Sakai-Ohgishi-Kasahara Identity-Based Non-Interactive Key Exchange Revisited and More

Yu Chen[*] yuchen.prc@gmail.com

Qiong Huang[†] csqhuang@gmail.com

Zongyang Zhang[‡] zongyang.zhang@gmail.com

December 1, 2014

Abstract

Identity-based non-interactive key exchange (IB-NIKE) is a powerful but a bit overlooked primitive in identity-based cryptography. While identity-based encryption and signature have been extensively investigated over the past three decades, IB-NIKE has remained largely unstudied. Currently, there are only few IB-NIKE schemes in the literature. Among them, Sakai-Ohgishi-Kasahara (SOK) scheme is the first efficient and secure two-party IB-NIKE scheme, which has great influence on follow-up works. However, the SOK scheme required its identity mapping function to be modeled as a random oracle to prove security. Moreover, its existing security proof heavily relies on the ability of programming the random oracle. It is unknown whether such reliance is inherent.

In this work, we intensively revisit the SOK IB-NIKE scheme, and present a series of possible and impossible results in the random oracle model and the standard model. In the random oracle model, we first improve previous security analysis for the SOK IB-NIKE scheme by giving a tighter reduction. We then use meta-reduction technique to show that the SOK scheme is unlikely proven to be secure based on the computational bilinear Diffie-Hellman (CBDH) assumption without programming the random oracle. In the standard model, we show how to instantiate the random oracle in the SOK scheme with a concrete hash function from admissible hash functions (AHFs) and indistinguishability obfuscation. The resulting scheme is adaptively secure based on the decisional bilinear Diffie-Hellman inversion (DBDHI) assumption. To the best of our knowledge, this is the first adaptively secure IB-NIKE scheme in the standard model that does not explicitly require multilinear maps. Previous schemes in the standard model either have merely selective security or require programmable hash functions in the multilinear setting. At the technical heart of our scheme, we generalize the definition of AHFs, and propose a generic construction which enables AHFs with previously unachieved parameters, which might be of independent interest.

In addition, we present some new results about IB-NIKE. On the first place, we present a generic construction of multiparty IB-NIKE from extractable witness PRFs and existentially

[*] State Key Laboratory of Information Security (SKLOIS), Institute of Information Engineering, Chinese Academy of Sciences, China. Yu Chen is supported by the National Natural Science Foundation of China under Grant No. 61303257, 61379141, the IIEs Cryptography Research Project, the Strategic Priority Research Program of CAS (Chinese Academy of Sciences) under Grant No. XDA06010701, and the National 973 Program of China under Grant No. 2011CB302400.

[†] College of Informatics, South China Agricultural University, China. Qiong Huang is supported by the National Natural Science Foundation of China under Grant No. 61103232, 61472146, the Guangdong Natural Science Foundation under Grant No. S2013010011859, and the Research Fund for the Doctoral Program of Higher Education of China under Grant No. 20114404120027.

[‡] Research Institute for Secure Systems, National Institute of Advanced Industrial Science and Technology (AIST), Japan. Zongyang Zhang is an International Research Fellow of JSPS and supported by the National Natural Science Foundation of China under Grant No. 61303201.

unforgeable signatures. On the second place, we investigate the relation between semi-adaptive security and adaptive security for IB-NIKE. Somewhat surprisingly, we show that these two notions are polynomially equivalent.

Keywords: identity-based non-interactive key exchange, non-programming ROM, meta-reduction, indistinguishability obfuscation, puncturable PRFs, admissible hash functions, extractable witness PRFs

1 Introduction

Identity-based non-interactive key exchange (IB-NIKE) is an analog of NIKE [DH76] in the identity-based setting, which enables a group of users registered in the same key generator center (KGC) to agree on a unique shared key without any interaction. IB-NIKE has important applications in managing keys and enabling secure communications in mobile ad hoc and sensor networks. The advantages of IB-NIKE, in terms of reducing communication costs and latency in a realistic adversarial environment, are demonstrated in [CGP$^+$13].

In 2000, Sakai, Ohgishi and Kasahara [SOK00] proposed the first efficient two-party IB-NIKE scheme in the random oracle model, namely the SOK scheme (with security models and formal proofs in follow-up works [DE06, PS09]). Despite the appearing of IB-NIKE in this celebrated work [SOK00] on identity-based cryptography, it had received less attention as a fundamental primitive in its own right over the past decade. In the last year, we have seen remarkable progress on this topic. Freire et al. [FHPS13] constructed $(\mathsf{poly}, 2)$-programmable hash functions (PHFs) from multilinear maps. By substituting the random oracle in the original SOK scheme with $(\mathsf{poly}, 2)$-PHFs, they obtained the first two-party IB-NIKE scheme in the standard model. Boneh and Waters [BW13] demonstrated that constrained pseudorandom functions (CPRFs) that support left/right predicate imply two-party IB-NIKE. Particularly, they constructed such specific CPRFs based on the decisional bilinear Diffie-Hellman (DBDH) assumption, and the resulting IB-NIKE scheme (the BW scheme) can be viewed as a variant of the SOK scheme, which is also only proven secure in the random oracle model. Boneh and Zhandry [BZ14] proposed a construction of multiparty IB-NIKE from pseudorandom generator, constrained PRFs, and indistinguishability obfuscation. However, their construction only has selective security. Very recently, Hofheinz [Hof14] constructed the first fully secure "bix-fixing" CPRFs, which directly gives rise to the first adaptively secure multiparty IB-NIKE. However, currently the construction of such powerful CPRFs requires several heavyweight tools, including multilinear maps, indistinguishability obfuscation, and random oracles.

1.1 Motivations

For a security reduction $\mathcal{R}$ that converts any adversary $\mathcal{A}$ with advantage $\mathsf{Adv}_\mathcal{A}$ against some hard problem in running time $\mathsf{Time}_\mathcal{A}$ to an algorithm $\mathcal{B}$ with advantage $\mathsf{Adv}_\mathcal{B}$ against the target cryptographic scheme in running time $\mathsf{Time}_\mathcal{B}$, we say it is tight if $\mathsf{Adv}_\mathcal{B}/\mathsf{Adv}_\mathcal{A}$ (advantage loose factor) is close to 1 and $\mathsf{Time}_\mathcal{B} - \mathsf{Time}_\mathcal{A}$ (time loose factor) is close to 0, and loose otherwise. It has been well known that besides theoretical interest, a tighter reduction is of utmost practical importance. To obtain the same security level, cryptographic schemes with tighter reduction generally admit more efficient implementations [BR09]. The existing proof [PS09] for the SOK scheme programs the random oracle H (acting as the identity mapping function in the construction) with "all-but-one" technique to implement partitioning strategy.[1] As a consequence, the advantage loose factor is around $1/2^{180}$, which is far from tight. It is interesting to know if we can provide an alternative proof with tighter reduction.

Both the original security reduction [PS09] and our new security reduction (as we will show in Section 3.1) for the SOK scheme exploit full programmability of the random oracle model (ROM) to implement partitioning strategy. As we recall in Section 2.2, such property allows the reduction to program the random oracle (RO) arbitrarily as long as the output distributes uniformly and independently over the range. This full-fledged model is usually refereed as fully programming ROM (FPROM). Full programmability is a strong property in that it does not

[1] In the case of IB-NIKE, the partitioning strategy is to partition the set of all identities into "extractable" and "unextractable" ones. The reduction hopes that all identities for which an adversary requests for a secret key are extractable, while the target identities are unextractable.

quite match with the features of cryptographic hash functions. Therefore, two weaker random oracle models are proposed by constraining the ability of the reduction to program the RO. The randomly programming ROM (RPROM) [FLR$^+$10] allows the reduction to program the RO with random instead of arbitrary values, while the non-programming ROM (NPROM) forbids the reduction to program the RO. Since the NPROM is the weakest one among the above three random oracle models and is closest to the standard model, it is curious to know if the SOK scheme could be proven secure in the NPROM.

As previously mentioned, Freire et al. [FHPS13] successfully instantiated the SOK scheme in the standard model by substituting the random oracle H with $(\mathsf{poly}, 2)$-programmable hash functions (PHFs). However, the construction of $(\mathsf{poly}, 2)$-PHFs requires multilinear maps [GGH13a]. So far, we do not have candidates for multilinear maps between groups with cryptographically hard problems. Instead, we only have concrete candidate for an "approximation" of multilinear maps, named graded encoding systems [GGH13a]. Hence, we are motivated to find an alternative approach of substituting the random oracle in the SOK scheme, with the hope that the replacements are not explicitly involved with multilinear maps. Recently, Hohenberger, Sahai and Waters [HSW14] gave a way to instantiate the random oracle with concrete hash functions from indistinguishability obfuscation[2] in the "full domain hash" signatures. It is natural to ask if their approach can extend to other applications, and in particular, the SOK scheme. After shifting our attention from the SOK scheme to general IB-NIKE scheme, we find that currently there is no satisfying generic construction of IB-NIKE. Existing constructions either only have selective security or rely on random oracles. We are motivated to find a new generic construction which enjoys adaptive security in the standard model. We also note that there exist two notions capturing the full security[3] of IB-NIKE, namely semi-adaptive security and adaptive security. Although the former one seems weaker than the latter one, it is unknown if there exists a black-box separation between them. It is of theoretical interest to prove or disprove it.

1.2 Our Results

In the remainder of this paper, we give negative or affirmative answers to the above questions. We summarize our main results as below:

Being aware of the usage of "all-but-one" programming technique is the reason that makes the original reduction loose, we are motivated to find an alternative programming technique that admits tighter reduction. Observing the structural similarities between the SOK IB-NIKE scheme and the Boneh-Franklin [BF01] IBE scheme and the Boneh-Lynn-Shacham (BLS) [BLS01] short signature, we are inspired to program the random oracle H in the SOK scheme with the flipping coin technique developed in [Cor00], which were successfully employed in the reductions for the latter two well-known schemes. Roughly speaking, the flipping coin technique usually conducts as follows: to program $\mathsf{H}(x)$ (x is an identity in the IBC setting or a message in the signature setting), the reduction flips a random coin once, then programs $\mathsf{H}(x)$ according to the coin value in two different manners. One allows the reduction to embed a trapdoor in order to extract a secret key or produce a signature, while the other allows the reduction to embed some fixed component of the challenge instance. However, this approach does not work well in the case of the SOK scheme. This is because the reduction has to embed two group elements g_2 and g_3 from the CBDH instance to $\mathsf{H}(id_a^*)$ and $\mathsf{H}(id_b^*)$ respectively, where id_a^* and id_b^* are two target identities adaptively chosen by the adversary. We overcome this difficulty by flipping random coins twice. Looking ahead, to program $\mathsf{H}(x)$, the reduction first flips a random

[2] Although currently the only known construction of indistinguishability obfuscation ($i\mathcal{O}$) is from multilinear maps [GGH$^+$13c], it is still possible that $i\mathcal{O}$ can be constructed from other primitives.

[3] Full security allows the adversary to arbitrarily choose the target identities. In contrast, selective security requires that the adversary has to commit the target identities at the very beginning.

biased coin to determine the partitioning, namely either embedding a trapdoor or embedding a component from the CBDH instance. If the first round coin value indicates the latter choice, then $\mathcal{R}$ further flips an independent and unbiased coin to determine which component is going to be embedded. As a result, we obtain a new reduction with a loose factor around $1/2^{120}$, which significantly improves the original result. We note that the same technique can also be used to improve Boneh-Waters constrained PRFs supporting left/right predicate [BW13], by minimizing the number of RO and tightening the reduction.

Following the work of Fischlin and Fleischhacker [FF13], we use meta-reduction technique to show that the SOK scheme is unlikely proven secure to be based on the CBDH assumption in NPROM, assuming the hardness of an intractable problem called one-more CBDH problem. We obtain this result by showing that if there is a black-box reduction $\mathcal{R}$ basing the adaptive security of the SOK IB-NIKE scheme on the CBDH assumption in NPROM, then there exists a meta-reduction $\mathcal{M}$ breaking the one-more CBDH assumption. Our black-box separation result holds with respect to single-instance reduction which invokes only one instance of the adversary and can rewind it arbitrarily to the point after sending over the master public key. Though single-instance reduction is a slightly restricted type of reductions, it is still general enough to cover the original reduction [PS09] and our new reduction shown in Section 3.1. Moreover, our result holds even for selective semi-adaptive one-way security.

Realizing the technical heart of Hohenberger-Sahai-Waters approach [HSW14] is to replace the programmable RO with a specific hash function H satisfying suitable programmability, we successfully extend their approach in the case of IB-NIKE, which goes beyond the "full domain hash" signatures. More precisely, we first create a replacement hash function H for RO from puncturable PRFs and $i\mathcal{O}$. The resulting IB-NIKE scheme is selective-secure in the standard model. To attain adaptive security, we hope to create a specific hash function H with $(\mathsf{poly}, 2)$-programmability from admissible hash functions (AHFs) and $i\mathcal{O}$. This potentially requires the AHFs to be $(\mathsf{poly}, 2)$-admissible, which is not met by current constructions of AHFs. We circumvent this technical difficulty by giving a generic construction of (poly, n)-AHF (n could be any constant integer) from any $(\mathsf{poly}, 1)$-AHF, which utilizes Cartesian product as the key mathematical tool. We note that beyond the usage in the above construction, (poly, c)-AHF may find more important applications as an information-theoretically cryptographic primitive.

When broadening our horizon to general IB-NIKE, we present a generic construction of multiparty IB-NIKE from extractable witness PRFs and existentially unforgeable signatures. We also study the relation between semi-adaptive security and adaptive security for IB-NIKE. Somewhat surprisingly, we show that these two security notions are polynomially equivalent.

2 Preliminaries and Definitions

Notations. For a distribution or random variable X, we write $x \xleftarrow{\text{R}} X$ to denote the operation of sampling a random x according to X. For a set X, we use $x \xleftarrow{\text{R}} X$ to denote the operation of sampling x uniformly at random from X, use U_X to denote the uniform distribution over set X, and use $|X|$ to denote its size. We write κ to denote the security parameter, and all algorithms (including the adversary) are implicitly given κ as input. We write $\mathsf{poly}(\kappa)$ to denote an arbitrary polynomial function in κ. We write $\mathsf{negl}(\kappa)$ to denote an arbitrary negligible function in κ, one that vanishes faster than the inverse of any polynomial. A probability is said to be overwhelming if it is $1-\mathsf{negl}(\kappa)$, and said to be noticeable if it is $1/\mathsf{poly}(\kappa)$. A probabilistic polynomial-time (PPT) algorithm is a randomized algorithm that runs in time $\mathsf{poly}(\kappa)$.

For every NP language L, we associate a corresponding relation R_L such that an instance $x \in L$ iff there exists a witness w such that $(x, w) \in \mathsf{R}_L$. Furthermore, we say that an instance x is a "valid" (or a true) instance iff $x \in L$. Correspondingly, those instances that don't belong to the language are refereed to as invalid (or false) statements.

2.1 Cartesian Product and Power of Vectors

The Cartesian product of a m-dimension vector $X = (x_1, \ldots, x_m)$ and a n-dimension vector $Y = (y_1, \ldots, y_n)$ over some finite set S is defined as:

$$X \times Y = \{z_{ij} := z_{(i-1)n+j} = (x_i, y_j)\}_{1 \leq i \leq m, 1 \leq j \leq n},$$

where × denotes the Cartesian product operation, and $X \times Y$ can be viewed as a mn-dimension vector over S^2 or a $2mn$-dimension vector over S. The Cartesian k-power of a m-dimension vector $X = (x_1, \ldots, x_m)$ over S is defined as:

$$X^k = \underbrace{X \times \cdots \times X}_{k},$$

where X^k can be viewed as a m^k-dimension vector over S^k or a km^k-dimension vector over S.

2.2 Random Oracle Model

Random oracle model (ROM) [FS86, BR93] is a paradigm of designing and analyzing cryptographic schemes that offers trade-off between provable security and practical efficiency. When implementing ROM, some hash function $\mathsf{H} : X \to Y$ is idealized as a publicly accessible random function (random oracle), which on input $x \in X$ outputs a random and independent value $y \in Y$.

The standard ROM implicitly embodies another two properties, namely *observability* and *programmability* [FLR+10]. The observability means that the reduction can see all RO queries made by the adversary, whereas the programmability means that the reduction can totally control the answers to RO queries. Since we focus on programmability in this work, we recap the classification of ROM according to programmability as below.

Full-Programmable ROM: This formalizes the standard ROM, in which the reduction can program the RO arbitrarily as long as the outputs distribute randomly and independently over the range.

Random-Programming ROM: This formalizes a restricted version of ROM, in which the reduction can program the RO with random instead of arbitrary values.

Non-Programmable ROM: This formalizes the most restricted version of ROM, in which the reduction has no control of the answers of RO queries. In non-programmable ROM, the RO queries are answered by invariable external ROs, which are independent of reduction. Nevertheless, the reduction can still observe all the RO queries issued by adversary, but has no influence on the answers.

2.3 Bilinear Maps and Related Hardness Assumptions

A bilinear group system consists of two cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p, along with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ which satisfies the following properties:

- bilinear: $\forall g \in \mathbb{G}$ and $\forall a, b \in \mathbb{Z}_p$, we have $e(g^a, g^b) = e(g, g)^{ab}$.
- non-degenerate: $\forall g \in \mathbb{G}^*$, we have $e(g, g) \neq 1_{\mathbb{G}_T}$.

In the following, we write $\mathsf{BLGroupGen}$ to denote bilinear group system generator which on input a security parameter κ, output $(p, \mathbb{G}, \mathbb{G}_T, e)$.

Assumption 2.1 (Computational Bilinear Diffie-Hellman Assumption (CBDH))**.** The CBDH assumption in bilinear group system $(p, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BLGroupGen}(\kappa)$ is that for any PPT adversary $\mathcal{A}$, it holds that:

$$\Pr[\mathcal{A}(g, g^x, g^y, g^z) = e(g, g)^{xyz}] \leq \mathsf{negl}(\kappa),$$

where the probability is taken over the choice of $g \xleftarrow{\text{R}} \mathbb{G}$, $x, y, z \xleftarrow{\text{R}} \mathbb{Z}_p$. Hereafter, we write $\vec{v}$ to denote a CBDH instance $(g, g^x, g^y, g^y) \in \mathbb{G}^4$. The decisional bilinear Diffie-Hellman (DBDH) assumption is that the two distributions (g, g^x, g^y, g^z, T_0) and (g, g^x, g^y, g^z, T_1) are computationally indistinguishable, where $T_0 \xleftarrow{\text{R}} \mathbb{G}_T$ and $T_1 = e(g, g)^{xyz}$.

Assumption 2.2 (n-one-more CBDH (omCBDH) Assumption)**.** The n-omCBDH assumption in bilinear group system $(p, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BLGroupGen}(\kappa)$ is that for any PPT adversary $\mathcal{A}$, it holds that:

$$\Pr[\mathcal{A}^{\mathsf{DL}_g(\cdot)}(g, \{g^{x_i}, g^{y_i}, g^{z_i}\}_{i=1}^{n+1}) = (\{e(g, g)^{x_i y_i z_i}\}_{i=1}^{n+1})] \leq \mathsf{negl}(\kappa),$$

where the probability is taken over the choices of $g \xleftarrow{\text{R}} \mathbb{G}$, and $x_i, y_i, z_i \xleftarrow{\text{R}} \mathbb{Z}_p$ for $i \in [n+1]$. To solve $n+1$ CBDH instances, $\mathcal{A}$ is allowed to query $\mathsf{DL}_g(\cdot)$ at most n times, where $\mathsf{DL}_g(\cdot)$ is a discrete logarithm oracle which outputs $t \in \mathbb{Z}_p$ on input $h = g^t$.

One may wonder whether this newly introduced problem is hard or not. Unfortunately, we are not able to directly prove its hardness in known models, such as the generic group model. The difficulty of showing its hardness is demonstrated in a recent work [ZZC$^+$14], which relies on concurrent rewinding technique to argue that no black-box reduction can be used to base its hardness on any weaker non-interactive cryptographic assumption.

Assumption 2.3 (n-Decisional Bilinear Diffie-Hellman Inversion Assumption (n-DBDHI))**.** The n-DBDHI assumption in bilinear group system $(p, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BLGroupGen}(\kappa)$ is that for any PPT adversary $\mathcal{A}$, it holds that:

$$|\Pr[\mathcal{A}(g, g^x, \ldots, g^{x^n}, T_\beta) = 1] - 1/2| \leq \mathsf{negl}(\kappa),$$

where $T_0 \xleftarrow{\text{R}} \mathbb{G}_T$, $T_1 = e(g, g)^{1/x} \in \mathbb{G}_T$, and the probability is taken over the choices of $g \xleftarrow{\text{R}} \mathbb{G}$, $x \xleftarrow{\text{R}} \mathbb{Z}_p$, and $\beta \xleftarrow{\text{R}} \{0, 1\}$.

As observed in [BB04a], the n-DBDHI assumption is equivalent to the n-DBDHI* assumption, which is identical to the standard one except that T_1 is set as $e(g, g)^{x^{2n+1}}$ instead of $e(g, g)^{1/x}$. We will, for notational convenience, base our proofs on the n-DBDHI* assumption in this work.

2.4 Identity-Based Non-Interactive Key Exchange

An identity-based non-interactive key exchange (IB-NIKE) scheme consists of the following polynomial-time algorithms:

- $\mathsf{Setup}(\kappa, n)$: on input a security parameter κ and a parameter n for the number of participants,[4] output master public key mpk and master secret key msk. Let I be the identity space and SHK be the shared key space.
- $\mathsf{Extract}(msk, id)$: on input msk and identity $id \in I$, output a secret key sk_{id} for id.
- $\mathsf{Share}(sk_{id}, \mathcal{I})$: on input secret key sk_{id} for identity id and an list $\mathcal{I} \in I^n$ consisting of n identities, output a shared key shk for $\mathcal{I}$. We assume that the identities in $\mathcal{I}$ are always lexicographically ordered.

[4] The second input is occasionally omitted when $n = 2$.

Correctness: For any $\kappa \in \mathbb{N}$ and $n \geq 2$, any $(mpk, msk) \leftarrow \mathsf{Setup}(\kappa, n)$, any list of n distinct identities $\mathcal{I} \in I^n$, any $id \in I$ and any $sk_{id} \leftarrow \mathsf{Extract}(msk, id)$, we require $\mathsf{Share}(sk_{id}, \mathcal{I})$ agree on a common group key $shk_{\mathcal{I}}$.

Security: We follow the security notions presented in [Hof14]. Let $\mathcal{A}$ be an adversary against IB-NIKE and define its advantage as:

$$\mathrm{Adv}_{\mathcal{A}}(\kappa) = \Pr\left[\beta = \beta' : \begin{array}{l} (mpk, msk) \leftarrow \mathsf{Setup}(\kappa, n); \\ \mathcal{I} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{extract}}(\cdot), \mathcal{O}_{\mathsf{reveal}}(\cdot)}(mpk); \\ shk_0^* \xleftarrow{\mathrm{R}} SHK, shk_1^* \leftarrow shk_{\mathcal{I}}; \\ \beta \xleftarrow{\mathrm{R}} \{0,1\}; \\ \beta' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{extract}}(\cdot), \mathcal{O}_{\mathsf{reveal}}(\cdot)}(shk_\beta^*); \end{array}\right] - \frac{1}{2},$$

where $\mathcal{O}_{\mathsf{extract}}(id) = \mathsf{Extract}(msk, id)$, $\mathcal{O}_{\mathsf{reveal}}(\mathcal{I}) = shk_{\mathcal{I}}$, and $\mathcal{A}$ is not allowed to query $\mathcal{O}_{\mathsf{extract}}(\cdot)$ for any identity $id \in \mathcal{I}$ and query $\mathcal{O}_{\mathsf{reveal}}(\cdot)$ for $\mathcal{I}$. We say that an IB-NIKE is adaptively secure if no PPT adversary has non-negligible advantage in the above security experiment. The adaptive security is the strongest security notion for IB-NIKE so far. We note that the selective security can be defined similarly as above by requiring the adversary to commit the target identity list $\mathcal{I}$ even before it seeing mpk, while the semi-adaptive security can be defined similarly above by discarding $\mathcal{O}_{\mathsf{reveal}}(\cdot)$.

3 Revisit Sakai-Ohgishi-Kasahara IB-NIKE

We begin this section by recalling the SOK IB-NIKE scheme [SOK00], which is given by the following three algorithms:

- $\mathsf{Setup}(\kappa, 2)$: run $\mathsf{BLGroupGen}(\kappa)$ to generate $(p, \mathbb{G}, \mathbb{G}_T, e)$, pick $\mathsf{H} : I \rightarrow \mathbb{G}$ as identity mapping function and $\mathsf{G} : \mathbb{G}_T \rightarrow \{0,1\}^n$ as shared key encoding function, pick $x \xleftarrow{\mathrm{R}} \mathbb{Z}_p$, $g \xleftarrow{\mathrm{R}} \mathbb{G}^*$, set $h = g^x$; output $mpk = (g, h, \mathsf{H}, \mathsf{G})$[5] and $msk = x$.
- $\mathsf{Extract}(msk, id)$: on input $msk = x$ and $id \in I$, output $sk_{id} \leftarrow \mathsf{H}(id)^x$.
- $\mathsf{Share}(sk_{id_a}, \mathcal{I})$: on input sk_{id_a} and $\mathcal{I} = (id_a, id_b)$ or $\mathcal{I} = (id_b, id_a)$, output $shk \leftarrow \mathsf{G}(e(sk_{id_a}, \mathsf{H}(id_b)))$.

We observe that in the SOK IB-NIKE, the secret key is "publicly checkable", which means there exists an efficient algorithm $\mathsf{SKCheck}$ which can check if sk is a valid secret key associated to id with respect to mpk. More precisely, in the SOK IB-NIKE, this algorithm can be constructed with the help of bilinear map as follows:

- $\mathsf{SKCheck}(mpk, sk, id)$: on input $mpk = (g, h, \mathsf{H}, \mathsf{G})$, a secret key sk, and an identity id, output "true" if $e(h, \mathsf{H}(id)) = e(g, sk)$ or "false" otherwise.

Theorem 3.1 ([PS09])**.** *The SOK IB-NIKE scheme is adaptively secure in the random oracle model assuming the CBDH assumption holds in bilinear group system generated by* $\mathsf{BLGroupGen}(\kappa)$. *Suppose* H *and* G *are random oracles, for any adversary* $\mathcal{A}$ *breaking the SOK IB-NIKE scheme with advantage* $\mathsf{Adv}_{\mathcal{A}}(\kappa)$ *that makes* q_h *and* q_g *times queries to random oracle* H *and* G *respectively, there is an algorithm* $\mathcal{B}$ *that solves the CBDH problem with advantage* $\mathsf{Adv}_{\mathcal{A}}(\kappa)/q_h^2 q_g$.

[5] We note that g and h are not used in the rest algorithms, but are essential in the augumented algorithm $\mathsf{SKCheck}$, which plays a crucial role in proving the negative result of the SOK IB-NIKE.

3.1 An Improved Proof for the SOK IB-NIKE

The original reduction [PS09] for the SOK IB-NIKE lose a factor of $1/q_h^2 q_g$. In this subsection, we show that adaptive security for the SOK scheme can be reduced to the CBDH problem with a tighter security reduction.

Theorem 3.2. *The SOK IB-NIKE scheme is adaptively secure in the random oracle model assuming the CBDH assumption holds in bilinear group system generated by* $\mathsf{BLGroupGen}(\kappa)$*. Suppose* H *and* G *are random oracles, for any adversary* $\mathcal{A}$ *breaking the SOK IB-NIKE scheme with advantage* $\mathsf{Adv}_{\mathcal{A}}(\kappa)$ *that makes at most* q_e *extraction queries and* q_r *reveal queries and* q_g *random oracle queries to* G*, there is an algorithm* $\mathcal{B}$ *that solves the CBDH problem with advantage* $4\mathsf{Adv}_{\mathcal{A}}(\kappa)/e^2(q_e+q_r)^2 q_g$*, where* e *is the natural logarithm.*

Proof. Given the CBDH instance $(g, g_1 = g^x, g_2 = g^y, g_3 = g^z)$ in bilinear group system $(p, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BLGroupGen}(\kappa)$, $\mathcal{B}$ interacts with $\mathcal{A}$ as follows:

- Setup: $\mathcal{B}$ sets $h = g^x$, then sets $mpk = (g, h, \mathsf{H}, \mathsf{G})$ and $msk = x$ (which is unknown to him), treats H and G as random oracles, sends mpk to $\mathcal{A}$.
- Random oracle queries: To process random oracle queries, $\mathcal{B}$ maintains two associated lists H and G. Each entry in H is of the form $(id, mark, coin, t, pk)$, where $id \in I$, $mark, coin \in \{0,1\}$, $t \in \mathbb{Z}_p$, and $pk \in \mathbb{G}$. Each entry in G is of the form (k, shk), where $k \in \mathbb{G}_T$ and $shk \in \{0,1\}^n$. Both of them are initially empty. When processing a random oracle query $\langle k \rangle$ to $\mathsf{G}(\cdot)$, $\mathcal{B}$ returns the corresponding shk value in the G list if it is defined, otherwise $\mathcal{B}$ initializes the entry by picking a random value shk from $\{0,1\}^n$, and then adds the entry (k, shk) to the G list and returns shk to $\mathcal{A}$ as $\mathsf{G}(k)$. When processing a random oracle query $\langle id \rangle$ to $\mathsf{H}(\cdot)$, $\mathcal{B}$ returns the corresponding value if it is defined, otherwise picks $t \xleftarrow{\text{R}} \mathbb{Z}_p$ and initializes the corresponding entry as follows:
  - with probability $(1-\delta)$ set $mark = 1$, then further pick a random unbiased coin, if $coin = 0$ then set $pk = g_2^t$, else set $pk = g_3^t$.
  - with probability δ set $mark = 0$, set $coin = \perp$ indicating undefined, and set $pk = g^t$.

  then adds the entry $(id, mark, coin, t, pk)$ to the H list and returns pk to $\mathcal{A}$ as $\mathsf{H}(id)$. The value of δ will be determined later. We note that the value of $mark$ is completely hidden from $\mathcal{A}$ since in either case, the responses to the H-queries are uniform and independent over $\mathbb{G}$. Hereafter, let $P : I \to \{0,1\}$ be a predicate and define $P(id) = 1$ if and only if the associated $mark = 1$.
- Phase 1: $\mathcal{A}$ can issue two types of queries:
  - Extraction queries: Let t be the associated value of id in the H list. If $P(id) = 0$, $\mathcal{B}$ computes $sk_{id} = g_1^t$ and sends it to $\mathcal{A}$. Else, $\mathcal{B}$ aborts and outputs a random bit.
  - Reveal queries: If $P(id_a) = 1 \wedge P(id_b) = 1$, $\mathcal{B}$ aborts and outputs a random bit. Else $\mathcal{B}$ picks one identity marked with 0, extracts its secret key, then computes the shared key and responds it to $\mathcal{A}$.
- Challenge: $\mathcal{A}$ outputs two distinct identities (id_a^*, id_b^*) with the restriction that either id_a^* or id_b^* has not been queried for secret key and (id_a^*, id_b^*) has not been queried for shared key. Let $(t_a, coin_a)$ and $(t_b, coin_b)$ be the associated value of id_a^* and id_b^* in the H list respectively. If $P(id_a^*) = 1 \wedge P(id_b^*) = 1 \wedge coin_a \neq coin_b$, $\mathcal{B}$ returns a random string from $\{0,1\}^n$ as the challenge. Else, $\mathcal{B}$ aborts.
- Phase 2: $\mathcal{A}$ can continue to issue extraction queries and reveal queries as in Phase 1, except that extraction queries for id_a^*, id_b^* and reveal query for (id_a^*, id_b^*) will be denied.
- Guess: $\mathcal{A}$ outputs its guess β' for β.

At the end of the simulation, $\mathcal{B}$ picks an entry (k, shk) randomly from the G list and computes $k^{t_a^{-1}t_b^{-1}}$ as the solution to the CBDH instance. If $k = e(g_2^{t_a}, g_3^{t_b})^x = e(g,g)^{xyzt_at_b}$, then $k^{t_a^{-1}t_b^{-1}}$ is exactly the desired CBDH solution. It is easy to see that conditioned on $\mathcal{B}$ does not abort, $\mathcal{A}$'s view in the above game is identical to the real IB-NIKE security game. Let F be the event that $\mathcal{B}$ does not abort, we have $\mathsf{Adv}_{\mathcal{B}}(\kappa) = \Pr[F] \cdot 2\mathsf{Adv}_{\mathcal{A}}(\kappa)/q_g$. In what follows, we compute the low bound of $\Pr[F]$. Let $\{id_i\}_{1 \le i \le q_e}$ be q_e distinct extraction queries, $\{(id_{j,1}, id_{j,2})\}_{1 \le j \le q_r}$ be q_r distinct reveal queries. To ease the analysis, we further define the following events:

$$\begin{aligned} F_1 &: \ \textstyle\bigwedge_{i=1}^{q_e}(P(id_i) = 0) \\ F_2 &: \ \textstyle\bigwedge_{j=1}^{q_r}(P(id_{j,1}) = 0 \vee P(id_{j,2}) = 0) \\ F_3 &: \ P(id_a^*) = 1 \wedge P(id_b^*) = 1 \wedge coin_a \neq coin_b \end{aligned}$$

Obviously, we have $F = F_1 \wedge F_2 \wedge F_3$. Therefore, we have:

$$\Pr[F] = \Pr[F_1] \cdot \Pr[F_2 \wedge F_3 \mid F_1]$$

Since each coin toss for *mark* is independent, we have $\Pr[F_1] = \delta^{q_e}$. Note that in each reveal query there exists at least one identity different from both id_a^* and id_b^*, and the choice of $coin_a$ and $coin_b$ are random and independent, then we have $\Pr[F_2 \wedge F_3 \mid F_1] \ge \delta^{q_r}(1-\delta)^2/2$. Finally, we arrive at $\Pr[F] \ge \delta^{q_e+q_r}(1-\delta)^2/2$. Let $f(\delta) = \delta^{q_e+q_r}(1-\delta)^2/2$. This function achieves the maximum value at the zero point $1 - 2/(q_e + q_r + 2)$ of $f'(\delta)$, therefore we have:

$$\Pr[F] \ge \frac{2}{(q_e+q_r)^2} \cdot \left(1 - \frac{2}{q_e+q_r+2}\right)^{q_e+q_r+2}.$$

According to the estimation that $\lim_{x\to 0}(1+x)^{\frac{1}{x}} = e$, the maximum value of the lower bound for $\Pr[F]$ is approximate $2/e^2(q_e+q_r)^2$. According to the estimation [BR96] that $q_e \approx 2^{30}$, $q_r \approx 2^{30}$, and $q_g \approx 2^{60}$, the overall reduction roughly lose a factor of $1/2^{120}$. □

3.2 SOK IB-NIKE is not Provably Secure Under NPROM

We now show that the SOK IB-NIKE scheme can not be proven secure without programming the random oracle with respect to a slightly restricted type of reductions, which is called *single-instance* reduction in [FF13]. In the case of identity-based schemes (including IBE, IBS as well as IB-NIKE), the restrictions lie at such a type of reductions can only invoke a single instance of the adversary and, can not rewind the adversary to a point before it hands over the master public key for the first time.

Theorem 3.3 (Non-Programming Irreducibility for SOK IB-NIKE)**.** *Assume the 1-omCBDH assumption holds in bilinear group system generated by* $\mathsf{BLGroupGen}(\kappa)$*, then there exists no non-programming single-instance fully-black-box reduction that reduces the adaptive security of SOK IB-NIKE to the CBDH problem. More precisely, assume there exists such a reduction* $\mathcal{R}$ *that converts any adversary* $\mathcal{A}$ *against the SOK IB-NIKE into an algorithm against the CBDH problem. Assume further that the reduction* $\mathcal{R}$ *has success probability* $\mathsf{Succ}_{\mathcal{R}^{\mathcal{A}}}^{\mathrm{CBDH}}$ *for given* $\mathcal{A}$ *and runtime* $\mathsf{Time}_{\mathcal{R}}(\kappa)$*. Then, there exists a family* $\mathbb{A}$ *of successful (but possibly inefficient) adversaries* $\mathcal{A}_{\mathcal{R},a}$ *against adaptive security of SOK IB-NIKE and a meta-reduction* $\mathcal{M}$ *that breaks the 1-omCBDH assumption with non-negligible success probability* $\mathsf{Succ}_{\mathcal{M}}^{\text{1-omCBDH}}(\kappa) \ge (\mathsf{Succ}_{\mathcal{R}^{\mathcal{A}_{\mathcal{R},a}}}^{\mathrm{CBDH}}(\kappa))^2$ *for a random* $\mathcal{A}_{\mathcal{R},a} \in \mathbb{A}$ *and runtime* $\mathsf{Time}_{\mathcal{M}}(\kappa) = 2 \cdot \mathsf{Time}_{\mathcal{R}}(\kappa) + \mathsf{poly}(\kappa)$*.*

Proof. We prove this theorem using meta-reduction technique summarized in [Fis12]. We show that if such black-box reduction $\mathcal{R}$ exists, then we can build a reduction against the reduction (meta-reduction $\mathcal{M}$) to the 1-omCBDH problem. Briefly, we first show the existence of a successful adversary $\mathcal{A}$ against the SOK IB-NIKE scheme by building an inefficient adversary which succeeds using its unbounded computation power. We then show how to build a meta-reduction $\mathcal{M}$ to simulate this specific adversary in order to turn $\mathcal{R}$ in a black-box manner into an efficient and successful algorithm $\mathcal{M}^{\mathcal{R}}$ against the 1-omCBDH problem. We describe such inefficient adversary $\mathcal{A}$ and meta-reduction $\mathcal{M}$ in details as below.

A Family of Adversaries $\mathcal{A}_{\mathcal{R},a}$. We first describe an unbounded adversary $\mathcal{A}$, depicted in Figure 3.2, which depends on the reduction $\mathcal{R}$. For ease of exposition, we will think of the adversary as a family $\mathbb{A}$ of adversaries $\mathcal{A}_{\mathcal{R},a}$ depending on the reduction $\mathcal{R}$ and some randomness a. We will make these dependence implicit when it is clear from the context. Let A be defined as the set $I^3 \times \mathbb{G}^4 \times \{0,1\}^{\mathsf{poly}(\kappa)}$. For every $a = (id, id_c, id_d, \vec{v}, \bar{\omega}) \in A$ (where $\vec{v}$ is an instance of the CBDH problem and $\bar{\omega}$ is a random tape) and every reduction $\mathcal{R}$ we define the adversary $\mathcal{A}_{\mathcal{R},a}$ as below.

1. If the three identities id, id_a, id_b are not distinct, $\mathcal{A}_{\mathcal{R},a}$ aborts.
2. $\mathcal{A}_{\mathcal{R}}$ receives a master public key mpk from $\mathcal{R}$.
3. $\mathcal{A}_{\mathcal{R}}$ issues the extraction query for id to $\mathcal{R}$. Upon receiving $sk_{id} \leftarrow \mathcal{R}.\mathsf{Extract}(id)$, $\mathcal{A}_{\mathcal{R}}$ verifies its validity by running $\mathsf{SKCheck}(mpk, sk_{id}, id)$. If $\mathcal{R}$ is unable to provide a valid secret key, $\mathcal{A}$ aborts.
4. $\mathcal{A}_{\mathcal{R}}$ invokes an internal copy of $\mathcal{R}$ (denoted $\mathcal{R}^*$ hereafter) on input $\vec{v}$ and random tape $\bar{\omega}$, then interacts with $\mathcal{R}^*$ as below:
   (a) $\mathcal{A}_{\mathcal{R}}$ receives a master public key mpk^* from $\mathcal{R}^*$.
   (b) $\mathcal{A}_{\mathcal{R}}$ issues the extraction query for id_a to $\mathcal{R}^*$. After receiving secret key $sk^*_{id_a} \leftarrow \mathcal{R}^*.\mathsf{Extract}(id_a)$, $\mathcal{A}_{\mathcal{R}}$ terminates the interaction with $\mathcal{R}^*$, and then verifies its validity by testing $\mathsf{SKCheck}(mpk, sk_{id}, id)$. If $\mathcal{R}^*$ is unable to provide a valid secret key, $\mathcal{A}_{\mathcal{R}}$ aborts. Else, $\mathcal{A}_{\mathcal{R}}$ forwards all random oracle queries issued by $\mathcal{R}^*$ to the external random oracles.
5. $\mathcal{A}_{\mathcal{R}}$ submits (id_a, id_b) to $\mathcal{R}$ as the target identities, and then receives shk_β from $\mathcal{R}$ as the challenge. To determine if $\beta = 0$ (indicating $shk_\beta \xleftarrow{\text{R}} SHK$) or $\beta = 1$ (indicating $shk_\beta = \mathsf{G}(e(\mathsf{H}(id_a), \mathsf{H}(id_b))^{msk})$, $\mathcal{A}_{\mathcal{R}}$ exhaustively searches $\delta \in \mathbb{Z}_p$ such that $mpk^* \cdot mpk^{-1} = g^\delta$ (i.e., $\delta = msk^* - msk \bmod p$). Given the difference of the master secret keys, $\mathcal{A}_{\mathcal{R}}$ adapts $sk^*_{id_a}$ to mpk by computing $sk_{id_a} := sk^*_{id_a} \cdot g^{-\delta}$ and then computes $shk \leftarrow \mathsf{Share}(sk_{id_a}, id_b)$.
6. Finally, $\mathcal{A}_{\mathcal{R}}$ sets $\beta' := (shk \overset{?}{=} shk_\beta)$ and outputs β' to $\mathcal{R}$.

Observe that for any reduction $\mathcal{R}$, $\mathcal{A}_{\mathcal{R}}$ is successful with at least the same probability with which an instance of $\mathcal{R}$ is able to produce a valid secret key for a randomly chosen identity (minus a negligible probability for the three identities are not distinct).

Description of $\mathcal{M}$. We then describe the meta-reduction $\mathcal{M}$, depicted in Figure 2, which on input $\vec{v}_0, \vec{v}_1$ invokes two instances of $\mathcal{R}$ with independent random tapes. The first reduction $\mathcal{R}_0$ gets as input $\vec{v}_0$. The second reduction $\mathcal{R}_1$ gets as input $\vec{v}_1$. All random oracle queries issued by either $\mathcal{R}_0$ or $\mathcal{R}_1$ are answered by forwarding to the external random oracles and returning the answers. Both reduction instances can now invoke an adversary instance $\mathcal{A}$ at most once. To simulate $\mathcal{A}$ for each copy, $\mathcal{M}$ interacts with $\mathcal{R}_0$ and $\mathcal{R}_1$ as follows:

1. $\mathcal{M}$ chooses $id_a, id_b, id_c, id_d \xleftarrow{\text{R}} I$. If id_a, id_c, id_d are not distinct or id_c, id_a, id_b are not distinct, then $\mathcal{M}$ aborts.
2. $\mathcal{M}$ receives the master public key mpk_0 (resp. mpk_1) from $\mathcal{R}_0$ (resp. $\mathcal{R}_1$).

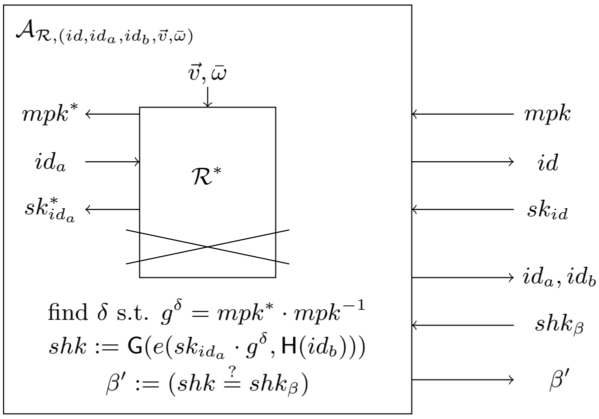
$\mathcal{A}_{\mathcal{R},(id,id_a,id_b,\vec{v},\bar{\omega})}$
$\vec{v},\bar{\omega}$
mpk^*
id_a
$sk^*_{id_a}$
$\mathcal{R}^*$
find δ s.t. $g^\delta = mpk^* \cdot mpk^{-1}$
$shk := \mathsf{G}(e(sk_{id_a} \cdot g^\delta, \mathsf{H}(id_b)))$
$\beta' := (shk \overset{?}{=} shk_\beta)$
mpk
id
sk_{id}
id_a, id_b
shk_β
β'

Figure 1: For each reduction $\mathcal{R}$, the associated inefficient adversary $\mathcal{A}_{\mathcal{R}}$ internally invokes an instance of $\mathcal{R}$ and uses its unbounded computational power to adapt the obtained secret key for id_a under mpk^* to a secret key under mpk, then submits (id_a, id_b) as the target identities and outputs its guess β' for β using the adapting secret key.

3. $\mathcal{M}$ issues extraction query for id_c (resp. id_a) to $\mathcal{R}_0$ (resp. $\mathcal{R}_1$). Upon receiving $sk_{id_c} \leftarrow \mathcal{R}_0.\mathsf{Extract}(id_c)$ and $sk_{id_a} \leftarrow \mathcal{R}_1.\mathsf{Extract}(id_a)$, $\mathcal{M}$ verifies the validness of both secret keys following the same approach used by $\mathcal{A}_{\mathcal{R}}$ as described before. If either $\mathcal{R}_0$ or $\mathcal{R}_1$ is unable to produce a valid secret key, $\mathcal{M}$ aborts. Else, let $Q_{\mathsf{RO},0}$ (resp. $Q_{\mathsf{RO},1}$) be the sequence of random oracle queries issued by $\mathcal{R}_0$ (resp. $\mathcal{R}_1$) up to now, $\mathcal{M}$ queries $Q_{\mathsf{RO},0}$ (resp. $Q_{\mathsf{RO},1}$) to the random oracle interface provided by $\mathcal{R}_1$ (resp. $\mathcal{R}_0$) to emulate the same hash queries the adversary instance for each reduction would issue. This operation is necessary to make sure $\mathcal{M}$ issues exactly the same random oracle queries as the specific $\mathcal{A}$ depicted in Figure 3.2, since we are working in the random oracle model, and thus the instances of $\mathcal{R}$ expect to see all the random oracle queries, including the ones issued by (simulated) adversary.
4. $\mathcal{M}$ submits id_c, id_d (resp. (id_a, id_b)) to $\mathcal{R}_0$ (resp. $\mathcal{R}_1$) as the target identities.
5. $\mathcal{M}$ receives challenge $s\mathring{h}k_\beta$ (resp. $s\ddot{h}k_\gamma$) from $\mathcal{R}_0$ (resp. $\mathcal{R}_1$). $\mathcal{M}$ queries $\delta \leftarrow \mathsf{DL}_g(mpk_0 \cdot mpk_1)$ and adapts secret keys $\tilde{s}k_{id_a} := sk_{id_a} \cdot g^\delta$ and $\hat{s}k_{id_c} := sk_{id_c} \cdot g^{-\delta}$, then computes $s\hat{h}k \leftarrow \mathsf{Share}(\hat{s}k_{id_c}, id_d)$ and $s\tilde{h}k \leftarrow \mathsf{Share}(\tilde{s}k_{id_a}, id_b)$.
6. $\mathcal{M}$ sets $\beta' := (s\tilde{h}k \overset{?}{=} s\mathring{h}k_\beta)$ and $\gamma' := (s\hat{h}k \overset{?}{=} s\ddot{h}k_\gamma)$, then returns β' (resp. γ') to $\mathcal{R}_0$ (resp. $\mathcal{R}_1$).

As aforementioned, we focus on single-instance reductions. This type of reductions are only allowed to invoke one adversary instance and forbidden to rewind the adversary to a point before handing the master public key. If $\mathcal{R}_0$ (resp. $\mathcal{R}_1$) tries to rewind $\mathcal{A}_0$ (resp. $\mathcal{A}_1$), $\mathcal{M}$ will keeping on querying id_c (resp. id_a), issuing $Q_{\mathsf{RO},1}$ (resp. $Q_{\mathsf{RO},0}$), submitting (id_a, id_b) (resp. (id_c, id_d)) as the target identities, and outputting β' (resp. γ') as the answer.

At the end of the simulation, if both $\mathcal{R}_0$ and $\mathcal{R}_1$ output their candidate solution T_0 and T_1, $\mathcal{M}$ forwards T_0 and T_1 to the 1-omCBDH challenger as its solution. Else, $\mathcal{M}$ reports failure.

Success Probability. Before calculating the success probability of $\mathcal{M}$ against the 1-omCBDH problem, we first argue the correctness of the adversary simulation.

Claim 3.1. *$\mathcal{M}$ and $\mathcal{A}_{\mathcal{R}}$ are perfectly indistinguishable in the view of $\mathcal{R}$.*

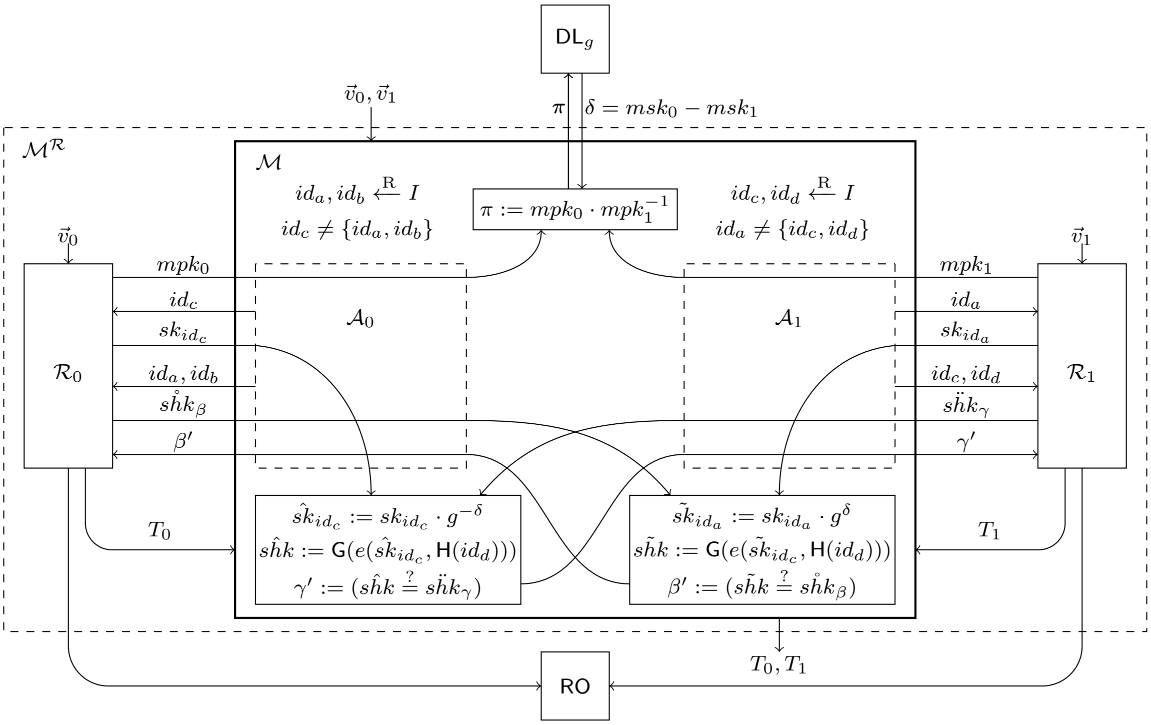
DL_g
π
$\delta = msk_0 - msk_1$
$\vec{v}_0, \vec{v}_1$
$\mathcal{M}^{\mathcal{R}}$
$\mathcal{M}$
$id_a, id_b \xleftarrow{\mathrm{R}} I$
$id_c \neq \{id_a, id_b\}$
$\pi := mpk_0 \cdot mpk_1^{-1}$
$id_c, id_d \xleftarrow{\mathrm{R}} I$
$id_a \neq \{id_c, id_d\}$
$\vec{v}_0$
$\vec{v}_1$
mpk_0
id_c
sk_{id_c}
id_a, id_b
$\mathring{shk}_\beta$
β'
$\mathcal{R}_0$
$\mathcal{A}_0$
$\mathcal{A}_1$
mpk_1
id_a
sk_{id_a}
id_c, id_d
$\ddot{shk}_\gamma$
γ'
$\mathcal{R}_1$
$\hat{sk}_{id_c} := sk_{id_c} \cdot g^{-\delta}$
$\hat{shk} := \mathsf{G}(e(\hat{sk}_{id_c}, \mathsf{H}(id_d)))$
$\gamma' := (\hat{shk} \overset{?}{=} \ddot{shk}_\gamma)$
$\tilde{sk}_{id_a} := sk_{id_a} \cdot g^{\delta}$
$\tilde{shk} := \mathsf{G}(e(\tilde{sk}_{id_c}, \mathsf{H}(id_d)))$
$\beta' := (\tilde{shk} \overset{?}{=} \mathring{shk}_\beta)$
T_0
T_1
T_0, T_1
RO

Figure 2: The meta-reduction uses two instances of $\mathcal{R}$ and simulates the adversary $\mathcal{A}$ by obtaining the difference between the master secret keys and adapting the secret key under one master public key output by $\mathcal{R}$ to a secret key under the other master public key, respectively.

Proof. Observe that both $\mathcal{M}$ and $\mathcal{A}_{\mathcal{R}}$ interact with the reduction with identical transcript. In details, on input a master public key from $\mathcal{R}$, both algorithms query a secret key for a randomly chosen identity, and issue exactly the random oracle queries needed to verify the received secret key. Then they submit another two random identities as the target identities. Upon receiving the challenge from $\mathcal{R}$, both algorithms invoke an instance of $\mathcal{R}$ on a random CBDH instance and an independent random tape, then proceed to query the secret key for the first target identity and issue the random oracle queries needed to verify the received secret key, then continue to adapt the secret key to the master public key received as input using the difference of the master secret keys, issue the associated random oracle queries which are needed to compute the target shared key. Finally, they both output the guess. When being rewinded, the behavior of both algorithms remains the same. □

Therefore, $\mathcal{M}$ perfectly mimics $\mathcal{A}_{\mathcal{R}}$ and thus $\mathsf{Succ}^{\mathrm{CBDH}}_{\mathcal{R}^{\mathcal{M}}}(\kappa) = \mathsf{Succ}^{\mathrm{CBDH}}_{\mathcal{R}^{\mathcal{A}_{\mathcal{R}}}}(\kappa)$. As mentioned before, according to $\mathcal{M}$'s strategy, it succeeds whenever both the two instances of reduction are successful. Thereby, we have $\mathsf{Succ}^{\text{1-omCBDH}}_{\mathcal{M}}(\kappa) = (\mathsf{Succ}^{\mathrm{CBDH}}_{\mathcal{R}^{\mathcal{M}}}(\kappa))^2 = (\mathsf{Succ}^{\mathrm{CBDH}}_{\mathcal{R}^{\mathcal{A}_{\mathcal{R}}}}(\kappa))^2$. According to the assumption of this theorem, the reduction $\mathcal{R}$ here must succeed to solve the CBDH problem with some non-negligible probability given any (black-box) adversary. Therefore, $\mathsf{Succ}^{\mathrm{CBDH}}_{\mathcal{R}^{\mathcal{A}_{\mathcal{R}}}}(\kappa)$ is non-negligible, so is $\mathsf{Succ}^{\text{1-omCBDH}}_{\mathcal{M}}(\kappa)$.

Running Time. The time overhead of $\mathcal{M}$ consists of two executions of $\mathcal{R}$, several random oracle queries, and a constant number of modular inversions, multiplications, additions, and pairings. Therefore, we have: $\mathsf{Time}_{\mathcal{M}}(\kappa) = 2 \cdot \mathsf{Time}_{\mathcal{R}}(\kappa) + \mathsf{poly}(\kappa)$.

This completes the proof. □

Remark 3.1. In the above simulation, we implicitly assume that the reduction $\mathcal{R}_b$ will use the attacking ability of the simulated adversary $\mathcal{A}_b$. However, if $\mathcal{R}_b$ never invokes $\mathcal{A}_b$ or directly outputs a solution, the meta-reduction $\mathcal{M}$ then would have one of the solutions to the instances $\vec{v}_0$ and $\vec{v}_1$ without invoking the oracle of DL_g. Therefore, it could just abort the other reduction instance, solve the other CBDH instance by querying the oracle DL_g, and output the solutions. In the following, we omit this simple case and assume the reductions rely on the attacking ability of the adversary.

Remark 3.2. Note that the meta-reduction shown in the above proof implicitly exploits the fact that the SOK IB-NIKE scheme is defined with respect to a fixed and instance-independent identity hash function. This fact ensures that one identity id maps to the same "public-key" $\mathsf{H}(id)$ in different simulations, which is crucial for the secret key adapting trick. Observe that in the programming random oracle model, one identity id may corresponds to different public key since the programming of H might be different. Therefore, it is not straightforward to extend our black-box separation result in the programming random oracle model.

4 IB-NIKE from Indistinguishability Obfuscation

4.1 Warmup: Selectively Secure IB-NIKE from $i\mathcal{O}$

As a warmup, we show how to create a replacement for the RO $\mathsf{H}(\cdot)$ in the SOK scheme from puncturable PRFs and $i\mathcal{O}$. The resulting scheme is selective-secure in the standard model.

Selectively Secure Construction from $i\mathcal{O}$

- $\mathsf{Setup}(\kappa)$: run $\mathsf{BLGroupGen}(\kappa)$ to generate $(p, \mathbb{G}, \mathbb{G}_T, e)$, pick $x \xleftarrow{\text{R}} \mathbb{Z}_p$ and $g \xleftarrow{\text{R}} \mathbb{G}^*$; pick a secret key k for puncturable PRF $\mathsf{F} : I \to \mathbb{Z}_p$; then create an obfuscation of the program H shown in Figure 3. The size of the program is padded to be the maximum of itself and the program H^* shown in Figure 4. We refer to the obfuscated program as the function $\mathsf{H} : I \to \mathbb{G}$, which acts as the random oracle type hash function in the SOK scheme. The msk is x, whereas mpk is the hash function $\mathsf{H}(\cdot)$.
- Algorithm $\mathsf{Extract}$ and Share are identical to that in the SOK scheme.

Selective Hash H

Constants: Punctured PRF key k, $g \in \mathbb{G}^*$.

Input: Identity *id*.

1. Output $g^{\mathsf{F}_k(id)}$.

Figure 3: Selective Hash H

Theorem 4.1. *The above IB-NIKE construction is selective-secure if the obfuscation scheme is indistinguishably secure,* F *is a secure punctured PRF, and the DBDH assumption holds.*

Proof. We organize the proof as a sequence of hybrid games, where the first game corresponds to selective security game. We prove that any two successive games are computationally indistinguishable. Then, we show that any PPT adversary that succeeds with non-negligible probability in the final game can be used to break the DBDH assumption.

Selective Hash H^*

```
Constants: Punctured PRF key $k(S)$ for $S = \{id_a^*, id_b^*\}$, $id_a^*, id_b^* \in I$, $z_1^*, z_2^* \in \mathbb{G}$, $g \in \mathbb{G}^*$.
Input: Identity $id$.
  1. If $id = id_a^*$ output $z_1^*$ and exit.
  2. If $id = id_b^*$ output $z_2^*$ and exit.
  3. Else output $g^{\mathsf{F}_{k(S)}(id)}$.
```

Figure 4: Selective Hash H^*

Game 0: This game is identical to standard selective security game played between adversary $\mathcal{A}$ and challenger $\mathcal{CH}$:

- Initialize: $\mathcal{A}$ commits the target identities (id_a^*, id_b^*) to $\mathcal{CH}$.
- Setup: $\mathcal{CH}$ runs $\mathsf{BLGroupGen}(\kappa)$ to generate bilinear group system, $(p, \mathbb{G}, \mathbb{G}_T, e)$ picks $x \xleftarrow{\text{R}} \mathbb{Z}_p$ as msk, sets $h = g^x$, picks a secret key k for the puncturable PRF, then creates hash function H as obfuscations of the program Selective Hash H shown in Figure 3, sends $mpk = (h, \mathsf{H})$ to $\mathcal{A}$.
- Phase 1: $\mathcal{A}$ can issue the following two types of queries:
  - extraction query $\langle id \rangle \neq \langle id_a^* \rangle, \langle id_b^* \rangle$: $\mathcal{CH}$ responds with $sk_{id} = \mathsf{H}(id)^x$.
  - reveal query $\langle id_a, id_b \rangle \neq \langle id_a^*, id_b^* \rangle, \langle id_b^*, id_a^* \rangle$: $\mathcal{CH}$ first extracts sk_{id_a} for id_a, then responds with $\mathsf{Share}(sk_{id_a}, id_b)$.
- Challenge: $\mathcal{CH}$ picks $shk_0^* \xleftarrow{\text{R}} \mathbb{G}_T$ and computes $shk_1^* \leftarrow \mathsf{Share}(sk_{id_a^*}, id_b^*)$. $\mathcal{CH}$ picks $\beta \xleftarrow{\text{R}} \{0, 1\}$, then sends shk_β^* to $\mathcal{A}$ as the challenge.
- Phase 2: $\mathcal{A}$ can continue to issue the extraction queries and the reveal queries, $\mathcal{CH}$ proceeds the same way as in Phase 1.
- Guess: $\mathcal{A}$ outputs its guess β' and wins if $\beta = \beta'$.

Game 1: same as Game 0 except that we let $z_1^* = g^{\mathsf{F}_k(id_a^*)}$ and $z_2^* = g^{\mathsf{F}_k(id_b^*)}$, and create hash functions H as obfuscation of the program Selective Hash H^* shown in Figure 4.

Game 2: same as Game 1 except that for $i \in \{1, 2\}$ we let $z_i^* = g^{t_i^*}$ for t_i^* chosen uniformly at random in $\mathbb{Z}_p$.

Lemma 4.1. *Game 0 and Game 1 are computationally indistinguishable if the underlying obfuscation scheme is indistinguishable secure.*

Proof. We show the computational indistinguishability between Game 0 and Game 1 by giving a reduction to the indistinguishability security of the obfuscator. More precisely, suppose there is a PPT adversary $\mathcal{A}$ can distinguish Game 0 and Game 1, then we can build algorithms $(\mathcal{S}, \mathcal{D})$ against the indistinguishability of the obfuscator by interacting with $\mathcal{A}$ as follows.

Sample: $\mathcal{S}$ invokes adversary $\mathcal{A}$ in selective security game for IB-NIKE. $\mathcal{A}$ commits the target identities (id_a^*, id_b^*) to $\mathcal{S}$. $\mathcal{S}$ sets $S = \{id_a^*, id_b^*\}$, picks $g \xleftarrow{\text{R}} \mathbb{G}^*$, chooses a secret key k for the puncturable PRF F, sets $z_1^* = g^{\mathsf{F}_k(id_a^*)}$ and $z_2^* = g^{\mathsf{F}_k(id_b^*)}$, then builds C_0 as the program of Selective Hash H, and C_1 as the program of Selective Hash H^*. Finally, $\mathcal{S}$ sets $\tau = (id_a^*, id_b^*, k)$. Before describing $\mathcal{D}$, we observe that by construction and the functionality preserving property of puncturable PRFs, the circuits C_0 and C_1 always behave identically on every input. After

padding, both C_0 and C_1 have the same size. Thus, $\mathcal{S}$ satisfies the conditions needed for invoking the indistinguishability property of the obfuscator. Now, we can describe the algorithm $\mathcal{D}$, which takes as input τ as given above, and the obfuscation of either C_0 or of C_1.
Distinguish: $\mathcal{D}$ picks $x \in \mathbb{Z}_p$ as msk, sets $h = g^x$, and creates mpk by including C_b with it. It then sends mpk to $\mathcal{A}$. When $\mathcal{A}$ issues extraction queries and reveal queries, $\mathcal{D}$ responds with msk. If $\mathcal{A}$ wins, $\mathcal{D}$ outputs 1.

By construction, if $\mathcal{D}$ receives an obfuscation of C_0, then the probability that $\mathcal{D}$ outputs 1 is exactly the probability of $\mathcal{A}$ winning in Game 0. On the other hand, if $\mathcal{D}$ receives an obfuscation of C_1, then the probability that $\mathcal{D}$ outputs 1 is the probability of $\mathcal{A}$ winning in Game 1. The indistinguishability of the obfuscator implies Game 0 and Game 1 are computationally indistinguishable. The lemma immediately follows. □

Lemma 4.2. *Game 1 and Game 2 are computationally indistinguishable if the underlying puncturable PRF is secure.*

Proof. We prove the computational indistinguishability between Game 1 and Game 2 by giving a reduction to the security of the puncturable PRFs. We build an adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ against puncturable PRFs as follows. $\mathcal{B}_1$ invokes $\mathcal{A}$ to obtain id_a^* and id_b^*, outputs a set $S = \{id_a^*, id_b^*\}$ and state τ. $\mathcal{B}_2$ receives a punctured key $k(S)$ for S and a challenge value t_1^*, t_2^*, where t_1^* (resp. t_2^*) is either $\mathsf{F}_k(id_a^*)$ (resp. $\mathsf{F}_k(id_b^*)$) or a uniformly random value from $\mathbb{Z}_p$. Then, $\mathcal{B}_2$ picks $x \xleftarrow{\text{R}} \mathbb{Z}_p$ as msk, picks $g \xleftarrow{\text{R}} \mathbb{G}$, sets $h = g^x$, and computes $z_1^* = g^{t_1^*}$ and $z_2^* = g^{t_2^*}$, produces obfuscation of Selective Hash H^* with $k(S)$, id_a^*, id_b^*, z_1^*, z_2^*, and g, then executes $\mathcal{A}$ and answers its extraction queries and reveal queries with msk. Finally, $\mathcal{B}_2$ outputs 1 if $\mathcal{A}$ succeeds. By construction, if $t_1^* = \mathsf{F}_k(id_a^*)$ and $t_2^* = \mathsf{F}_k(id_b^*)$, then $\mathcal{A}$'s view is identical to Game 1; else if $t_1^*, t_2^* \xleftarrow{\text{R}} \mathbb{Z}_p$, then $\mathcal{A}$'s view is identical to Game 2. The security for puncturable PRFs implies Game 1 and Game 2 are computationally indistinguishable. The lemma immediately follows. □

Lemma 4.3. *If the DBDH problem is hard, then the advantage of any PPT adversary in Game 2 is negligible.*

Proof. We prove this lemma by giving a reduction to the hardness of the DBDH problem. Suppose there exists an adversary $\mathcal{A}$ that has non-negligible advantage in Game 2, then we can build an algorithm $\mathcal{B}$ that has non-negligible advantage against the DBDH problem. Given the DBDH challenge instance $(g, g^x, g^y, g^z, T_\beta)$, $\mathcal{B}$ interacts with $\mathcal{A}$ as follows:

- Initialize: $\mathcal{B}$ invokes $\mathcal{A}$ to obtain the target identities (id_a^*, id_b^*).
- Setup: $\mathcal{B}$ sets $msk = x$, picks $g \xleftarrow{\text{R}} \mathbb{G}^*$, sets $z_1^* = g^y$, $z_2^* = g^z$, picks $k \xleftarrow{\text{R}} \mathbb{Z}_p$ as the secret key for puncturable PRF, computes $k(S) \leftarrow \text{PRF}.\mathsf{Puncture}(k, S)$ for $S = (id_a^*, id_b^*)$. $\mathcal{B}$ then produces the obfuscation program of H^* uses $k(S)$, id_a^*, id_b^*, z_1^*, z_2^*, and g.
- Phase 1: $\mathcal{A}$ can issue the following two types of queries:
  - extraction query $\langle id \rangle$: since $id \neq id_a^*, id_b^*$, $\mathcal{B}$ first computes $\mathsf{F}_{k(S)}(id)$ using $k(S)$, then responds with $sk_{id} = (g^x)^{\mathsf{F}_{k(S)}(id)}$.
  - reveal query $\langle id_a, id_b \rangle$: since among (id_a, id_b) there exists at least one identity different from both id_a^* and id_b^*, thus $\mathcal{B}$ can extract a secret key for this identity and then compute the shared key.
- Challenge: $\mathcal{B}$ sends T_β to $\mathcal{A}$ as the challenge.
- Phase 2: the same as in Phase 1.
- Guess: When $\mathcal{A}$ outputs its guess β', $\mathcal{B}$ forwards β' to its own challenger.

According to the construction of $\mathcal{B}$, the probability that $\mathcal{B}$ solves the DBDH problem is exactly the probability that $\mathcal{A}$ succeeds in Game 2. The lemma follows. □

These three lemmas together yield our main theorem that the above IB-NIKE is selectively secure. □

4.2 Main Result: Adaptively Secure IB-NIKE from $i\mathcal{O}$

We now show how to create a replacement for the RO $\mathsf{H}(\cdot)$ in the SOK IB-NIKE scheme from $(\mathsf{poly}, 2)$-AHF and $i\mathcal{O}$ to attain adaptive security in the standard model. We first recap the definition of AHF and present a generic construction of $(\mathsf{poly}, 2)$-AHF.

Admissible Hash Functions. Our definition below is a generalization of "admissible hash function"(AHF) [BB04b, CHKP10, FHPS13].

Definition 4.1 (AHF)**.** Let ℓ and θ be efficiently computable univariate polynomials of κ. For an efficiently computable function $\mathsf{AHF} : X \to Y^\ell$, define the predicate $P_u : X \to \{0,1\}$ for any $u \in (Y \cup E)^\ell$ as $P_u(x) = 0 \iff \forall i : \mathsf{AHF}(x)_i \neq u_i$, where $Y \cap E = \emptyset$, $\mathsf{AHF}(x)_i$ and u_i denote the i-th component of $\mathsf{AHF}(x)$ and u respectively. We say that AHF is (m,n)-admissible if there exists a PPT algorithm $\mathsf{AdmSample}$ and a polynomial $\theta(\kappa)$, such that for all $x_1, \dots, x_m, z_1, \dots, z_n \in X$, where $x_i \neq z_j$ for all $1 \leq i \leq m$ and $1 \leq j \leq n$, we have that:

$$\Pr[P_u(x_1) = \cdots = P_u(x_m) = 1 \wedge P_u(z_1) = \cdots = P_u(z_n) = 0] \geq 1/\theta(\kappa) \tag{1}$$

where the probability is over the choice of $u \leftarrow \mathsf{AdmSample}(\kappa, q)$. Particularly, we say that AHF is (poly, n)-admissible if AHF is (q,n)-admissible for any polynomial $q = q(\kappa)$ and constant $n > 0$. To show the existence of (q,n)-AHF for $n \geq 1$, we present the following theorem.

Theorem 4.2. *Let $q = q(\kappa)$ be a polynomial, n be a constant, and AHF (with $\mathsf{AdmSample}$) be a $(q,1)$-AHF from X into Y^ℓ. Then the hash function $\widehat{\mathsf{AHF}}$ with*

- $\widehat{\mathsf{AHF}}(x) = \underbrace{\mathsf{AHF}(x) \times \cdots \times \mathsf{AHF}(x)}_{n}$;
- $\hat{P}_{\hat{u}} : X \to \{0,1\}$ *for any* $\hat{u} \in ((Y \cup E)^n)^{\ell^n}$ *is defined as* $\hat{P}_{\hat{u}}(x) = 0 \iff \forall i : \widehat{\mathsf{AHF}}(x)_i \neq \hat{u}_i$, *where* $1 \leq i \leq \ell^n$.
- $\widehat{\mathsf{AdmSample}}(\kappa, q)$: *run* $\mathsf{AdmSample}(\kappa, q)$ *independently* n *times to obtain* $u_1, \dots, u_n \in (Y \cup E)^\ell$, *output* $\hat{u} = \underbrace{u_1 \times \cdots \times u_n}_{n}$.

is a (q,n)-AHF from X into $(Y^n)^{\ell^n}$. Here $\times$ denotes the operation of Cartesian product defined in Section 2.1. $\widehat{\mathsf{AHF}}(x)$ can be viewed as a ℓ^n-dimension vector over $\overline{Y} = Y^n$, $\hat{u}$ can be viewed as a ℓ^n-dimension vector over $(Y \cup E)^n$, and the associated $\overline{E} = (Y \cup E)^n \backslash \overline{Y}$.

Proof. We first note that the definition of $\hat{P}_{\hat{u}}$ for $\widehat{\mathsf{AHF}}$ is compatible with that of P_u for AHF. According the construction of $\widehat{\mathsf{AHF}}$ and $\widehat{\mathsf{AdmSample}}(\kappa, q)$, it is easy to verify that $\hat{P}_{\hat{u}}(x) = 1$ is equivalent to $P_{u_i}(x) = 1$ for all $i \in [n]$, and $\hat{P}_{\hat{u}}(x) = 0$ is equivalent to $P_{u_i}(x) = 0$ for at least one $i \in [n]$. Therefore, we have $\hat{P}_{\hat{u}}(x) = P_{u_1}(x) \wedge \cdots \wedge P_{u_n}(x)$. Now fix $q + n$ distinct elements $x_1, \dots, x_q, z_1, \dots, z_n \in X$. For each $i \in [n]$, define event A_i as: $P_{u_i}(x_j) = 1$ for all $1 \leq j \leq q$ and $P_{u_i}(z_i) = 0$ (the predicate values on the rest $n - 1$ elements could be either 0 or 1). Define event A as: $\hat{P}_{\hat{u}}(x_j) = 1$ for all $1 \leq j \leq q$ and $\hat{P}_{\hat{u}}(z_i) = 0$ for all $1 \leq i \leq n$. According to the definition of $\hat{P}_{\hat{u}}$, we have: $A \supseteq A_1 \wedge \cdots \wedge A_n$. Since AHF is a $(q,1)$-AHF, thus each event A_i happens independently with probability at least $1/\theta(\kappa)$ (over the choice of $u_i \leftarrow \mathsf{AdmSample}(\kappa, q)$). Therefore, we have: $\Pr[A] \geq \prod_{i=1}^{n} \Pr[A_i] \geq 1/(\theta(\kappa))^n$, which indicates $\widehat{\mathsf{AHF}}$ is a (q, n)-AHF. This proves the theorem. □

Remark 4.1. Our generalization of AHF comes from three aspects: 1) the parameter choice $(\mathsf{poly}, 1)$ extends to (poly, n) for any constant $n \geq 1$; 2) the basic field of the range extends from $\{0, 1\}$ to an abstracted field Y; 3) E could be any set that is disjoint with Y instead of a singleton $\{\perp\}$. We also define the density ρ of AHF as $1/|Y|$. One should have the highest density is $1/2$ in mind here. Intuitively, AHF with high density admits less parameter-intensive applications.

According to the construction shown in [FHPS13, Theorem 2], one can build a $(\mathsf{poly}, 1)$-AHF from $\{0, 1\}^l$ to $\{0, 1\}^\ell$ from a family of codes $\mathcal{C}_l : \{0, 1\}^l \to \{0, 1\}^\ell$ with minimum distance at least $c \cdot \ell$ for a fixed constant $c > 0$. Using the above construction shown in Theorem 4.2, we can further build a $(\mathsf{poly}, 2)$-AHF from $\{0, 1\}^l$ to $(\{0, 1\}^2)^{\ell^2}$, and use it as a main ingredient to construct the adaptively secure IB-NIKE as below:

Adaptively Secure Construction from $i\mathcal{O}$

- $\mathsf{Setup}(\kappa)$: run $\mathsf{BLGroupGen}(\kappa)$ to generate $(p, \mathbb{G}, \mathbb{G}_T, e)$, pick $x \xleftarrow{\text{R}} \mathbb{Z}_p$ and $g \xleftarrow{\text{R}} \mathbb{G}^*$; pick a secret key k for puncturable PRF $\mathsf{F} : I \to \mathbb{Z}_p$[6]; set $t = \ell^2$, for $i \in [t]$ and $\alpha_i \in Y = \{(0, 0), (0, 1), (1, 0), (1, 1)\}$ pick c_{i,α_i} uniformly at random in $\mathbb{Z}_p$; then create an obfuscation of the program H shown in Figure 5, where the size of the program is padded to be the maximum of itself and the program of H^* shown in Figure 6. The *msk* is x, whereas *mpk* is the hash function $\mathsf{H}(\cdot)$.
- Algorithm $\mathsf{Extract}$ and Share are identical to that in the SOK IB-NIKE scheme.

```
Adaptive Hash H

Constants: $g \in \mathbb{G}^*$, exponents $c_{i,\alpha_i} \in \mathbb{Z}_p$ for $i \in [t]$ and $\alpha_i \in Y$.
Input: Identity $id$.
  1. Compute $w \leftarrow \mathsf{AHF}(id)$.
  2. Output $g^{\prod_{i=1}^{t} c_{i,w_i}}$.
```

Figure 5: Adaptive Hash H

Theorem 4.3. *The above IB-NIKE scheme is adaptively secure if the obfuscation scheme is indistinguishable secure and the t-DBDHI assumption holds in bilinear group system.*

Proof. We proceed via a sequence of hybrid games, where the first game corresponds to the standard adaptive security game. We first prove that any two successive games are computationally indistinguishable. We then show that any PPT adversary in the final game that succeeds with non-negligible probability can be used to break the n-DBDHI assumption.

Game 0: This game is identical to standard adaptive security game played between adversary $\mathcal{A}$ and challenger $\mathcal{CH}$:

[6] Without loss of generality, we assume the identity space I is $\{0, 1\}^l$. I can be extended to $\{0, 1\}^*$ by using a collision resistant hash function $\mathsf{CRF} : \{0, 1\}^* \to \{0, 1\}^t$ prior to secret key extraction and key sharing.

Adaptive Hash H^*

Constants: $g \in \mathbb{G}^*$, $g^x, \ldots, g^{x^t} \in \mathbb{G}$ for some $x \in \mathbb{Z}_p$, exponents $y_{i,\alpha_i} \in \mathbb{Z}_p$ for $i \in [t]$ and $\alpha_i \in Y$, $u \in \{Y, (\perp, 0), (\perp, 1), (0, \perp), (1, \perp), (\perp, \perp)\}^t$.

Input: Identity id.

1. Compute $w \leftarrow \mathsf{AHF}(id)$.
2. Compute the set size $|\mu(w)|$, where $\mu(w)$ is the set of i such that $w_i \neq u_i$.
3. Output $(g^{x^{|\mu(w)|}})^{\prod_{i=1}^{t} y_{i,w_i}}$.

Figure 6: Adaptive Hash H^*

- Setup: $\mathcal{CH}$ runs $\mathsf{BLGroupGen}(\kappa)$ to generate $(p, \mathbb{G}, \mathbb{G}_T, e)$, picks $x \xleftarrow{R} \mathbb{Z}_p$ and $g \xleftarrow{R} \mathbb{G}^*$. $\mathcal{CH}$ then chooses exponents c_{i,α_i} uniformly at random in $\mathbb{Z}_p$ for $i \in [t]$ and $\alpha_i \in Y$, creates the hash function $\mathsf{H}(\cdot)$ as an obfuscation of the program of H shown in Figure 5, and pads its size to be the maximum of itself and the program of H^* shown in Figure 6. $\mathcal{CH}$ sets $msk = x$ and $mpk = \mathsf{H}$.
- Phase 1: $\mathcal{A}$ can issue the following two types of queries:
  - extraction query $\langle id \rangle$: $\mathcal{CH}$ responds with $sk_{id} = \mathsf{H}(id)^x$.
  - reveal query $\langle id_a, id_b \rangle$: $\mathcal{CH}$ first extracts secret key sk_{id_a} for id_a, then responds with $shk \leftarrow \mathsf{Share}(sk_{id_a}, id_b)$.
- Challenge: $\mathcal{A}$ submits id_a^* and id_b^* as the target identities with the restriction that either id_a^* or id_b^* has not been queried for secret key. $\mathcal{CH}$ picks $shk_0^* \xleftarrow{R} SHK$ and computes $shk_1^* \leftarrow \mathsf{Share}(sk_{id_a^*}, id_b^*)$, then picks $\beta \xleftarrow{R} \{0, 1\}$ and sends shk_β^* to $\mathcal{A}$ as the challenge.
- Phase 2: $\mathcal{A}$ can continue to issue the extraction queries and the reveal queries, $\mathcal{CH}$ proceeds the same way as in Phase 1 except that the extraction queries to id_a^* or id_b^* and reveal query for (id_a^*, id_b^*) are not allowed.
- Guess: $\mathcal{A}$ outputs its guess β' and wins if $\beta = \beta'$.

Game 1: same as Game 0 except that $\mathcal{CH}$ generates the exponents $c_{i,\alpha}$ as follows: first samples u via $\mathsf{AdmSample}(\kappa, q)$, where q is the sum of q_e (the maximum number of extraction queries) and q_r (the maximum number of reveal queries), then for $i \in [t]$ and $\alpha_i \in Y$ chooses y_{i,α_i} each randomly from $\mathbb{Z}_p$ and sets:

$$c_{i,\alpha_i} = \begin{cases} y_{i,\alpha_i} & \text{if } \alpha_i = u_i \\ x \cdot y_{i,\alpha_i} & \text{if } \alpha_i \neq u_i \end{cases}$$

Game 2: same as Game 1 except that $\mathcal{CH}$ creates the hash function $\mathsf{H}(\cdot)$ as an obfuscation of program H^* shown in Figure 6.

Lemma 4.4. *Game 0 and Game 1 are statistically indistinguishable.*

Proof. This lemma immediately follows from the facts: (1) in Game 1 the sampling of u only determines the generation of c_{i,α_i} and it is independent of the rest game; (2) the value of c_{i,α_i} distributes uniformly at random in $\mathbb{Z}_p$ in both Game 0 and Game 1. □

Lemma 4.5. *Game 1 and Game 2 are computationally indistinguishable if the underlying obfuscation scheme is indistinguishability secure.*

Proof. We prove this lemma by giving a reduction to the indistinguishability security of the obfuscator. More precisely, suppose there is a PPT adversary $\mathcal{A}$ can distinguish Game 1 and Game 2, then we can build algorithms $(\mathcal{S}, \mathcal{D})$ against the indistinguishability of the obfuscator by interacting with $\mathcal{A}$ as follows.

Sample: $\mathcal{S}$ runs $\mathsf{BLGroupGen}(\kappa)$ to generate $(p, \mathbb{G}, \mathbb{G}_T, e)$, picks $x \xleftarrow{\text{R}} \mathbb{Z}_p$ and $g \xleftarrow{\text{R}} \mathbb{G}$, prepares g^{x^i} for $i \in [t]$, runs $\mathsf{AdmSample}(\kappa, q)$ to sample a string u, and for $i \in [t]$ and $\alpha_i \in Y$ chooses y_{i,α_i} each randomly from $\mathbb{Z}_p$, then sets:

$$c_{i,\alpha_i} = \begin{cases} y_{i,\alpha_i} & \text{if } \alpha_i = u_i \\ x \cdot y_{i,\alpha_i} & \text{if } \alpha_i \neq u_i \end{cases}$$

It sets $\tau = (c_{i,\alpha_i}, y_{i,\alpha_i}, u)$ and builds C_0 as the program of H, and C_1 as the program of H^*. Before describing $\mathcal{D}$, we observe that by construction, the circuits C_0 and C_1 behave identically on every input. To show program equivalence, note that for all $w \in \{0,1\}^n$, we have that:

$$g^{\prod_i^t c_{i,\alpha_i}} = g^{x^{|\mu(w)|} \cdot \prod_i^t y_{i,w_i}} = (g^{x^{|\mu(w)|}})^{\prod_i^t y_{i,w_i}}$$

With suitable padding, both C_0 and C_1 have the same size. Thus, $\mathcal{S}$ satisfies the conditions needed for invoking the indistinguishability property of the obfuscator. Now, we can describe the algorithm $\mathcal{D}$, which takes as input τ as given above, and the obfuscation of either C_0 or C_1.
Distinguish: $\mathcal{D}$ sets $msk = x$ and builds mpk from C_β, then invokes $\mathcal{A}$ in the adaptive security game for IB-NIKE. When $\mathcal{A}$ issues extraction queries and reveal queries, $\mathcal{D}$ responds with msk. If $\mathcal{A}$ wins, $\mathcal{D}$ outputs 1.

By construction, if $\mathcal{D}$ receives an obfuscation of C_0, then the probability that $\mathcal{D}$ outputs 1 is exactly the probability that $\mathcal{A}$ wins in Game 1. On the other hand, if $\mathcal{D}$ receives an obfuscation of C_1, then the probability that $\mathcal{D}$ outputs 1 is the probability that $\mathcal{A}$ wins in Game 2. The indistinguishability of the obfuscator implies Game 1 and Game 2 are computationally indistinguishable. The lemma immediately follows. □

Lemma 4.6. *$\mathcal{A}$'s advantage in Game 2 is negligible in κ.*

Proof. We prove this lemma by showing that any adversary $\mathcal{A}$ with non-negligible advantage in Game 2 implies an algorithm $\mathcal{B}$ with non-negligible advantage against the n-DBDHI problem. Given a n-DBDHI instance $(g, g^x, \ldots, g^{x^t}, T_\beta)$, $\mathcal{B}$ interacts with $\mathcal{A}$ as follows:

- Setup: $\mathcal{B}$ first runs $\mathsf{AdmSample}(\kappa, q)$ to sample u, where q is the sum of q_e (the maximum number of extraction queries) and q_r (the maximum number of reveal queries). For $i \in [t]$ and $\alpha_i \in Y$, $\mathcal{B}$ chooses $y_{i,\alpha_i} \xleftarrow{\text{R}} \mathbb{Z}_p$, then creates the hash function $\mathsf{H}(\cdot)$ as an obfuscation of the program H^* using the input DBDHI instance as well as y_{i,α_i} and u.
- Phase 1: $\mathcal{A}$ can issue the following two types of queries:
  - extraction queries $\langle id \rangle$: If $P_u(id) = 0$, then $\mathcal{B}$ aborts and outputs a random guess for β. Else, $\mathcal{B}$ extracts the secret key from the input n-DBDHI instance and the $y_{i,\alpha}$ values. $\mathcal{B}$ could do so since $P_u(id) = 1$ implies there exists at least one i such that $w_i = u_i$. In this case $\mathsf{H}(id)$ will contain a power of x that is strictly less than n.
  - reveal queries $\langle id_a, id_b \rangle$: If $P_u(id_a) = 0 \wedge P_u(id_b) = 0$, then $\mathcal{B}$ aborts and outputs a random guess for β. Otherwise, either $P_u(id_a) = 1$ or $P_u(id_b) = 1$. Therefore, $\mathcal{B}$ can at least extract a secret key for one identity and then computes the shared key.

- Challenge: $\mathcal{A}$ outputs the target identities (id_a^*, id_b^*). If $P_u(id_a^*) = 1 \vee P_u(id_b^*) = 1$, then $\mathcal{B}$ aborts and outputs a random guess for β. Else, we have $P_u(id_a^*) = 0 \wedge P_u(id_b^*) = 0$, which means $\mathsf{AHF}(id_a^*)_i \neq u_i$ and $\mathsf{AHF}(id_b^*)_i \neq u_i$ for all $i \in [t]$. In this situation, both the hash values of id_a^* and id_b^* will be g^{a^t} raised to some known product of some $y_{i,\alpha}$ values. Denote the products by y_a^* and y_b^*, respectively. $\mathcal{B}$ thus sends $shk_\beta^* = (T_\beta)^{y_a^* y_b^*}$ to $\mathcal{A}$ as the challenge. It is easy to verify that if $T_\beta \xleftarrow{\text{R}} \mathbb{G}_T$ then shk_β^* also distributes uniformly over $\mathbb{G}_T$, else if $T_\beta = e(g,g)^{x^{2t+1}}$ then $shk_\beta^* = e(\mathsf{H}(id_a^*), \mathsf{H}(id_b^*))^a$.
- Phase 2: same as in Phase 1 except that the extraction queries $\langle id_a^* \rangle$, $\langle id_b^* \rangle$ and the reveal query $\langle id_a^*, id_b^* \rangle$ are not allowed.
- Guess: When $\mathcal{A}$ outputs its guess β', $\mathcal{B}$ forwards β' to its own challenger.

Since the choice of $u \leftarrow \mathsf{AdmSample}(\kappa, q)$ determines whether or not $\mathcal{B}$ aborts and it is independent of the rest of the interaction. We conclude that conditioned on $\mathcal{B}$ does not abort, $\mathcal{A}$'s view in the above game is identical to that in Game 2. Let F be the event that $\mathcal{B}$ does not abort, we have $\mathsf{Adv}_{\mathcal{B}}(\kappa) = \Pr[F] \cdot \mathsf{Adv}_{\mathcal{A}}(\kappa)$. In what follows, we estimate the low bound of $\Pr[F]$. Let $\{id_i\}_{1 \leq i \leq q_e}$ be q_e distinct extraction queries, $\{(id_{j,1}, id_{j,2})\}_{1 \leq j \leq q_r}$ be q_r distinct reveal queries. During the game, $\mathcal{B}$ will abort if one of the following events does not happen.

$$\begin{aligned} F_1 : &\quad \textstyle\bigwedge_{i=1}^{q_e} (P(id_i) = 1) \\ F_2 : &\quad \textstyle\bigwedge_{j=1}^{q_r} (P(id_{j,1}) = 1 \vee P(id_{j,2}) = 1) \\ F_3 : &\quad P_u(id_1^*) = 0 \wedge P_u(id_2^*) = 0 \end{aligned}$$

We have $F = F_1 \wedge F_2 \wedge F_3$. Note that in each extraction query, there exists at least one identity different from both id_1^* and id_2^*. Suppose $q_e + q_r \leq q$, then according to the fact that AHF is $(q, 2)$-admissible, we have $\Pr[F] \geq \theta(\kappa)$. The lemma immediately follows. □

Combining the above three lemmas, our main theorem immediately follows. □

5 IB-NIKE from Extractable Witness PRFs

As previously mentioned, program obfuscation has proven to be an extremely powerful tool and has been used to construct a variety of cryptographic primitives with amazing properties. In particular, Boneh and Zhandry [BZ14] showed how to use $i\mathcal{O}$ to construct both multiparty NIKE and IB-NIKE.

Very recently, Zhandry [Zha14] showed that for several applications of obfuscation, a weak primitive named witness PRFs (WPRFs) actually suffices. In particular, Zhandry use witness PRFs to replace obfuscator in the $i\mathcal{O}$-based NIKE construction [BZ14], and prove that the same security still holds. However, we note that analogous replacement does not directly works for the $i\mathcal{O}$-based IB-NIKE construction [BZ14] mainly of the following reasons: 1) in the IB-NIKE setting, the target identities are chosen by the adversary, which might be elements in L. Nevertheless, the standard WPRFs only guarantee pseudorandomness on elements outside L. Therefore, we do not know how to reduce the security of IB-NIKE to the pseudorandomness of WPRFs. 2) in the IB-NIKE security experiment, the adversary is allowed to obtain secret keys for any identities other than the target identities. However, standard WPRFs do not allowed key delegation.

We overcome the above hurdles by employing WPRFs with strongly extractable property together with a signature scheme in charge of secret key generation.

5.1 Witness Pseudorandom Functions

Roughly speaking, witness PRFs are defined with respect to an NP language L. One can evaluate the PRF value at an instance $x \in L$ in two ways, either via the secret key or via the public evaluation key and a witness $w \in W$ for $x \in L$. More formally, WPRFs are given by the following algorithms:

- $\mathsf{KeyGen}(\kappa, \mathsf{R})$: on input a security parameter κ and a binary relation $\mathsf{R} : X \times W \to \{0, 1\}$, output a public evaluation key ek and a secret key k.
- $\mathsf{Eval}(ek, x, w)$: on input ek, an element $x \in X$, and a witness $w \in W$, output $y \in Y \cup \perp$.

Correctness: For any $(k, ek) \leftarrow \mathsf{KeyGen}(\kappa, \mathsf{R})$, and any $x \in X$ and $w \in W$, we have:

$$\mathsf{Eval}(ek, x, w) = \begin{cases} \mathsf{F}(k, x) & \text{if } \mathsf{R}(x, w) = 1 \\ \perp & \text{if } \mathsf{R}(x, w) = 0 \end{cases}$$

Pseudorandomness: Here we define the pseudorandomness of WPRFs for inputs in the language L.[7] Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary against WPRFs and define its advantage as:

$$\mathsf{Adv}_{\mathcal{A}}(\kappa) = \Pr\left[b = b' : \begin{array}{l} (ek, k) \leftarrow \mathsf{KeyGen}(\kappa, \mathsf{R}); \\ (x^*, state) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{eval}}(\cdot)}(ek); \\ y_0^* \xleftarrow{\text{R}} Y, y_1^* \leftarrow \mathsf{F}(k, x^*); \\ b \leftarrow \{0, 1\}; \\ b' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{eval}}(\cdot)}(state, y_b^*); \end{array} \right] - \frac{1}{2},$$

where $\mathcal{O}_{\mathsf{eval}}(x) = \mathsf{F}(k, x)$, and x^* is required to be an element in L. Both $\mathcal{A}_1$ and $\mathcal{A}_2$ are not allowed to query $\mathcal{O}_{\mathsf{eval}}(\cdot)$ for x^*. We say that WPRFs are adaptively pseudorandom if for any PPT adversary its advantage function $\mathsf{Adv}_{\mathcal{A}}(\kappa)$ is negligible in κ.

Extractability: For some applications, we will need an extractable property of WPRFs, which roughly states that for $x^* \in L$ the value $\mathsf{F}(x^*)$ is pseudorandom unless the adversary knows a witness w for x^*. More formally, we say WPRFs are extractable if for any PPT adversary $\mathcal{A}$ with non-negligible advantage $\mathrm{Adv}_{\mathcal{A}}(\kappa)$ in distinguishing $\mathsf{F}(x^*)$ from random, there exists an efficient extractor $\mathcal{E}^{\mathcal{A}}$ which outputs w such that $(x^*, w) \in \mathsf{R}$ with non-negligible advantage.

Enhancement: Observe that every NP language L with relation $\mathsf{R} : X \times W$ naturally induces an extended NP language $\overline{L}$ with relation $\overline{\mathsf{R}} : \overline{X} \times \overline{W}$, where $\overline{X} = X^n$ for some integer n and $\overline{W} = W \times [n]$. For $\overline{x} = (x_1, \dots, x_n)$ and $\overline{w} = (w, i)$ we define $(\overline{x}, \overline{w}) \in \overline{\mathsf{R}}$ iff $(x_i, w) \in \mathsf{R}$. Obviously, a witness w for $x_i^* \in L$ corresponds to a witness $\overline{w} = (w, i)$ for instances of the form $\overline{x} = (x_1, \dots, x_{i-1}, x_i^*, x_{i+1}, \dots, x_n) \in \overline{L}$. For WPRFs defined with respect to extended NP language $\overline{L}$ induced by L, we can consider an enhanced notion of adaptive pseudorandomness. More precisely, we given adversary access to an additional oracle $\mathcal{O}_{\mathsf{wit}}(\cdot)$, which on input $x \in L$ outputs a witness w for $x \in L$. Let $\overline{x}^* = (x_1^*, \dots, x_n^*)$ be the target element. To prevent trivial attack, the adversary is not allowed to query $\mathcal{O}_{\mathsf{wit}}(\cdot)$ with x_i^* for any $i \in [n]$. The extractable property for adaptively pseudorandom WPRFs in the enhanced sense can be defined naturally. We say such WPRFs are strongly extractable.

[7] We note that in [Zha14] the pseudorandomness of WPRFs is defined with respect to inputs outside the language L. Since we focus on extractable WPRFs in this work, it is sufficient to only consider pseudorandomness of WPRFs for inputs in L.

5.2 Multiparty IB-NIKE from Extractable Witness PRFs

In this section, we show how to construct multiparty IB-NIKE from extractable WPRF and an EUF-CMA secure signature (c.f. definition in Section A.3) with message space I and signature space Σ.

- $\mathsf{KeyGen}(\kappa, n)$: on input κ and n, run Sig.$\mathsf{KeyGen}(\kappa)$ to generate a verification/signing key pair (vk, sk), run WPRF.$\mathsf{KeyGen}(\kappa, \overline{\mathsf{R}})$ to generate (ek, k), where $\overline{\mathsf{R}} : I^n \times (\Sigma, [n]) \rightarrow \{0,1\}$ is a binary relation induced by $\mathsf{R} : I \times \Sigma \rightarrow \{0,1\}$ that outputs Sig.$\mathsf{Verify}(vk, id, \sigma)$ on input tuple (id, σ); output $mpk = ek$ and $msk = (k, sk)$.
- $\mathsf{Extract}(msk, id)$: on input msk and id, output $sk_{id} \leftarrow$ Sig.$\mathsf{Sign}(sk, id)$. Note that a secret key sk_{id} actually is a signature of id, which serves as a witness for $id \in I$.
- $\mathsf{Share}(sk_{id}, \mathcal{I})$: on input sk_{id} and $\mathcal{I} = (id_1, \ldots, id_n)$, find i such that $id = id_i$, output WPRF.$\mathsf{Eval}(ek, (id_1, \ldots, id_n), (sk_{id}, i))$.

Theorem 5.1. *The above IB-NIKE construction is adaptively secure if the underlying WPRFs are strongly extractable and the signature scheme is EUF-CMA.*

Proof. We prove this theorem via the following two lemmas.

Lemma 5.1. *The WPRFs with respect to $\overline{\mathsf{L}}$ are adaptively pseudorandom in the enhanced sense if the WPRFs are extractable and the signature scheme is EUF-CMA.*

Proof. Suppose there is an adversary $\mathcal{A}$ which has non-negligible advantage against the adaptive pseudorandomness in the enhanced sense of the WPRFs, then according to assumed extractability there exists a PPT algorithm which can extract a witness (w^*, i) for $\overline{x}^*$, which in turn violates the assumed EUF-CMA of the signature scheme. More precisely, let $\mathcal{F}$ be an adversary against the underlying signature scheme. Given vk, $\mathcal{F}$ interacts with $\mathcal{A}$ as follows:

Setup: $\mathcal{F}$ runs WPRF.$\mathsf{KeyGen}(\kappa, \overline{\mathsf{R}})$ to generate (ek, k), and sends ek to $\mathcal{A}$ as mpk.
Phase 1: $\mathcal{A}$ can adaptively make two types of queries:

- Witness query $\langle x \rangle$: $\mathcal{F}$ makes signing query $\langle x \rangle$ to its own challenger and forwards the reply to $\mathcal{A}$.
- Evaluation query $\langle \overline{x} \rangle$: $\mathcal{F}$ replies $\mathcal{A}$ with $\mathsf{F}(k, \overline{x})$. Note that $\mathcal{F}$ can answer any evaluation queries with k.

Challenge: $\mathcal{A}$ chooses $\overline{x}^* = (x_1^*, \ldots, x_n^*)$ as the target point. $\mathcal{F}$ picks $y_0^* \xleftarrow{\text{R}} Y$, computes $y_1^* \leftarrow \mathsf{F}(k, \overline{x}^*)$, picks a random bit b, and sends y_b^* to $\mathcal{A}$.
Phase 2: $\mathcal{F}$ proceeds the same way as in Phase 1.
Guess: $\mathcal{A}$ outputs its guess b' for b.

It is easy to verify that $\mathcal{F}$ provides a perfect simulation for $\mathcal{A}$. $\mathcal{F}$ then runs extractor $\mathcal{E}$. As soon as $\mathcal{E}$ outputs a witness (w^*, i) for $\overline{x}^*$, $\mathcal{F}$ outputs (x_i^*, w^*) to its own challenger as the forgery. The forgery is valid since x_i^* has never been queried for signature. Suppose $\mathcal{A}$ has non-negligible advantage against the WPRFs, then according to the extractability $\mathcal{F}$ also has non-negligible advantage against the signature scheme, which contradicts to the assumed unforgeability. Thereby, the WPRFs are adaptively pseudorandom in the enhanced sense. □

Lemma 5.2. *The IB-NIKE construction is adaptively secure if the WPRFs with respect to $\overline{\mathsf{L}}$ are adaptively pseudorandom in the enhanced sense.*

Proof. Suppose there is an adversary $\mathcal{A}$ has non-negligible advantage against the IB-NIKE construction, then we can build an algorithm $\mathcal{D}$ breaks the enhanced adaptive pseudorandomness of the WPRFs. Given ek where $(ek, k) \leftarrow \text{WPRF}.\mathsf{KeyGen}(\kappa, \overline{\mathsf{R}})$, $\mathcal{D}$ simulates $\mathcal{A}$'s challenger as below:

Setup: $\mathcal{D}$ forwards ek to $\mathcal{A}$ as mpk.
Phase 1: $\mathcal{A}$ can issue the following two types of queries:

- Extract queries $\langle id \rangle$: $\mathcal{D}$ makes witness query $\langle id \rangle$ to its own challenger, and then forwards the reply to $\mathcal{A}$.
- Reveal queries $\langle \mathcal{I} \rangle$: $\mathcal{D}$ issues evaluation query $\langle \mathcal{I} \rangle$ to its own challenger and forwards the reply to $\mathcal{A}$.

Challenge: $\mathcal{A}$ outputs $\mathcal{I}^*$ to $\mathcal{D}$, $\mathcal{D}$ submits $\mathcal{I}^*$ to its own challenger and forwards the reply y_b^* to $\mathcal{A}$.
Phase 2: The same as in Phase 1.
Guess: $\mathcal{A}$ outputs its guess b' of b. $\mathcal{D}$ sends b' to its own challenger.

It is easy to verify that $\mathcal{D}$'s simulation is perfect and thus $\mathcal{D}$ has the same advantage as $\mathcal{A}$. □

Theorem 5.1 immediately follows from Lemma 5.1 and Lemma 5.2. □

6 Relation Between Semi-Adaptive Security and Adaptive Security

Semi-adaptive security model gives the adversary access to an $\mathcal{O}_{\mathsf{extract}}$ oracle, which captures collusion attacks but does not give the adversary any direct oracle access to shared keys. Adaptive security model further gives the adversary access to a $\mathcal{O}_{\mathsf{reveal}}$ oracle. It seems that semi-adaptive security model is weaker than adaptive security model, since the adversary in the former model is usually considered to be more restricted. Actually, as noted in [PS09], a semi-adaptive adversary can always compute any shared key after extracting the secret key of one relevant identity other than the target identities, which instructively indicates that shared key reveal oracle may be simulated by secret key extraction oracle and thus semi-adaptive security is equivalent to adaptive security. Looking ahead, when reducing adaptive security to semi-adaptive security, for each reveal query the reduction first guesses which identity will not be chosen as the target identity and then issues extraction query for this identity. We remark that trivial guessing trick will lead to exponential security loss. To avoid this issue, our reduction employs the flipping coin technique again to guess smartly. In what follows, we formally prove the equivalence.

Theorem 6.1. *For IB-NIKE, semi-adaptive security is polynomially equivalent to adaptive security.*

Proof. We prove this theorem via the following two lemmas.

Lemma 6.1. *For IB-NIKE, adaptive security implies semi-adaptive security.*

Proof. This direction is trivial. □

Lemma 6.2. *For IB-NIKE, semi-adaptive security implies adaptive security. In other words, if an IB-NIKE scheme* Π *is semi-adaptive secure, then it is also adaptive secure.*

Proof. Let $\mathcal{A}$ be an adversary against adaptive security of Π, we construct an adversary $\mathcal{B}$ against the assumed semi-adaptive security of Π. Given mpk where $(mpk, msk) \leftarrow \Pi.\mathsf{Setup}(n, \kappa)$, $\mathcal{B}$ simulates $\mathcal{A}$'s challenger as follows:

- **Setup:** $\mathcal{B}$ sends mpk to $\mathcal{A}$. To prepare the interaction, $\mathcal{B}$ maintains a list H, which is initially empty. Each entry in H is of the form $(id, mark)$, where $id \in I$, $mark \in \{0, 1\}$. Intuitively, id with $mark = 0$ will correspond to identities that $\mathcal{B}$ can query for secret keys, while id with $mark = 1$ will correspond to identities that $\mathcal{B}$ can choose as target identities. For each fresh id appears in extraction queries, reveal queries, and the challenge queries, $\mathcal{B}$ will mark it with "0" with probability δ (which will be determined later) and mark it with "1" with probability $1 - \delta$. During the simulation, $\mathcal{B}$ may abort its interaction with $\mathcal{A}$. In such cases, $\mathcal{B}$ continues its interaction with its own challenger, and outputs a random guess at the end.
- **Phase 1:** $\mathcal{A}$ can issue the following two types of queries:
  - extraction queries $\langle id \rangle$: If id is marked with "1", $\mathcal{B}$ aborts its interaction with $\mathcal{A}$. Else, $\mathcal{B}$ issues extraction query $\langle id \rangle$ to its own challenger and forwards the reply to $\mathcal{A}$.
  - reveal queries $\langle \mathcal{I} \rangle$: Suppose $\mathcal{I} = (id_1, \ldots, id_n)$. If all id_i are marked with "1", $\mathcal{B}$ aborts its interaction with $\mathcal{A}$. Else, $\mathcal{B}$ randomly chooses an id_i marked with "0", issues secret key query $\langle id_i \rangle$ to its own challenger, and uses the received secret key to answer the reveal query.
- **Challenge:** $\mathcal{A}$ submits $\mathcal{I}^* = (id_1^*, \ldots, id_n^*)$ as the target identities with the restriction that id_i^* has not been queried for secret key and $\mathcal{I}^*$ has not been queried for shared key. If there exists an index $i \in [n]$ such that id_i^* is marked with "0", $\mathcal{B}$ aborts. Else, $\mathcal{B}$ submits $\mathcal{I}^*$ to its own challenger and forwards the challenge shk_β^* to $\mathcal{A}^*$.
- **Phase 2:** $\mathcal{A}$ can continue to issue extraction queries and reveal queries as in Phase 1. $\mathcal{B}$ proceeds the same way as in Phase 1.
- **Guess:** $\mathcal{A}$ outputs its guess β for β'. $\mathcal{B}$ forwards β' to its own challenger.

It is easy to see that conditioned on $\mathcal{B}$ does not abort, $\mathcal{A}$'s view in the above game is identical to the real IB-NIKE adaptive security game. Let F be the event that $\mathcal{B}$ does not abort, we have $\mathsf{Adv}_{\mathcal{B}}(\kappa) = \Pr[F] \cdot \mathsf{Adv}_{\mathcal{A}}(\kappa)$. In what follows, we compute the low bound of $\Pr[F]$. Let $\{id_i\}_{1 \le i \le q_e}$ be q_e distinct extraction queries, $\{(id_{j,1}, \ldots, id_{j,n})\}_{1 \le j \le q_r}$ be q_r distinct reveal queries. To ease the analysis, we further define the following events:

$$\begin{aligned} F_1 : &\quad \textstyle\bigwedge_{i=1}^{q_e} (P(id_i) = 0) \\ F_2 : &\quad \textstyle\bigwedge_{j=1}^{q_r} (P(id_{j,1}) = 0 \vee \cdots \vee \cdots \vee P(id_{j,n}) = 0) \\ F_3 : &\quad P(id_1^*) = 1 \wedge \cdots \wedge \cdots \wedge P(id_n^*) = 1 \end{aligned}$$

Obviously, we have $F = F_1 \wedge F_2 \wedge F_3$. Therefore, we have:

$$\Pr[F] = \Pr[F_1] \cdot \Pr[F_2 \wedge F_3 \mid F_1]$$

Since each coin toss for $mark$ is independent, we have $\Pr[F_1] = \delta^{q_e}$. Note that in each reveal query there exists at least one identity different from the identities in $\mathcal{I}^*$, then we have $\Pr[F_2 \wedge F_3 \mid F_1] \ge \delta^{q_r}(1 - \delta)^n$. Without lose of generality, we assume the sets $\bigcup_i^{q_e} id_i$ and $\bigcup_j^{q_r} (id_{j,1}, \ldots, id_{j,n})$ are disjointed. Note that if these two sets have intersection, $\Pr[F]$ could only be larger. Finally, we arrive at $\Pr[F] \ge \delta^{q_e + q_r}(1 - \delta)^n$. Let $f(\delta) = \delta^{q_e + q_r}(1 - \delta)^n$. This function achieves the maximum value at the zero point $1 - n/(q_e + q_r + n)$ of $f'(\delta)$, therefore we have:

$$\Pr[F] \ge \frac{(q_e + q_r)^{q_e + q_r} n^n}{(q_e + q_r + n)^{q_e + q_r + n}}.$$

This proves Lemma 6.2. □

Theorem 6.1 immediately follows from Lemma 6.1 and Lemma 6.2. □

While the adaptive security is our preferred security notion, the semi-adaptive security is more simple and easy to use. Since we have shown these two security notions are polynomially equivalent, it suffices to analyze schemes under the semi-adaptive security notion if the concrete security is not overly concerned.

7 Open Problems

In this paper, we throughly revisited the SOK IB-NIKE scheme, as well as presented some new results about general IB-NIKE. Here we point out a few possible directions for future research.

Acknowledgment

We greatly thank Dennis Hofheinz for insightful discussions and helpful advice. In particular, we are grateful to Dennis for suggesting the construction of (poly, n)-AHFs in Section 4.2.

References

[BB04a] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, 2004.

[BB04b] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *LNCS*, pages 443–459. Springer, 2004.

[BF01] Dan Boneh and Matthew Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology - CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, 2001.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In *17th International Conference on Practice and Theory in Public-Key Cryptography, PKC 2014*, volume 8383 of *LNCS*, pages 501–519. Springer, 2014.

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532, 2001.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *1st ACM Conference on Computer and Communications Security*, pages 62–73. ACM, 1993.

[BR96] Mihir Bellare and Phillip Rogaway. The exact security of digital signatures - how to sign with rsa and rabin. In *Advances in Cryptology - EUROCRYPT 1996*, volume 1070 of *LNCS*, pages 399–416, 1996.

[BR09] Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for waters' ibe scheme. In *Advances in Cryptology - EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 407–424. Springer, 2009.

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *Advances in Cryptology - ASIACRYPT 2013*, volume 8270 of *LNCS*, pages 280–300. Springer, 2013.

[BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In *Advances in Cryptology - CRYPTO 2014*, volume 8616 of *LNCS*, pages 480–499. Springer, 2014.

[CGP$^+$13] Cagatay Capar, Dennis Goeckel, Kenneth G. Paterson, Elizabeth A. Quaglia, Don Towsley, and Murtaza Zafer. Signal-flow-based analysis of wireless security protocols. *Information and Computation*, 226:37–56, 2013.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In *Advances in Cryptology - EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 523–552. Springer, 2010.

[Cor00] Jean-Sébastien Coron. On the exact security of full domain hash. In *Advances in Cryptology - CRYPTO 2000*, volume 1880 of *LNCS*, pages 229–235, 2000.

[DE06] Régis Dupont and Andreas Enge. Provably secure non-interactive key distribution based on pairings. *Discrete Applied Mathematics*, 154(2):270–276, 2006.

[DH76] Whitefield Diffie and Martin E. Hellman. New directions in cryptograpgy. *IEEE Transactions on Infomation Theory*, 22(6):644–654, 1976.

[FF13] Marc Fischlin and Nils Fleischhacker. Limitations of the meta-reduction technique: The case of schnorr signatures. In *Advances in Cryptology - EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 444–460. Springer, 2013.

[FHPS13] Eduarda S. V. Freire, Dennis Hofheinz, Kenneth G. Paterson, and Christoph Striecks. Programmable hash functions in the multilinear setting. In *CRYPTO 2013*, volume 8042 of *LNCS*, pages 513–530. Springer, 2013.

[Fis12] Marc Fischlin. Black-box reductions and separations in cryptography. In *AFRICACRYPT 2012*, volume 7374 of *LNCS*, pages 413–422. Springer, 2012.

[FLR$^+$10] Marc Fischlin, Anja Lehmann, Thomas Ristenpart, Thomas Shrimpton, Martijn Stam, and Stefano Tessaro. Random oracles with(out) programmability. In *Advances in Cryptology - ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 303–320. Springer, 2010.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO 1986*, volume 263 of *LNCS*, pages 186–194, 1986.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *Advances in Cryptology - EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 1–17. Springer, 2013.

[GGH$^+$13b] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013*, pages 40–49. IEEE Computer Society, 2013.

[GGH$^+$13c] Sanjam Garg, Craig Gentry, Shai Halevi, Amit Sahai, and Brent Waters. Attribute-based encryption for circuits from multilinear maps. In *Advances in Cryptology - CRYPTO 2013*, volume 8043 of *LNCS*, pages 479–499. Springer, 2013.

[Hof14] Dennis Hofheinz. Fully secure constrained pseudorandom functions using random oracles, 2014. http://eprint.iacr.org/2014/372.

[HSW14] Susan Hohenberger, Amit Sahai, and Brent Waters. Replacing a random oracle: Full domain hash from indistinguishability obfuscation. In *Advances in Cryptology - EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 201–220. Springer, 2014.

[KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS 2013*, pages 669–684. ACM, 2013.

[PS09] Kenneth G. Paterson and Sriramkrishnan Srinivasan. On the relations between non-interactive key distribution, identity-based encryption and trapdoor discrete log groups. *Des. Codes Cryptography*, 52(2):219–241, 2009.

[SOK00] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. *The 2000 Symposium on Cryptography and Information Security, Japan*, 45:26–28, 2000.

[Zha14] Mark Zhandry. How to avoid obfuscation using witness prfs. IACR Cryptology ePrint Archive, Report 2014/301, 2014. http://eprint.iacr.org/2014/301.

[ZZC$^+$14] Jiang Zhang, Zhenfeng Zhang, Yu Chen, Yanfei Guo, and Zongyang Zhang. Black-box separations for one-more (static) problems and its generalizations. Accepted by ASIACRYPTO 2014, 2014.

A Review of Standard Primitives

A.1 Indistinguishability Obfuscation

We recall the definition of indistinguishability obfuscator from [GGH$^+$13b] as below.

Definition A.1 (Indistinguishability Obfuscator ($i\mathcal{O}$))**.** A uniform PPT machine $i\mathcal{O}$ is called an indistinguishability obfuscator for a circuit class $\{\mathcal{C}_\kappa\}$ if the following properties satisfied:

- **Functionality Preserving:** For all security parameters $\kappa \in \mathbb{N}$, for all $C \in \mathcal{C}_\kappa$, for all inputs x, we have that:

$$\Pr[C'(x) = C(x) : C' \leftarrow i\mathcal{O}(\kappa, C)] = 1$$

- **Indistinguishability Obfuscation:** For any pairs of PPT adversaries $(\mathcal{S}, \mathcal{D})$, there exists a negligible function α such that if $\Pr[\forall x, C_0(x) = C_1(x) : (C_0, C_1, state) \leftarrow \mathcal{S}(\kappa)] \geq 1 - \alpha(\kappa)$, then we have:

$$|\Pr[\mathcal{D}(state, i\mathcal{O}(\kappa, C_0)) = 1] - \Pr[\mathcal{D}(state, i\mathcal{O}(\kappa, C_1)) = 1]| \leq \alpha(\kappa)$$

In this paper, we are interested in indistinguishability obfuscators for all polynomial-size circuits.

A.2 Constrained PRFs

Recently, the concept of constrained pseudorandom functions[8] was proposed in the concurrent works of Kiayias, Papadopoulos, Triandopoulos and Zacharias [KPTZ13], Boneh and Waters [BW13], and Boyle, Goldwasser and Ivan [BGI14]. More precisely, constrained PRFs are defined as below:

Definition A.2 (Constrained PRFs)**.** A family of constrained PRFs $\mathsf{F} : K \times X \to Y$ is defined over a key space K, a domain X, and a range Y (these sets may be parameterized by the security parameter κ) with respect to a predicate family $P = \{p : X \to \{0, 1\}\}$. It consists of three polynomial-time algorithms KeyGen, $\mathsf{Constrain}$, and Eval satisfying the following properties:

[8] They were alternatively called delegatable PRFs [KPTZ13] and functional PRFs [BGI14]. In this work, we will mostly adopt the terminology of [BW13].

- $\mathsf{KeyGen}(\kappa)$: on input a security parameter κ, output a secret key $k \in K$. As shorthand we will occasionally write $\mathsf{F}_k(x)$ for $\mathsf{F}(k, x)$.
- $\mathsf{Constrain}(k, p)$: on input a secret key k and a predicate $p \in P$, output a constrained key k_p. The key k_p enables the evaluation of $\mathsf{F}(k, x)$ for all x such that $p(x) = 1$ and no other x. As shorthand we will occasionally write $k(p)$ for k_p.
- $\mathsf{Eval}(k_p, x)$: on input a constrained key k_p and an $x \in X$, output $\mathsf{F}(k_p, x)$.

Correctness: For any $k \leftarrow \mathsf{KeyGen}(\kappa)$, any $S \in \mathcal{S}$, any $k_p \leftarrow \mathsf{Constrain}(k, p)$, and any $x \in X$, we have:

$$\mathsf{F}(k_p, x) = \begin{cases} \mathsf{F}(k, x) & \text{if } p(x) = 1 \\ \perp & \text{otherwise} \end{cases}$$

Security: Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary against constrained PRFs and define its advantage as:

$$\mathsf{Adv}_{\mathcal{A}}(\kappa) = \Pr\left[b = b' : \begin{array}{l} (pp, k) \leftarrow \mathsf{KeyGen}(\kappa); \\ (x^*, state) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{constrain}}(\cdot), \mathcal{O}_{\mathsf{eval}}(\cdot)}(pp); \\ y_0^* \xleftarrow{\text{R}} Y, y_1^* \leftarrow \mathsf{F}(k, x^*); \\ b \leftarrow \{0, 1\}; \\ b' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{constrain}}(\cdot), \mathcal{O}_{\mathsf{eval}}(\cdot)}(state, y_b^*); \end{array} \right] - \frac{1}{2},$$

where $\mathcal{O}_{\mathsf{constrain}}(p) = \mathsf{Constrain}(k, p)$, $\mathcal{O}_{\mathsf{eval}}(x) = \mathsf{F}(k, x)$. Both $\mathcal{A}_1$ and $\mathcal{A}_2$ are not allowed to query $\mathcal{O}_{\mathsf{constrain}}(\cdot)$ for p such that $p(x^*) = 1$ and not allowed to query $\mathcal{O}_{\mathsf{eval}}(\cdot)$ for x^*. We say that constrained PRFs are pseudorandom if for any PPT adversary its advantage function $\mathsf{Adv}_{\mathcal{A}}(\kappa)$ is negligible in κ.

A.3 Signatures

We recall the definition of signature as below.

Definition A.3 (Signature)**.** A signature scheme with message space M and signature space Σ consists of three PPT algorithms as follows:

- $\mathsf{KeyGen}(\kappa)$: take as input a security parameter κ, output a verification key vk and a signing key sk. Let M be the message space and Σ be the signature space.
- $\mathsf{Sign}(sk_\sigma, m)$: take as input a signing key sk and a message $m \in M$, output a signature $\sigma \in \Sigma$.
- $\mathsf{Verify}(vk, m, \sigma)$: take as input a verification key vk, a message m, and a signature σ, output 1 indicates "acceptance" and 0 indicates "rejection".

Correctness: For all $(vk, sk) \leftarrow \mathsf{KeyGen}(\kappa)$ and all $m \in M$, we have $\mathsf{Verify}(vk, m, \mathsf{Sign}(sk, m)) = 1$. If (σ, m) satisfies $\mathsf{Verify}(vk, m, \sigma) = 1$, then σ is said to be a valid signature of message m under the verification key vk.

Security: Let $\mathcal{A}$ be an adversary against the signature and define its advantage as:

$$\mathsf{Adv}_{\mathcal{A}}(\kappa) = \Pr\left[\mathsf{Verify}(vk, m^*, \sigma^*) = 1 : \begin{array}{l} (vk, sk) \leftarrow \mathsf{KeyGen}(\kappa); \\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{sign}}(\cdot)}(vk); \end{array} \right],$$

where $\mathcal{O}_{\mathsf{sign}}(m) = \mathsf{Sign}(sk, m)$. $\mathcal{A}$ is not allowed to query $\mathcal{O}_{\mathsf{constrain}}(\cdot)$ for m^*. We say that a signature is existentially unforgeability under adaptive chosen-message attack (EUF-CMA) if for any PPT adversary its advantage function $\mathsf{Adv}_{\mathcal{A}}(\kappa)$ is negligible in κ.