. We have $r = O(\lambda)$. In this case, each $\mathsf{P}_{i\in[\mathsf{n_p}]}$ consumes about $q = n\sqrt[\mathsf{n_p}]{\frac{2\lambda}{n}} = O(\lambda^{\mathsf{n_p}})$ 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 [11]. Assume that that all users have access to a 40 Gbit/s broadcast channel used for $2 \times 10^5$ 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 $\epsilon_2 = 10^{-2}$, the parameters are $\delta = 0.476$, $p = 0.077$, $\lambda = 1.3 \times 10^7$, 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 share $\lambda$ bits in a 3-party protocol, each party consumes about $1.2 \times 10^{13}$ bits of memory, and the communication consist of about $3.9 \times 10^7$ 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 Appendix A for the details.

# 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, 5.)

[3] 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.)

[4] 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, 3.)

[5] 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.)

[6] 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.)

[7] 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.)

[8] 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 6, 27.)

[9] 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.)

[10] 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.)

[11] 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, 5, 7, 20, 21, 23.)

[12] 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, 3.)

[13] 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 1, 2, 3, 8, 9.)

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

[15] 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, 7.)

[16] Egashira, S., Wang, Y., Tanaka, K.: Fine-grained cryptography revisited. Journal of Cryptology 34(3), 23 (Jul 2021) (Cited on page 2, 3, 4, 8, 9.)

[17] 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.)

[18] 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.)

[19] Goldreich, O., Levin, L.A.: A hard-core predicate for all one-way functions. In: STOC. pp. 25–32. ACM (1989) (Cited on page 5, 19.)

[20] 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.)

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

[22] 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 4.)

[23] 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 3, 6, 28.)

[24] Joux, A.: A one round protocol for tripartite diffie-hellman. J. Cryptol. 17(4), 263–276 (2004) (Cited on page 1, 3.)

[25] 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, 3, 4, 13, 14.)

[26] 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.)

[27] 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.)

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

[29] Mitchell, C.J.: A storage complexity based analogue of maurer key establishment using public channels. In: Boyd, C. (ed.) 5th IMA International Conference on Cryptography and Coding. LNCS, vol. 1025, pp. 84–93. Springer, Berlin, Heidelberg (Dec 1995) (Cited on page 2.)

[30] 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, 8.)

[31] 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.)

[32] 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, 8.)

[33] 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, 3, 4, 9.)

[34] 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, 3, 9.)

[35] 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 20, 21.)

# A Fine-Grained Multi-Party IB-NIKE and BPRF

In this section, we propose multi-party IB-NIKE schemes in the fine-grained settings. 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. In this whole section, we only focus on IB-NIKE for a constant number with a larger and constant-size identity space of parties, and BPRF with constant-size input space.

## A.1 Definitions

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

**Definition A.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 $\mathsf{ibmsk}$.
- $\mathsf{IBExtract}_\lambda$ is a deterministic algorithm that on input a master secret key $\mathsf{ibmsk}$ 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 $\mathsf{id}$.
- $\mathsf{IBShare}_\lambda$ 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 a shared key $\mathsf{ibshk} \in \mathcal{K}_\lambda$ for $\mathcal{I}$, where $\mathcal{K}_\lambda$ is the shared key space. We assume that the identities in $\mathcal{I}$ are always lexicographically ordered.

*$\epsilon$-correctness* is satisfied if for all $\lambda \in \mathbb{N}$, all list 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 $\mathsf{ibmsk} \xleftarrow{\$} \mathsf{IBSetup}_\lambda(\mathsf{urs})$.

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

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

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

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

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

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

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 NIKEs as we will show later, as 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 [8].

**Definition A.2** (Bit-fixing pseudorandom functions). A $\mathcal{C}_1$-$\ell_{\mathsf{inp}}$-*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

- $\mathsf{Setup}_\lambda$ on input $\mathsf{urs} \in \{0,1\}^n$ for some $n = n(\lambda)$ outputs a master secret key $\mathsf{msk}$.
- $\mathsf{Eval}_\lambda$ is a deterministic algorithm that on input a master secret key $\mathsf{msk}$ and a preimage $\mathsf{x} \in \{0,1\}^{\ell_{\mathsf{inp}}}$ outputs an image $\mathsf{y} \in \mathcal{Y}$.
- $\mathsf{Constrain}_\lambda$ is a deterministic algorithm that on input a master secret key $\mathsf{msk}$ 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.
- $\mathsf{CEval}_\lambda$ 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 an image $\mathsf{y} \in \mathcal{Y}$.

*$\epsilon$-correctness* is satisfied if for all $\hat{\mathsf{x}} \in (\{0,1,\bot\})^{\ell_{\mathsf{inp}}}$ and all $\mathsf{x}$ 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 $\mathsf{urs} \leftarrow \{0,1\}^n$ and $\mathsf{msk} \xleftarrow{\$} \mathsf{Setup}_\lambda(\mathsf{urs})$.

*$\mathcal{C}_2$-$\epsilon$-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

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

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 $\hat{\mathsf{x}} \in \mathcal{T}_{\mathsf{x},\ell_{\mathsf{inp}}}$.

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

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

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 A.3** (Restricted bit-fixing pseudorandom functions.). A *$\mathcal{C}_1$-$\ell_{\mathsf{inp}}$-restricted bit-fixing pseudorandom function (restricted BPRF)* is defined in exactly the same way as $\mathcal{C}_1$-$\ell_{\mathsf{inp}}$-BPRF except that we require that $\hat{\mathsf{x}} \in \{\bot\}^{\mathsf{t}_1 - 1} \cup \{0,1\}^{\mathsf{t}_2 - \mathsf{t}_1 + 1} \cup \{\bot\}^{\ell_{\mathsf{inp}} - \mathsf{t}_2}$ for all $\mathsf{t}_1, \mathsf{t}_2 \in [\ell_{\mathsf{inp}}]$.

## A.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 that in [23], with the distinction that the underlying BPRF can be a restricted BPRF 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 $\mathsf{AC}^0[2]$-$\mathsf{n}_\mathsf{p}$-restricted BPRF with URS length $n$ and image space $\mathcal{Y}$ for some constant $\mathsf{n}_\mathsf{p}$, and some $n = n(\lambda)$. Let $\mathsf{m}_\mathsf{p}$ be any constant. Our $\mathsf{NCIB\text{-}NIKE} = \{\mathsf{IBSetup}_\lambda, \mathsf{IBExtract}_\lambda, \mathsf{IBShare}_\lambda\}$ with identity space $\{0,1\}^{\mathsf{n}_\mathsf{p}/\mathsf{m}_\mathsf{p}}$ and URS length $n$ is defined as in Figure 15.

**Theorem A.4** *If* NCBPRF *satisfies $\epsilon_1$-correctness and $\mathsf{NC}^1$-$\epsilon_2$-security, then* NCIB-NIKE *in Figure 15 is an $\mathsf{AC}^0[2]$-$\mathsf{m}_\mathsf{p}$-IB-NIKE with $2 \cdot \epsilon_1$-correctness and $\mathsf{NC}^1$-$\epsilon_2$-security.*

*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 $\mathsf{CEval}_\lambda$ for a constant of times.

**Correctness.** For any list of $\mathsf{m}_\mathsf{p}$ distinct identities $\mathcal{I}^* \subseteq \{0,1\}^{\mathsf{n}_\mathsf{p}/\mathsf{m}_\mathsf{p}}$ and any $i, j \in [\mathsf{m}_\mathsf{p}]$, by the $\epsilon_1$-correctness of NCBPRF, we have

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

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

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 $\hat{\mathsf{x}}_i \in \{0,1,\bot\}^{\mathsf{n}_\mathsf{p}}$ and any $i \in [\mathsf{m}_\mathsf{p}]$. Therefore by a union bound, we have

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

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

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

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

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

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

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 $\mathsf{NCIB\text{-}NIKE} = \{\mathsf{IBSetup}_\lambda, \mathsf{IBExtract}_\lambda, \mathsf{IBShare}_\lambda\}$.

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

On receiving $(\mathsf{y}, \{\mathsf{bfk}_{\hat{\mathsf{x}}}\}_{\hat{\mathsf{x}}\in\mathcal{T}_{\mathsf{x},\mathsf{n_p}}})$ where $\mathsf{bfk}_{\hat{\mathsf{x}}} = \mathsf{Constrain}_\lambda(\mathsf{msk}, \hat{\mathsf{x}})$ for all $\hat{\mathsf{x}} \in \mathcal{T}_{\mathsf{x},\mathsf{n_p}}$ and $\mathsf{y} \xleftarrow{\$} \mathsf{Eval}_\lambda(\mathsf{msk}, \mathsf{x})$ or $\mathsf{y} \xleftarrow{\$} \mathcal{Y}$ where $\mathsf{msk} \xleftarrow{\$} \mathsf{Setup}_\lambda(\mathsf{urs})$ for $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $\mathsf{advb}_\lambda$ returns $\mathsf{ibshk} = \mathsf{y}$ and $\{\mathsf{bfk}_{\hat{\mathsf{x}}}\}_{\hat{\mathsf{x}}\in\mathcal{T}_{\mathsf{x},\mathsf{n_p}}}$ to $\mathsf{adv}_\lambda$. On receiving $b \in \{0,1\}$ from $\mathsf{adv}_\lambda$, $\mathsf{advb}_\lambda$ forwards $b$ to its challenger.

First we note that all operations in $\mathsf{advb}_\lambda$ are in $\mathsf{NC}^1$, since $\mathsf{advb}_\lambda$ does not execute any additional operations except running $\mathsf{adv}_\lambda$. Moreover, when $\mathsf{y} \xleftarrow{\$} \mathsf{Eval}_\lambda(\mathsf{msk}, \mathsf{x})$ (respectively, $\mathsf{y} \xleftarrow{\$} \mathcal{Y}$) for any $\mathsf{x} \in \{0,1\}^{\mathsf{n_p}}$ where $\mathsf{msk} \xleftarrow{\$} \mathsf{Setup}_\lambda(\mathsf{urs})$ for $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, the view of $\mathsf{adv}_\lambda$ is identical to its view in the security game for NCIB-NIKE when ibshk is honestly generated (respectively, generated as a randomness). Hence, the advantage of $\mathsf{advb}_\lambda$ in breaking the security of NCBPRF is the same as the advantage of $\mathsf{adv}_\lambda$ in breaking the security of NCIB-NIKE, completing the proof of Theorem A.4. □

**Construction in the bounded time model and BSM.** Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the set 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 $\lambda \in \mathbb{N}$, $f_\lambda^1$ runs in time $\tilde{O}(T_1(\lambda))$ and $f_\lambda^2$ runs in time $\tilde{O}(T_2(\lambda))$. Let $\mathsf{m_p}$ be any constant. 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 $\ell$ for some constant $\mathsf{n_p}$, some $\ell$, and some $n = n(\lambda)$. Notice that for our instantiation of $\mathcal{C}_1$-$\mathsf{n_p}$-BPRF given later in Appendix A.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. 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 $\ell'$ for some constant $\mathsf{n_p}$, some $\ell'$, and some $n' = n'(\lambda)$. We have the following theorems.

**Theorem A.5** *If* TBPRF *is a* $\mathcal{C}_1$-$\mathsf{n_p}$*-BPRF with* $\epsilon_1$*-correctness and* $\mathcal{C}_2$-$\epsilon_2$*-security, then* TIB-NIKE *is a* $\mathcal{C}_1$-$\mathsf{m_p}$*-IB-NIKE with* $2 \cdot \epsilon_1$*-correctness and* $\mathcal{C}_2$-$\epsilon_2$*-security.*

**Theorem A.6** *If* BSBPRF *is a* $\mathcal{C}_1'$-$\mathsf{n_p}$*-BPRF with* $\epsilon_1'$*-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* $2 \cdot \epsilon_1'$*-correctness and* $(\mathsf{m_a}, \epsilon', \epsilon_2')$*-security in the BSM.*

The proof of Theorems A.5 and A.6 are very similar to that of Theorem A.4 and thus we omit the details.

## A.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 $\epsilon_1$-extendability with associated algorithm families $\{\mathsf{Combine}_\lambda, \mathsf{Extract}_\lambda\}$ and $\mathsf{NC}^1$-$\epsilon_2$-security. Our $\mathsf{NCBPRF} = \{\mathsf{Setup}_\lambda, \mathsf{Eval}_\lambda, \mathsf{Constrain}_\lambda, \mathsf{CEval}_\lambda\}$ is defined as in Figure 16.

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

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

$(\mathsf{pk}_{i,\beta}, \mathsf{sk}_{i,\beta}) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ 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 $\mathcal{V} = \{i \in [\mathsf{n_p} - 1] \mid \hat{\mathsf{x}}_i \neq \bot\}$, $j \in \mathcal{V}$

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

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

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

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 $\mathsf{NCBPRF} = \{\mathsf{Setup}_\lambda, \mathsf{Eval}_\lambda, \mathsf{Constrain}_\lambda, \mathsf{CEval}_\lambda\}$.

**Theorem A.7** NCBPRF *is an* $\mathsf{AC}^0[2]$-$(\mathsf{n_p} - 1)$*-restricted BPRF with* $\epsilon_1$*-correctness and* $\mathsf{NC}^1$-$\epsilon_2$*-security.*

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

**Correctness.** Let $\mathsf{t}_1, \mathsf{t}_2 \in [\mathsf{n_p} - 1]$. For any $\hat{\mathsf{x}} = (\hat{\mathsf{x}}_i)_{i\in[\mathsf{n_p}-1]} \in \{\bot\}^{\mathsf{t}_1-1} \cup \{0,1\}^{\mathsf{t}_2-\mathsf{t}_1+1} \cup \{\bot\}^{\mathsf{n_p}-1-\mathsf{t}_2}$, we have $\mathcal{V} = [\mathsf{t}_1, \mathsf{t}_2]$. Moreover, for any $j \in [\mathsf{n_p} - 1]$, according to the $\epsilon_1$-restricted extendability of NCNIKE, there exists some event $\mathsf{E}$ such that $\Pr[\mathsf{E}] \geq 1 - \epsilon_1$ and conditioned on $\mathsf{E}$, we have

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

where $\mathsf{urs} \xleftarrow{\$} \{0,1\}^n$, $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for $i \in [\mathsf{n_p}]$, $(\mathsf{pk}_{i,\beta}, \mathsf{sk}_{i,\beta}) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$, and $\mathsf{localkey}^* = \mathsf{Combine}_\lambda(\mathsf{localkey}, \{\{\mathsf{pk}_{i,\mathsf{x}_i}\}_{i\in[\mathsf{n_p}-1]\setminus\mathcal{V}}, \mathsf{pk}_{\mathsf{n_p}}\})$ for $\mathsf{localkey} = \mathsf{Combine}_\lambda(\mathsf{sk}_{i'}, \{\mathsf{pk}_i\}_{i\in\mathcal{V}\setminus\{i'\}})$ for any $i' \in \mathcal{V}$. Thus for any $\mathsf{x} = (\mathsf{x}_i)_{i\in[\mathsf{n_p}-1]} \in \{0,1\}^{\mathsf{n_p}-1}$ such that $\mathsf{x}_i = \hat{\mathsf{x}}_i \vee \hat{\mathsf{x}}_i = \bot$ for all $i \in [\mathsf{n_p} - 1]$, and according to the $\epsilon_1$-restricted extendability of NCNIKE, we have

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

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

**Security.** Let $\lambda$ be the security parameter, and $\mathcal{A} = \{\mathsf{adv}_\lambda\} \in \mathsf{NC}^1$ be any adversary against the $\mathsf{NC}^1$-security of NCNIKE. 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 $((\mathsf{pk}_i)_{i\in[\mathsf{n_p}]}, \mathsf{key})$ 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 $\mathsf{key} \xleftarrow{\$} \mathcal{K}_\lambda$, $\mathsf{advb}_\lambda$ firstly samples $(\mathsf{pk}'_i, \mathsf{sk}'_i) \xleftarrow{\$} \mathsf{Gen}_\lambda(\mathsf{urs})$ for every $i \in [\mathsf{n_p} - 1]$. Then $\mathsf{advb}_\lambda$ 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 $\hat{\mathsf{x}} \in \mathcal{T}_{\mathsf{x},\mathsf{n_p}-1}$ 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 $\mathsf{advb}_\lambda$. Then $\mathsf{advb}_\lambda$ returns $\mathsf{y} = \mathsf{key}$ and $\{\mathsf{bfk}_{\hat{\mathsf{x}}}\}_{\hat{\mathsf{x}}\in\mathcal{T}_{\mathsf{x},\mathsf{n_p}-1}}$ to $\mathsf{adv}_\lambda$. When $\mathsf{adv}_\lambda$ finally returns $b \in \{0,1\}$, $\mathsf{advb}_\lambda$ forwards $b$ to its challenger.

First we note that all operations in $\mathsf{advb}_\lambda$ are in $\mathsf{NC}^1$, since $\mathsf{advb}_\lambda$ runs $\mathsf{Gen}_\lambda$ 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 $\mathsf{adv}_\lambda$ is identitcal to its view in the security game for NCBPRF when y is honestly generated (respectively, generated as a randomness). Hence, the advantage of $\mathsf{advb}_\lambda$ in breaking the security of NCNIKE is the same as the advantage of $\mathsf{adv}_\lambda$ in breaking the security of BPRF, completing the proof of Theorem A.7. □

**BPRF in the bounded time model and BSM.** Let $\mathcal{C}_1$ and $\mathcal{C}_2$ represent the set 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 $\lambda \in \mathbb{N}$, $f^1_\lambda$ runs in time $\tilde{O}(T_1(\lambda))$ and $f^2_\lambda$ 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 $\epsilon_1$-extendability and $\mathcal{C}_2$-$\epsilon_2$-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 BSNIKE, which is a $\mathcal{C}'_1$-$\mathsf{n_p}$-NIKE with shared key space $\mathcal{K}'_\lambda$ and URS length $n'$ satisfying $\epsilon'_1$-extendability and $(\mathsf{m_a}, \epsilon', \epsilon'_2)$-security in the BSM for some constant $\mathsf{n_p}$, some $\ell'$, and some $n' = n'(\lambda)$.

**Theorem A.8** TBPRF *is a* $\mathcal{C}_1$-$(\mathsf{n_p}-1)$-*BPRF with* $\epsilon_1$-*correctness and* $\mathcal{C}_2$-$\epsilon_2$-*security.*

**Theorem A.9** BSBPRF *is a* $\mathcal{C}'_1$-$(\mathsf{n_p}-1)$-*BPRF with* $\epsilon'_1$-*correctness and* $(\mathsf{m_a}, \epsilon', \epsilon'_2)$-*security.*

The proof of Theorem A.8 and Theorem A.9 are very similar to that of Theorem A.7 and thus we omit the details.

## A.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 (see Figure 5) satisfies 0-correctness, $\mathsf{TNIKE}^* = \{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\}$ (see Figure 12) satisfies $((1-(\frac{q}{\ell})^{\mathsf{n_p}-1})^q + 2\ell\mathsf{n_p}\lambda^k/R)$-correctness, and BSMKE (see Figure 14) satisfies $\frac{2}{\lambda}$-correctness. Then we have the following theorem.

**Theorem A.10** NCNIKE *(see Figure 5) satisfies* 0*-restricted extendability,* $\mathsf{TNIKE}^* = \{\mathsf{Gen}_\lambda, \mathsf{Share}_\lambda\}$ *(see Figure 12) satisfies* $(2\ell\mathsf{n_p}\lambda^k/R + (1-(\frac{q}{\ell})^{\mathsf{n_p}-1})^q)$*-extendability, and* BSMKE *(see Figure 14) satisfies* $\frac{2}{\lambda}$*-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. Then Theorem A.10 follows immediately from the analysis of complexity and proofs of correctness for NCNIKE, TNIKE, and BSMKE (see Theorems 3.8, 4.8 and 5.9). Indeed, one can easily see that for each scheme, the combining and extracting algorithms are exactly the intermediate steps of the sharing algorithm.

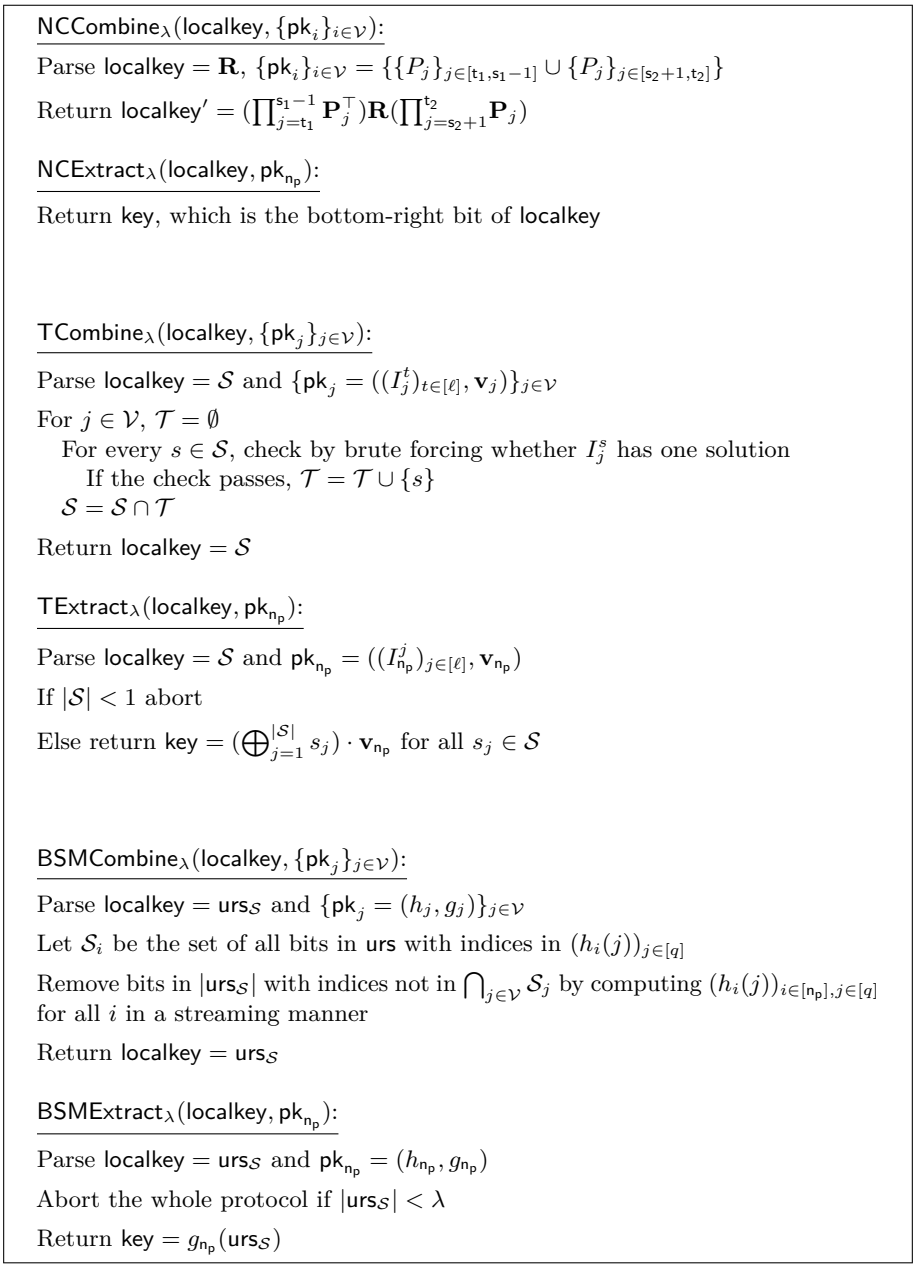


Figure 17: Constructions of $\{\mathsf{NCCombine}_\lambda, \mathsf{NCExtract}_\lambda\}$, $\{\mathsf{TCombine}_\lambda, \mathsf{TExtract}_\lambda\}$, and $\{\mathsf{BSMCombine}_\lambda, \mathsf{BSMExtract}_\lambda\}$. Recall that for any set $\mathcal{S}$, $\mathsf{urs}_\mathcal{S}$ denotes the sequence of all bits in urs with indices (ordered lexicographically) in $\mathcal{S}$.