Self-bilinear Map on Unknown Order Groups from Indistinguishability Obfuscation and Its Applications[*]

Takashi Yamakawa[†] Shota Yamada[‡] Goichiro Hanaoka[§] Noboru Kunihiro[¶]

Abstract

A self-bilinear map is a bilinear map where the domain and target groups are identical. In this paper, we introduce a *self-bilinear map with auxiliary information* which is a weaker variant of a self-bilinear map, construct it based on indistinguishability obfuscation and prove that a useful hardness assumption holds with respect to our construction under the factoring assumption. From our construction, we obtain a multilinear map with interesting properties: the level of multilinearity is not bounded in the setup phase, and representations of group elements are compact, i.e., their size is independent of the level of multilinearity. This is the first construction of a multilinear map with these properties. Note, however, that to evaluate the multilinear map, auxiliary information is required. As applications of our multilinear map, we construct multiparty non-interactive key-exchange and distributed broadcast encryption schemes where the maximum number of users is not fixed in the setup phase. Besides direct applications of our self-bilinear map, we show that our technique can also be used for constructing somewhat homomorphic encryption based on indistinguishability obfuscation and the Φ-hiding assumption.

Keywords: self-bilinear map, indistinguishability obfuscation, multilinear map

1 Introduction

1.1 Background

Bilinear maps are an important tool in constructions of many cryptographic primitives, such as identity-based encryption (IBE)[8, 7, 44], attribute-based encryption (ABE) [40, 5, 28], non-interactive zero-knowledge (NIZK) proof systems [29, 30] etc. The bilinear maps which are mainly used in cryptography, are constructed on elliptic curve groups. In these constructions, the target group is different from the domain groups.

This leads to the natural question: is it possible to construct a bilinear map where the domain and target groups are identical? Such a bilinear map is called a *self-bilinear map*, and has previously been studied by Cheon and Lee [14]. They showed that a self-bilinear map is useful to construct cryptographic primitives by highlighting that it can be used for constructing a multilinear map [9]. However, in contrast to this useful property, they also proved an impossibility result: the computational Diffie-Hellman (CDH) assumption cannot hold in a group G of *known prime order* if there exists an efficiently computable self-bilinear map on G. This is undesirable for cryptographic

[*]An extended abstract of this paper appears in the proceedings of CRYPTO 2014. The first and second authors are supported by a JSPS Fellowship for Young Scientists.

[†]The University of Tokyo. yamakawa@it.k.u-tokyo.ac.jp

[‡]National Institute of Advanced Industrial Science and Technology (AIST). yamada-shota@aist.go.jp

[§]National Institute of Advanced Industrial Science and Technology (AIST). hanaoka-goichiro@aist.go.jp

[¶]The University of Tokyo. kunihiro@k.u-tokyo.ac.jp applications. The overview of the proof is as follows. Let $e : G \times G \to G$ be a self-bilinear map and g be a generator of G, then we have $e(g^x, g^y) = e(g, g)^{xy} = g^{cxy}$ where c is an integer such that $e(g, g) = g^c$. Then we can compute g^{xy} by computing c-th root of $e(g^x, g^y)$ since G is a prime and known order group.[1] However, their impossibility result cannot be applied for the case that G is a *composite and unknown* order group. This is the setting we focus on in this paper.

1.2 Our Contribution

In this paper, we consider a group of composite and unknown order and construct a *self-bilinear map with auxiliary information* which is a weaker variant of a self-bilinear map, by using indistinguishability obfuscation [20]. Though our self-bilinear map with auxiliary information has a limited functionality compared with a self-bilinear map, we show that it is still useful to construct various cryptographic primitives. Especially, it is sufficient to instantiate some multilinear-map-based cryptographic primitives such as multiparty non-interactive key exchange (NIKE), broadcast encryption and attribute-based encryption for circuits. Our multiparty NIKE and distributed broadcast encryption schemes are the first schemes where the number of users is not fixed in the setup phase. We also show that our technique can be used for constructing a somewhat homomorphic encryption scheme for NC^1 circuits.

Applications of our self-bilinear map with auxiliary information. As applications of our self-bilinear map with auxiliary information, we construct a multilinear map. From our construction, we obtain multiparty NIKE, distributed broadcast encryption and ABE for circuits schemes. The details follow.

- **Multilinear map.** We can construct a multilinear map by iterated usage of a self-bilinear map. Since our variant of a self-bilinear map in this paper requires auxiliary information to compute the map, the resulting multilinear map also inherits this property. However, we show that it is sufficient to replace existing multilinear maps in some applications which are given below. Moreover, our multilinear map has an interesting property that existing multilinear maps do not have: the level of multilinearity is *not bounded in the instance generation phase* and representations of group elements are *compact*, i.e., their sizes are independent of the level of multilinearity.

- **Multiparty NIKE.** We construct a multiparty NIKE scheme where the maximum number of users is not fixed in the setup phase. In particular, the size of both the public parameters and a public key generated by a user are independent of the number of users. The construction is a natural extension of the Diffie-Hellman key exchange by using our multilinear map [17, 9]. We note that [11] also constructed multiparty NIKE schemes based on indistinguishability obfuscation. However, in their schemes, the setup algorithm or key generation algorithm have to take the number of users as input unlike ours. On the other hand, our scheme requires a trusted setup whereas theirs does not.

- **Distributed broadcast encryption.** Distributed broadcast encryption is broadcast encryption where a user can join the system by himself without the assistance of a (semi) trusted third party holding a master key. We construct a distributed broadcast encryption scheme where the maximum number of users is not fixed in the setup phase based on our

[1]Here, we consider only the case in which c is known. However, [14] proved that the CDH assumption does not hold even if c is unknown as long as G is a group of known prime order.

multiparty NIKE scheme. In particular, the size of both the public parameters and a ciphertext overhead are independent of the number of users. We note that [11] also constructed a distributed broadcast encryption scheme based on indistinguishability obfuscation. However, in their scheme, the setup algorithm have to take the number of users as input unlike ours.

- **ABE for circuits.** We construct an ABE scheme for general circuits by using our multilinear map. The construction is an analogue of the scheme in [21]. Note that this is *not* the first ABE scheme for general circuits based on indistinguishability obfuscation since an indistinguishability obfuscation implies witness encryption [20], and [22] constructed ABE for circuits based on witness encryption. We also note that Gorbunov et al. [27] constructed attribute based encryption for circuit based on the standard learning with errors (LWE) assumption.

The above results can be interpreted as an evidence that our multilinear map can replace existing multilinear maps in some applications since all of the above constructions are simple analogues of known multilinear-map-based constructions.

Besides direct applications of our self-bilinear map with auxiliary information, we construct a somewhat homomorphic encryption scheme by using a similar technique. Our somewhat homomorphic encryption scheme is chosen plaintext (CPA) secure, NC^1 homomorphic and compact under the Φ-hiding assumption.

Note that all known candidate constructions of indistinguishability obfuscation are far from practical, and hence, the above constructions are mostly of theoretical interest.

Technical overview. Here, we give a technical overview of our result. Our basic idea is to avoid the impossibility result of self-bilinear maps which is explained above by considering a group of *composite and unknown order*. Note that even if we consider such a group, many decisional assumptions such as the decisional Diffie-Hellman (DDH) assumption cannot hold if there exists an efficiently computable self-bilinear map on the group. Therefore we consider only computational assumptions such as the CDH assumption. For a Blum integer N, we consider the group $\mathbb{QR}_N^+$ of signed quadratic residues [32]. On this group, we consider a self-bilinear map $e : \mathbb{QR}_N^+ \times \mathbb{QR}_N^+ \to \mathbb{QR}_N^+$ which is defined as $e(g^x, g^y) := g^{2xy}$. The reason why we define it in this manner is that we want to ensure that the CDH assumption holds in $\mathbb{QR}_N^+$, even if e is efficiently computable. That is, even if we can compute $e(g^x, g^y) = g^{2xy}$, it is difficult to compute g^{xy} from it since the Rabin function is hard to invert under the factoring assumption. However, given only the group elements g^x and g^y, we do not know how to compute $e(g^x, g^y)$ efficiently. To address this, we introduce *auxiliary information* τ_y for each element $g^y \in \mathbb{QR}_N^+$ which enables us to compute a map $e(\cdot, g^y)$ efficiently. This leads to the notion of *self-bilinear map with auxiliary information* which we introduce in this paper.

The problem is how to define auxiliary information τ_y which enables us to compute $e(\cdot, g^y)$ efficiently. The most direct approach is to define τ_y as a circuit that computes the $2y$-th power. However, if we define τ_y as a "natural" circuit that computes the $2y$-th power, then we can extract $2y$ from τ_y, and thus we can compute y. This clearly enables us to compute g^{xy}, which breaks the CDH assumption.

A cleverer way is to define τ_y as a circuit that computes the t_y-th power where $t_y = 2y \pm \mathrm{ord}(\mathbb{QR}_N^+)$.[2] In this way, it seems that τ_y does not reveal y since t_y is a "masked" value of $2y$ by $\mathrm{ord}(\mathbb{QR}_N^+)$ which is an unknown odd number. This idea is already used by Seurin [43] to construct a trapdoor DDH group. Actually, he proved that even if t_y is given in addition to g^x and g^y, it is still difficult to compute g^{xy}. In this way, it seems that we can construct a self-bilinear map with

[2]In the definition of t_y, whether $+$ or $-$ is used depends on y. See [43] for more details.

auxiliary information. However, this creates a problem: we do not have an efficient algorithm to compute t_y from y without knowing the factorization of N. If such an algorithm does not exist, then we cannot instantiate many bilinear map-based primitives using the resulting map such as the 3-party Diffie-Hellman key exchange [34].

To overcome the above difficulty, we use *indistinguishability obfuscation*. An indistinguishability obfuscator ($i\mathcal{O}$) is an efficient randomized algorithm that makes circuits C_0 and C_1 computationally indistinguishable if they have exactly the same functionality.

We observe that a circuit that computes the $2y$-th power and a circuit that computes the t_y-th power for an element of $\mathbb{QR}_N^+$ have exactly the same functionality since we have $t_y = 2y \pm \mathrm{ord}(\mathbb{QR}_N^+)$. Therefore if we obfuscate these circuits by $i\mathcal{O}$, then the resulting circuits are computationally indistinguishable. Then we define auxiliary information τ_y as an obfuscation of a circuit that computes the $2y$-th power. With this definition, it is clear that τ_y can be computed from y efficiently, and the above mentioned problem is solved. Moreover, τ_y is computationally indistinguishable from an obfuscation of a circuit that computes the t_y-th power. Therefore it must still be difficult to compute g^{xy} even if τ_y is given in addition to g^x and g^y.

Thus we obtain a self-bilinear map with auxiliary information on $\mathbb{QR}_N^+$ while ensuring that the auxiliary information does not allow the CDH assumption to be broken. Moreover, by extending this, we can prove that an analogue of multilinear CDH assumption holds with respect to a multilinear map which is constructed from our self-bilinear map with auxiliary information based on $i\mathcal{O}$ and the factoring assumption.

1.3 Related Work

In cryptography, bilinear maps on elliptic curves were first used for breaking the discrete logarithm problem on certain curves [37]. The first constructive cryptographic applications of a bilinear map are given in [34, 42, 8]. Since then, many constructions of cryptographic primitives based on a bilinear map have been proposed.

Boneh and Silverberg [9] considered a multilinear map which is an extension of a bilinear map, and showed its usefulness for constructing cryptographic primitives though they did not give a concrete construction of multilinear maps. Garg et al. [18] proposed a candidate construction of multilinear maps based on ideal lattices. Coron et al. [15] proposed another construction over the integers. We note that some cryptanalysis on these schemes have been discussed [13, 23, 10, 16]

The notion of indistinguishability obfuscation was first proposed by Barak et al. [3]. The first candidate construction of indistinguishability obfuscation was proposed by Garg et al. [20], followed by many works [39, 12, 2, 1, 24]. Since then, many applications of indistinguishability obfuscation have been proposed [41, 11, 33, 26, 19, 31, 38].

2 Preliminaries

2.1 Notations

We use $\mathbb{N}$ to denote the set of all natural numbers, and $[n]$ to denote the set $\{1, \ldots n\}$ for $n \in \mathbb{N}$. If S is a finite set, then we use $x \overset{\$}{\leftarrow} S$ to denote that x is chosen uniformly at random from S. If $\mathcal{A}$ is an algorithm, we use $x \leftarrow \mathcal{A}(y; r)$ to denote that x is output by $\mathcal{A}$ whose input is y and randomness is r. We often omit r. We say that a function $f(\cdot) : \mathbb{N} \to [0, 1]$ is negligible if for all positive polynomials $p(\cdot)$ and all sufficiently large $\lambda \in \mathbb{N}$, we have $f(\lambda) < 1/p(\lambda)$. We say f is overwhelming if $1 - f$ is negligible. We say that an algorithm $\mathcal{A}$ is efficient if there exists a polynomial p such that the running time of $\mathcal{A}$ with input length λ is less than $p(\lambda)$. For two integers $x \neq 0$ and y, we say that x and y are negligibly close if $|x-y|/x$ is negligible. For a set S and a random variable x over S, we say that x is almost random on S if the statistical distance between the distribution of x and the uniform distribution on S is negligible. When we treat a circuit, we use the similar notation as in [4, 21]. For a circuit f with input length n which has v wires, We identify the set of wires with $[v]$ and input wires with $[n]$, and we label the output wire by v. For a wire w which is an output wire of a gate, we denote the first input incoming wire of the gate by $A(w)$ and the second incoming wire of the gate by $B(w)$. We use λ to denote the security parameter

2.2 Indistinguishability Obfuscator

Here, we recall the definition of an indistinguishability obfuscator [20, 41].

Definition 1. *(Indistinguishability Obfuscator.) Let C_λ be the class of circuits of size at most λ. An efficient randomized algorithm $i\mathcal{O}$ is called an indistinguishability obfuscator for P/poly if the following conditions are satisfied:*

- *For all security parameters $\lambda \in \mathbb{N}$, for all $C \in \mathcal{C}_\lambda$, we have that*

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

- *For any (not necessarily uniform) efficient algorithm $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, there exists a negligible function α such that the following holds: if $\mathcal{A}_1(1^\lambda)$ always outputs (C_0, C_1, σ) such that we have $C_0, C_1 \in \mathcal{C}_\lambda$ and $\forall x\ C_0(x) = C_1(x)$, then we have*

$$\begin{aligned} &|\Pr[\mathcal{A}_2(\sigma, i\mathcal{O}(\lambda, C_0)) = 1 : (C_0, C_1, \sigma) \leftarrow \mathcal{A}_1(1^\lambda)] \\ &- \Pr[\mathcal{A}_2(\sigma, i\mathcal{O}(\lambda, C_1)) = 1 : (C_0, C_1, \sigma) \leftarrow \mathcal{A}_1(1^\lambda)]| \le \alpha(\lambda) \end{aligned}$$

Note that a candidate construction of $i\mathcal{O}$ that satisfies the above definition is given in [20].

2.3 Group of Signed Quadratic Residues

Here, we recall the definition and some properties of a group of signed quadratic residues [32] that we mainly work with in this paper. An integer $N = PQ$ is called a Blum integer if P and Q are distinct primes with the same length and $P \equiv Q \equiv 3 \bmod 4$ holds. Let $\mathsf{RSAGen}(1^\lambda)$ be an efficient algorithm which outputs a random ℓ_N-bit Blum integer $N = PQ$ and its factorization (P, Q) such that all prime factors of $\Phi(N)/4 = (P-1)(Q-1)$ are pair wise distinct and at least $\delta\ell_N$ bit integers for some positive constant δ like in [32]. We say that the factoring assumption holds with respect to RSAGen if for any efficient adversary $\mathcal{A}$, $\Pr[x \in \{P, Q\} : (N, P, Q) \leftarrow \mathsf{RSAGen}(1^\lambda), x \leftarrow \mathcal{A}(1^\lambda, N)]$ is negligible. We define the group of quadratic residues as $\mathbb{QR}_N := \{u^2 : u \in {\mathbb{Z}_N}^*\}$. Note that $\mathbb{QR}_N$ is a cyclic group of order $(P-1)(Q-1)/4$ and a random element of $\mathbb{QR}_N$ is a generator of the group with overwhelming probability if N is output by $\mathsf{RSAGen}(1^\lambda)$.

For any subgroup $H \in \mathbb{Z}_N^*$, we define its signed group as $H^+ := \{|x| : x \in H\}$ where $|x|$ is the absolute value of x when it is represented as an element of $\{-(N-1)/2, \ldots, (N-1)/2\}$. This is certainly a group by defining a multiplication as $x \circ y := |(xy \bmod N)|$ for $x, y \in H^+$. For simplicity, we often denote multiplications on H^+ as usual multiplication when it is clear that we are considering a signed group. If H is a subgroup of $\mathbb{QR}_N$, then $H \cong H^+$ by the natural projection since $-1 \notin \mathbb{QR}_N$. In particular, $\mathbb{QR}_N^+$ is a cyclic group of order $(P-1)(Q-1)/4$. We call $\mathbb{QR}_N^+$ a group of signed quadratic residues. A remarkable property of $\mathbb{QR}_N^+$ is that it is efficiently recognizable. That is, there exists an efficient algorithm that determines whether a given string is an element of $\mathbb{QR}_N^+$ or not [32].

3 Self-bilinear Maps

In this section, we recall the definition of a self-bilinear map [14]. Next, we introduce the notion of *self-bilinear map with auxiliary information* which is a weaker variant of a self-bilinear map. Finally we define hardness assumptions with respect to a multilinear map which is constructed from a self-bilinear map.

3.1 Definition of a Self-bilinear Map

First, we recall the definition of a self-bilinear map. A self-bilinear map is a bilinear map where the domain and target groups are identical. The formal definition is as follows.

Definition 2. *(Self-bilinear Map [14]) For a cyclic group G, a self-bilinear map $e : G \times G \to G$ has the following properties.*

- *For all $g_1, g_2 \in G$ and $\alpha \in \mathbb{Z}$, it holds that*

$$e(g_1^{\alpha}, g_2) = e(g_1, g_2^{\alpha}) = e(g_1, g_2)^{\alpha}.$$

- *The map e is non-degenerate, i.e, if $g_1, g_2 \in G$ are generators of G, then $e(g_1, g_2)$ is a generator of G.*

As shown in [14], we can construct an n-multilinear map for any integer $n \geq 2$ from a self-bilinear map e. We denote this n-multilinear map by e_n. This can be seen by easy induction: suppose that an n-multilinear map e_n can be constructed from a self-bilinear map e, then we can construct an $(n+1)$-multilinear map e_{n+1} by defining

$$e_{n+1}(g_1, \ldots, g_n, g_{n+1}) := e(e_n(g_1, \ldots, g_n), g_{n+1}).$$

3.2 Self-bilinear Map with Auxiliary Information

We usually expect a self-bilinear map to be efficiently computable for cryptographic applications. However, here we relax this requirement so that the map is efficiently computable only if "auxiliary information" is given. That is, when we compute $e(g^x, g^y)$, we require auxiliary information τ_x for g^x or τ_y for g^y. We call this relaxed notion a *self-bilinear map with auxiliary information.* We formalize it as a set of algorithms $\mathcal{SBP} = (\mathsf{InstGen}, \mathsf{AIGen}, \mathsf{Map}, \mathsf{AIMult})$.

$\mathsf{InstGen}(1^{\lambda}) \to \mathsf{params} = (G, e, g)$: $\mathsf{InstGen}$ takes the security parameter 1^{λ} as input and outputs the public parameters params which consists of descriptions of an efficiently recognizable cyclic group G on which the group operation is efficiently computable and a self-bilinear map e on G and an element g of G. We require that g is a generator of G with overwhelming probability and that an approximation $\mathrm{Approx}(G)$ of $\mathrm{ord}(G)$ can be computed efficiently from params, which is negligibly close to $\mathrm{ord}(G)$. By using g and $\mathrm{Approx}(G)$, we can generate an almost uniform element h of G by taking $x \xleftarrow{\$} [\mathrm{Approx}(G)]$ and outputting $h := g^x$. With a slight abuse of notation, we often simply write $h \xleftarrow{\$} G$ to mean the above procedure. Additionally, params specifies sets T_x^{ℓ} of auxiliary information for all integers x and $\ell \in \mathbb{N}$.

$\mathsf{AIGen}(\mathsf{params}, \ell, x) \to \tau_x$: AIGen takes params, an integer $\ell \geq 1$ and an integer $x \in [\mathrm{Approx}(G)]$ as input, and outputs corresponding auxiliary information $\tau_x \in T_x^{\ell}$.

$\mathsf{Map}(\mathsf{params}, g^x, \tau_y) \to e(g^x, g^y)$: Map takes params, $g^x \in G$ and $\tau_y \in \cup_{\ell\in\mathbb{N}} T_y^\ell$ as input and outputs $e(g^x, g^y)$. By using this algorithm iteratively, we can compute $e_n(g^{x_1}, \ldots, g^{x_n})$ if we are given $g^{x_1}, \ldots, g^{x_n}$ and $\tau_{x_1}, \ldots, \tau_{x_n}$. (Note that not all of these elements are required to evaluate the map.)

$\mathsf{AIMult}(\mathsf{params}, \ell, \tau_x, \tau_y) \to \tau_{x+y}$: AIMult takes params, ℓ, $\tau_x \in T_x^{\ell_1}$, $\tau_y \in T_y^{\ell_2}$ such that $\ell > \max\{\ell_1, \ell_2\}$ as input and outputs $\tau_{x+y} \in T_{x+y}^\ell$.

In addition to the above algorithms, we require for $\mathcal{SBP}$ to satisfy the following property.

Indistinguishability of auxiliary information. We require that any efficient algorithm which is given auxiliary information cannot tell whether it is generated by AIGen or AIMult. More formally, for any $\mathsf{params} \leftarrow \mathsf{InstGen}(1^\lambda)$, $\ell \in \mathbb{N}$ (which does not depend on λ), natural numbers $\ell_1, \ell_2 < \ell$, integers x, y and z (which are polynomially bounded in λ), such that $z \in [\mathrm{Approx}(G)]$ and $z \equiv x + y \bmod \mathrm{ord}(G)$, and auxiliary information $\tau_x \in T_x^{\ell_1}$ and $\tau_y \in T_y^{\ell_2}$, the following two distributions are computationally indistinguishable:

$$\mathcal{D}_1 = \{\tau_z : \tau_z \leftarrow \mathsf{AIGen}(\mathsf{params}, \ell, z)\}$$

$$\mathcal{D}_2 = \{\tau_{x+y} : \tau_{x+y} \leftarrow \mathsf{AIMult}(\mathsf{params}, \ell, \tau_x, \tau_y)\}.$$

Remark 1. *A level ℓ of auxiliary information grows by at least 1 when AIMult is applied. One can think of it as an analogue of a noise level in the GGH graded encoding [21]. In our construction, the size of auxiliary information grows exponentially in a level ℓ. Therefore an efficient algorithm can only handle auxiliary information of a constant level. Actually, in our applications in this paper, ℓ is set at most 2.*

Remark 2. *One can also define another efficient algorithm that computes auxiliary information for $e(g^x, g^y)$ from that for g^x and g^y. Actually, we can easily add this algorithm to our construction given in Sec. 4. This may be useful for some cryptographic applications. However, it is not used in this paper, therefore we omit it for simplicity.*

3.3 Hardness Assumptions

For cryptographic use, we introduce some hardness assumptions. We use $\mathcal{SBP}$ to construct a multilinear map, and thus our hardness assumptions are associated with a multilinear map which is constructed from $\mathcal{SBP}$. In the following, we let $\mathcal{SBP} = (\mathsf{InstGen}, \mathsf{AIGen}, \mathsf{Map}, \mathsf{AIMult})$ be self-bilinear map procedures. First, we define the multilinear computational Diffie-Hellman with auxiliary information (MCDHAI) assumption which is an analogue of the multilinear computational Diffie-Hellman (MCDH) assumption.

Definition 3. *(MCDHAI assumption) For an integer $n \geq 2$, we say that the n-MCDHAI assumption holds with respect to $\mathcal{SBP}$ if for any efficient algorithm $\mathcal{A}$,*

$$\Pr[e_n(g, \ldots, g)^{\Pi_{i=0}^n x_i} \leftarrow \mathcal{A}(\mathsf{params}, g^{x_0}, \ldots, g^{x_n}, \tau_{x_0} \ldots, \tau_{x_n})]$$

is negligible, where $\mathsf{params} = (G, e, g) \leftarrow \mathsf{InstGen}(1^\lambda)$, $x_0, \ldots, x_n \leftarrow [\mathrm{Approx}(G)]$, $\tau_{x_i} \leftarrow \mathsf{AIGen}(\mathsf{params}, 1, x_i)$ *for all $i = 0, 1 \ldots, n$.*

We say that the MCDHAI assumption holds with respect to $\mathcal{SBP}$ if the n-MCDHAI assumption holds with respect to $\mathcal{SBP}$ for any integer n which is polynomially bounded in λ.

We also define the multilinear hashed Diffie-Hellman with auxiliary information (MHDHAI) assumption which is an analogue of the multilinear hashed Diffie-Hellman (MHDH) assumption.

Definition 4. *(MHDHAI assumption) For an integer $n \geq 2$, we say that the n-MHDHAI assumption holds with respect to $\mathcal{SBP}$ and a family of hash functions $\mathcal{H} = \{H : G \to \{0,1\}^k\}$ if for any efficient algorithm $\mathcal{D}$,*

$$\begin{aligned}&|\Pr[1 \leftarrow \mathcal{D}(\mathsf{params}, g^{x_0}, \ldots, g^{x_n}, \tau_{x_0} \ldots, \tau_{x_n}, H, T)|\beta = 1]\\ &- \Pr[1 \leftarrow \mathcal{D}(\mathsf{params}, g^{x_0}, \ldots, g^{x_n}, \tau_{x_0} \ldots, \tau_{x_n}, H, T)|\beta = 0]|\end{aligned}$$

is negligible, where $\mathsf{params} = (G, e, g) \leftarrow \mathsf{InstGen}(1^\lambda)$*,* $x_0, \ldots, x_n \leftarrow [\mathrm{Approx}(G)]$*,* $\tau_{x_i} \leftarrow \mathsf{AIGen}(\mathsf{params}, 1, x_i)$ *for all* $i = 0, 1, \ldots, n$*,* $\beta \xleftarrow{\$} \{0,1\}$ *and* $T \xleftarrow{\$} \{0,1\}^k$ *if* $\beta = 0$*, and otherwise* $T = H(e_n(g, \ldots, g)^{\Pi_{i=0}^n x_i})$*. We say that the MHDHAI assumption holds with respect to* $\mathcal{SBP}$ *and* $\mathcal{H}$ *if the* n*-MHDHAI assumption holds with respect to* $\mathcal{SBP}$ *and* $\mathcal{H}$ *for any integer* n *which is polynomially bounded in* λ*.*

Note that if the MCDHAI assumption holds with respect to $\mathcal{SBP}$ then the MHDHAI assumption holds with respect to $\mathcal{SBP}$ and the Goldreich-Levin hardcore bit function [25].

4 Our Construction of a Self-bilinear Map

In this section, we construct a self-bilinear map with auxiliary information by giving a construction of self-bilinear map procedures $\mathcal{SBP}$. We prove that the MCDHAI assumption holds with respect to $\mathcal{SBP}$ if the factoring assumption holds and there exists an indistinguishability obfuscator for $P/poly$.

4.1 Construction

First we prepare some notations for circuits on $\mathbb{QR}_N^+$.

Notation for circuits on $\mathbb{QR}_N^+$. In the following, for an ℓ_N-bit RSA modulus N and an integer $x \in \mathbb{Z}$, $\mathcal{C}_{N,x}$ denotes a set of circuits $C_{N,x}$ that work as follows. For input $y \in \{0,1\}^{\ell_N}$, $C_{N,x}$ interprets y as an element of $\mathbb{Z}_N$ and returns y^x where the exponentiation is done on $\mathbb{QR}_N^+$ if $y \in \mathbb{QR}_N^+$ and otherwise returns 0^{ℓ_N} (which is interpreted as $\perp$). We define the canonical circuit $\tilde{C}_{N,x}$ in $\mathcal{C}_{N,x}$ in a natural way [3]. For circuits C_1, C_2 whose output can be interpreted as elements of $\mathbb{QR}_N^+$, $\mathsf{Mult}(C_1, C_2)$ denotes a circuit that computes $C_{\mathsf{mult}}(C_1(a), C_2(a))$ for input a where C_{mult} is a circuit that computes a multiplication for elements of $\mathbb{QR}_N^+$. If an input of C_{mult} is not a pair of two elements in $\mathbb{QR}_N^+$, then it outputs 0^{ℓ_N}.

Now we are ready to describe our construction. The construction of $\mathcal{SBP}$ is as follows.

$\mathsf{InstGen}(1^\lambda) \to \mathsf{params} = (N, g)$: Run $\mathsf{RSAGen}(1^\lambda)$ to obtain (N, P, Q), chooses $g \xleftarrow{\$} \mathbb{QR}_N^+$ and outputs $\mathsf{params} = (N, g)$. params defines the underlying group $G := \mathbb{QR}_N^+$, the self bilinear map $e(g^x, g^y) := g^{2xy}$ and $\mathrm{Approx}(G) := (N-1)/4$. For an integer x and $\ell \in \mathbb{N}$, the set T_x^ℓ is defined as $T_x^\ell = \{i\mathcal{O}(M_\ell, C_{N,2x}; r) : C_{N,2x} \in \mathcal{C}_{N,2x}$ such that $|C_{N,2x}| \leq M_\ell, r \in \{0,1\}^*\}$, where M_ℓ is defined later.

$\mathsf{AIGen}(\mathsf{params}, \ell, x) \to \tau_x$: Take the canonical circuit $\tilde{C}_{N,2x} \in \mathcal{C}_{N,2x}$, set $\tau_x \leftarrow i\mathcal{O}(M_\ell, \tilde{C}_{N,2x})$ and output τ_x.

[3]There is flexibility to define the canonical circuit. However, any definition works if the size of $\tilde{C}_{N,x}$ is polynomially bounded in λ and $|x|$.

$\mathsf{Map}(\mathsf{params}, g^x, \tau_y) \to e(g^x, g^y)$: Compute $\tau_y(g^x)$ and output it. (Recall that τ_y is a circuit that computes the $2y$-th power for an element of $\mathbb{QR}_N^+$.)

$\mathsf{AIMult}(\mathsf{params}, \ell, \tau_x, \tau_y) \to \tau_{x+y}$: Compute $\tau_{x+y} \leftarrow i\mathcal{O}(M_\ell, \mathsf{Mult}(\tau_x, \tau_y))$ and output it.

Definition of M_ℓ. M_ℓ represents an upper bound of the size of a circuit which is obfuscated by $i\mathcal{O}$ when auxiliary information with level ℓ is generated (by AIGen or AIMult). It can be defined recursively as follows. We let $M_1 := \max_{x \in [(N-1)/4]}\{|\tilde{C}_{N,2x}|\}$ and $M_{\ell+1} := 2\mathsf{poly}(M_\ell, \lambda) + |C_{\mathsf{Mult}}|$ for $\ell \geq 1$ where poly is a polynomial that satisfies $|i\mathcal{O}(M, C)| < \mathsf{poly}(M, \lambda)$ for any integer M and circuit C such that $|C| < M$.

Indistinguishability of auxiliary information. If $z \equiv x + y \bmod \mathrm{ord}(\mathbb{QR}_N^+)$ holds, then $C_{N,2z}$ and $\mathsf{Mult}(\tau_x, \tau_y)$ have exactly the same functionality. This can be seen by observing that $C_{N,2z}$ is a circuit that computes $2z$-th power, $\mathsf{Mult}(\tau_x, \tau_y)$ is a circuit that computes $(2x + 2y)$-th power and we have $2z \equiv 2x + 2y \bmod \mathrm{ord}(\mathbb{QR}_N^+)$. Auxiliary information τ_z generated by AIGen is an obfuscation of $C_{N,2z}$, and τ_{x+y} generated by AIMult is an obfuscation of $\mathsf{Mult}(\tau_x, \tau_y)$. Therefore they are computationally indistinguishable by the property of $i\mathcal{O}$.

4.2 Hardness Assumptions

We prove that the MCDHAI assumption holds with respect to our construction of a self-bilinear map if $i\mathcal{O}$ is an indistinguishability obfuscator for $P/poly$ and the factoring assumption holds. From that, we can immediately see that the MHDHAI assumption also holds with respect to our construction if we use the Goldreich-Levin hardcore bit function [25] as $\mathcal{H}$.

First, we prove that the MCDHAI assumption holds if $i\mathcal{O}$ is an indistinguishability obfuscator for $P/poly$ and the factoring assumption holds.

Theorem 1. *The MCDHAI assumption holds with respect to $\mathcal{SBP}_{\mathsf{Ours}}$ if the factoring assumption holds with respect to* RSAGen *and $i\mathcal{O}$ is an indistinguishability obfuscator for $P/poly$.*

Proof. For an algorithm $\mathcal{A}$ and an integer n (which is polynomially bounded by the security parameter), we consider the following games.

Game 1. This game is the original n-MCDHAI game. More precisely, it is as follows.

$$\begin{array}{l} (N, P, Q) \leftarrow \mathsf{RSAGen}(1^\lambda) \\ g \xleftarrow{\$} \mathbb{QR}_N^+ \\ x_0, \ldots, x_n \xleftarrow{\$} [(N-1)/4] \\ \tau_{x_i} \leftarrow i\mathcal{O}(M_1, \tilde{C}_{N,2x_i}) \text{ for } i = 0, \ldots, n \\ U \leftarrow \mathcal{A}(N, g, g^{x_0}, \ldots, g^{x_n}, \tau_{x_0} \ldots, \tau_{x_n}) \end{array}$$

Game $1'$ This game is the same as Game 1 except that $x_0, \ldots, x_n$ are chosen from $[\mathrm{ord}(\mathbb{QR}_N^+)]$.

Game $2'$ This game is the same as Game $1'$ except that g, $x_0, \ldots, x_n$, $\tau_{x_0}, \ldots, \tau_{x_n}$ are set differently. More precisely, it is as follows.

$$\begin{array}{l} (N, P, Q) \leftarrow \mathsf{RSAGen}(1^\lambda) \\ h \xleftarrow{\$} \mathbb{QR}_N^+ \end{array}$$

$g := h^2$
$x'_0, \ldots, x'_n \xleftarrow{\$} [\mathrm{ord}(\mathbb{QR}_N^+)]$
$g^{x_i} := g^{x'_i} h$ for $i = 0, \ldots, n$
(This implicitly defines $x_i \equiv x'_i + 1/2 \bmod \mathrm{ord}(\mathbb{QR}_N^+)$).
$\tau_{x_i} \leftarrow i\mathcal{O}(M_1, \tilde{C}_{N, 2x'_i+1})$ for $i = 0, \ldots, n$
$U \leftarrow \mathcal{A}(N, g, g^{x_0}, \ldots, g^{x_n}, \tau_{x_0} \ldots, \tau_{x_n})$

Game 2 This game is the same as Game $2'$ except that $x'_0, \ldots, x'_n$ are chosen from $[(N-1)/4]$.

We say that $\mathcal{A}$ wins if it outputs $U = e_n(g, \ldots, g)^{\Pi_{i=0}^n x_i}$. For $i = 1, 2$, we let T_i and T'_i be the events that $\mathcal{A}$ wins in Game i and Game i', respectively. What we want to prove is that $\Pr[T_1]$ is negligible. We prove it by the following lemmas.

Lemma 1. $|\Pr[T_i] - \Pr[T'_i]|$ *is negligible for* $i = 1, 2$

Proof. This follows since $(N-1)/4$ is negligibly close to $\mathrm{ord}(\mathbb{QR}_N^+)$. □

Lemma 2. $|\Pr[T'_1] - \Pr[T'_2]|$ *is negligible if* $i\mathcal{O}$ *is an indistinguishability obfuscator for P/poly.*

Proof. We consider hybrid games $H_0, \ldots H_{n+1}$. A hybrid game H_i is the same as Game $1'$ except that the first i auxiliary information (i.e, $\tau_{x_0}, \ldots, \tau_{x_{i-1}}$) are generated as in Game $2'$. It is clear that H_0 is identical to Game $1'$ and H_{n+1} is identical to Game $2'$. Let S_i be the event that $\mathcal{A}$ wins in Game H_i. It suffices to show that $|\Pr[S_i] - \Pr[S_{i-1}]|$ is negligible by the standard hybrid argument. We construct an algorithm $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ that breaks the security of $i\mathcal{O}$ for the security parameter M_1 by using $\mathcal{A}$ that distinguishes H_i and H_{i-1}.

$\mathcal{B}_1(1^\lambda)$**:** $\mathcal{B}_1$ runs $(N, P, Q) \leftarrow \mathsf{RSAGen}(1^\lambda)$, chooses $h \xleftarrow{\$} \mathbb{QR}_N^+$ and $x_0, \ldots, x_n \xleftarrow{\$} [\mathrm{ord}(\mathbb{QR}_N^+)]$ and sets $g := h^2$. $\mathcal{B}_1$ computes $x'_0, \ldots, x'_n \in \mathrm{ord}(\mathbb{QR}_N^+)$ such that $x_j \equiv x'_j + 1/2 \bmod \mathrm{ord}(\mathbb{QR}_N^+)$ for $j = 0, \ldots, n$. (This can be computed since $\mathcal{B}_1$ knows the factorization of N.) Then $\mathcal{B}_1$ sets $C_0 := \tilde{C}_{N, 2x_{i-1}}$, $C_1 := \tilde{C}_{N, 2x'_{i-1}+1}$ and $\sigma := (N, P, Q, h, g, x_0, \ldots, x_n, x'_0, \ldots, x'_n)$ and outputs (C_0, C_1, σ).

$\mathcal{B}_2(\sigma, C^*)$**:** $\mathcal{B}_2$ sets

$$\tau_{x_j} \leftarrow \begin{cases} i\mathcal{O}(M_1, \tilde{C}_{N, 2x'_j+1}) & \text{if } j = 0, \ldots, i-2 \\ C^* & \text{if } j = i-1 \\ i\mathcal{O}(M_1, \tilde{C}_{N, 2x_j}) & \text{if } j = i, \ldots, n. \end{cases}$$

Then $\mathcal{B}_2$ runs $\mathcal{A}(N, g, g^{x_0}, \ldots, g^{x_n}, \tau_{x_0}, \ldots, \tau_{x_n})$ to obtain U. If we have $U = e_n(g, \ldots, g)^{\Pi_{i=0}^n x_i}$, then $\mathcal{B}_2$ outputs 1, and otherwise outputs 0.

The above completes the description of $\mathcal{B}$. First, we show C_0 and C_1 output by $\mathcal{B}_1$ has completely the same functionality. Since we have $x_j \equiv x'_j + 1/2 \bmod \mathrm{ord}(\mathbb{QR}_N^+)$, we have $2x_j \equiv 2x'_j + 1 \bmod \mathrm{ord}(\mathbb{QR}_N^+)$. Therefore $2x_j$-th power and $2x'_j + 1$-th power return exactly the same value on the group $\mathbb{QR}_N^+$ and thus C_0 and C_1 have exactly the same functionality. We note that each of g^{x_j} ($j = 0, \ldots, n$) is distributed in $\mathbb{QR}_N^+$ independently of each other in all hybrid games H_i for $i = 0, \ldots, n+1$. Therefore $\mathcal{B}$ generates them in exactly the same way as those are generated in the hybrids H_{i-1} and H_i. Then we can see that $\mathcal{B}$ perfectly simulates H_{i-1} if $C^* \leftarrow i\mathcal{O}(M_1, C_0)$ and H_i if $C^* \leftarrow i\mathcal{O}(M_1, C_1)$ from the view of $\mathcal{A}$. If the difference between the probability that $\mathcal{A}$ wins in H_{i-1} and that in H_i is non-negligible, then $\mathcal{B}$ succeeds in distinguish whether C^* is computed as $C^* \leftarrow i\mathcal{O}(M_1, C_0)$ or $C^* \leftarrow i\mathcal{O}(M_1, C_1)$, with non-negligible advantage, and thus breaks the security of $i\mathcal{O}$.

$\square$

Lemma 3. $\Pr[T_2]$ *is negligible if the factoring assumption holds.*

Proof. Assuming that $\mathcal{A}$ wins in Game 2 with non-negligible probability, we construct an algorithm $\mathcal{B}$ that factorizes N. This part is very similar to that in [32, 43] The construction of $\mathcal{B}$ is as follows.

$\mathcal{B}(N)$: $\mathcal{B}$ chooses $h' \overset{\$}{\leftarrow} \mathbb{Z}_N^* \setminus \mathbb{QR}_N^+$, sets $h := |h'^2 \mod N| \in \mathbb{QR}_N^+$ and $g := h^2$ and chooses $x'_0, \ldots, x'_n \overset{\$}{\leftarrow} [(N-1)/4]$. Then $\mathcal{B}$ sets $g^{x_i} := g^{x'_i}h$ and $\tau_{x_i} \leftarrow i\mathcal{O}(M_1, \tilde{C}_{N,2x'_i+1})$ for all $i = 0, \ldots, n$. Then $\mathcal{B}$ runs $\mathcal{A}(N, g, g^{x_0}, \ldots, g^{x_n}, \tau_{x_0} \ldots, \tau_{x_n})$. Let U be the output of $\mathcal{A}$. Then $\mathcal{B}$ computes $X := \Pi_{i=1}^n(2x'_i+1)$ and $v = Ug^{-(x'_0X+(X-1)/2)}$. (Note that X is odd and therefore $(X-1)/2$ is an integer.) Then it outputs $\gcd(h', V)$.

Since $\mathcal{B}$ perfectly simulates Game 2 from the view of $\mathcal{A}$, $\mathcal{A}$ outputs $e_n(g, \ldots, g)^{\Pi_{i=0}^n x_i}$ with non-negligible probability. If it occurs, then we have

$$\begin{aligned} U = e_n(g, \ldots, g)^{\Pi_{i=0}^n x_i} &= g^{2^{n-1}\Pi_{i=0}^n x_i} = h^{2^n \Pi_{i=0}^n x_i} = h^{x_0 \Pi_{i=1}^n 2x_i} \\ &= h^{(x'_0+1/2)\Pi_{i=1}^n(2x'_i+1)} = h^{x'_0X+X/2} = h^{x'_0X+(X-1)/2+1/2} \end{aligned}$$

where we used that $x_i \equiv x'_i+1$ holds for $i = 0, \ldots, n$. Therefore we have $V = h^{1/2}$. Since $V \in \mathbb{QR}_N^+$, h' and V are distinct square roots of h in $\mathbb{QR}_N^+$. Therefore $\gcd(h', V)$ is a non-trivial factor of N. $\square$

Theorem 1 is proven by the above lemmas. $\square$

$\square$

The following is immediate from Theorem 1 and the Goldreich-Levin theorem.

Theorem 2. *The MHDHAI assumption holds with respect to* $\mathcal{SBP}_{\mathsf{Ours}}$ *and the Goldreich-Levin hardcore bit function if the factoring assumption holds with respect to* RSAGen *and* $i\mathcal{O}$ *is an indistinguishability obfuscator for P/poly.*

5 Applications of Our Self-bilinear Map

In Sec. 4, we constructed a self-bilinear map with auxiliary information. In this section, we construct a multilinear map, multiparty NIKE, distributed broadcast encryption and ABE for circuits by using it.

Multilinear map. Here, we consider a multilinear map which is constructed from a self-bilinear map with auxiliary information. As shown in Sec. 3.1 we can construct a multilinear map by iterated usage of a self-bilinear map. However, if we use a self-bilinear map with auxiliary information as a building block, then the resulting multilinear map has a restricted functionality: we need auxiliary information to compute the map. Similarly to self-bilinear map with auxiliary information, we formalize a multilinear map with an auxiliary information as a set of algorithms $\mathcal{MMP} = (\mathsf{MInstGen}, \mathsf{MAIGen}, \mathsf{MMap}, \mathsf{MAIMult})$ and a set R of integers. Actually, this is the almost the same as the definition of $\mathcal{SBP}$, we give a concrete formalization again for completeness.

$\mathsf{MInstGen}(1^\lambda) \to \mathsf{params}$: $\mathsf{InstGen}$ takes the security parameter 1^λ as input and outputs the public parameters params which consists of descriptions of an efficiently recognizable cyclic group G on which the group operation is efficiently computable and a n-multilinear map $e_n : G \times \cdots \times G \to G$ for any $n \geq 2$ and an element g of G. We require that g is a generator of G with overwhelming probability and that an approximation $\mathrm{Approx}(G)$ of $\mathrm{ord}(G)$ can be computed efficiently from params, which is negligibly close to $\mathrm{ord}(G)$. By using g and $\mathrm{Approx}(G)$, we can generate an almost uniform element h of G by taking $x \overset{\$}{\leftarrow} [\mathrm{Approx}(G)]$ and outputting $h := g^x$. With a slight abuse of notation, we often simply write $h \overset{\$}{\leftarrow} G$ to mean the above procedure. Additionally, params specifies sets T_x^ℓ of auxiliary information for all integers x and $\ell \in \mathbb{N}$.

$\mathsf{MAIGen}(\mathsf{params}, \ell, x) \to \tau_x$: AIGen takes params, level ℓ and an integer $x \in R$ as input, and outputs corresponding auxiliary information $\tau_x \in T_x^\ell$.

$\mathsf{MMap}(\mathsf{params}, g^{x_1}, \ldots, g^{x_n}, \tau_{x_1}, \ldots, \tau_{x_{n-1}}) \to e_n(g^{x_1}, \ldots, g^{x_n})$: MMap takes params, $g^{x_1}, \ldots, g^{x_n} \in G$ and $\tau_{x_i} \in \cup_{\ell \in \mathbb{N}} T_{x_i}^\ell$ for all $i \in [n-1]$ as input and outputs $e(g^{x_1}, \ldots, g^{x_n})$.

$\mathsf{MAIMult}(\mathsf{params}, \ell, \tau_x, \tau_y) \to \tau_{x+y}$: $\mathsf{MAIMult}$ takes params, ℓ, $\tau_x \in T_x^{\ell_1}$, $\tau_y \in T_y^{\ell_2}$ such that $\ell > \max\{\ell_1, \ell_2\}$ as input and outputs $\tau_{x+y} \in T_{x+y}^\ell$.

In addition to the above algorithms, we require for $\mathcal{MMP}$ to satisfy the following property.

Indistinguishability of auxiliary information. We require that any efficient algorithm which is given auxiliary information cannot tell whether it is generated by MAIGen or $\mathsf{MAIMult}$. More formally, for any $\mathsf{params} \leftarrow \mathsf{MInstGen}(1^\lambda)$, $\ell \in \mathbb{N}$ (which does not depend on λ), natural numbers $\ell_1, \ell_2 < \ell$, integers x, y and z (which are polynomially bounded in λ), such that $z \in R$ and $z \equiv x + y \bmod \mathrm{ord}(G)$, and auxiliary information $\tau_x \in T_x^{\ell_1}$ and $\tau_y \in T_y^{\ell_2}$, the following two distributions are computationally indistinguishable:

$$\mathcal{D}_1 = \{\tau_z : \tau_z \leftarrow \mathsf{MAIGen}(\mathsf{params}, \ell, z)\}$$

$$\mathcal{D}_2 = \{\tau_{x+y} : \tau_{x+y} \leftarrow \mathsf{MAIMult}(\mathsf{params}, \ell, \tau_x, \tau_y)\}.$$

It is clear that we can construct $\mathcal{MMP}$ from $\mathcal{SBP}$: All algorithms of $\mathcal{MMP}$ except MMap can be exactly the same as the corresponding algorithm of $\mathcal{SBP}$, and MMap can be constructed by an iterated usage of Map of $\mathcal{SBP}$. The MCDH and MHDH assumption with respect to $\mathcal{MMP}$ is defined similarly as those with respect to $\mathcal{SBP}$, and it is clear that if the MCDH (resp. MHDH) assumption holds with respect to $\mathcal{SBP}$, then the MCDH (resp. MHDH) assumption holds with respect to $\mathcal{MMP}$ which is constructed from $\mathcal{SBP}$.

In spite of the limitation that it requires auxiliary information to compute the map, a multilinear map with auxiliary information is sufficient to replace existing multilinear maps in some applications. Moreover, our multilinear map has interesting properties that existing multilinear maps do not have: the level of multilinearity is not bounded in the instance generation phase and representations of group elements are compact, i.e., their sizes are independent of the level of multilinearity. By this property, cryptographic primitives which are constructed from our multilinear map inherit these properties too.

Multiparty NIKE. Here, we construct a multiparty NIKE scheme. The main idea of our construction is to use our multilinear map (with auxiliary information) for the "multiparty" Diffie-Hellman key exchange [17, 9].

First, we formally define multiparty NIKE and its security. A multiparty NIKE scheme consists of the three algorithms (Setup,$\mathsf{Publish}$,KeyGen). Setup takes the security parameter 1^λ as input and outputs the public parameters PP.[4] $\mathsf{Publish}$ takes the public parameters PP and outputs a public key pk and a secret key sk. KeyGen takes the public parameters PP, a secret key sk and a set of public keys $\{pk_j\}_{j=1,\ldots,n}$ as input, and outputs a ℓ_K-bit derived key K. For correctness, we require that we have $K_1 = K_2 = \ldots = K_n$, where $\mathsf{PP} \leftarrow \mathsf{Setup}(1^\lambda)$, $(pk_i, sk_i) \leftarrow \mathsf{Publish}(\mathsf{PP})$, $K_i := \mathsf{KeyGen}(\mathsf{PP}, sk_i, \{pk_j\}_{j=1,\ldots,n})$ for $i = 1, \ldots, n$. We say that a multiparty NIKE scheme is statically secure if for any integer n which is polynomial in the security parameter, for any efficient adversary $\mathcal{A}$, $|\Pr[b \xleftarrow{\$} \mathcal{A}(\mathsf{PP}, \{pk_i\}_{i=1,\ldots,n}, K_b)] - 1/2|$ is negligible, where $\mathsf{PP} \leftarrow \mathsf{Setup}(1^\lambda)$, $(pk_i, sk_i) \leftarrow \mathsf{Publish}(\mathsf{PP})$ for $i = 1, \ldots, n$, $K_1 := \mathsf{KeyGen}(\mathsf{PP}, sk_i, \{pk_j\}_{j=1,\ldots,n})$, $K_0 \xleftarrow{\$} \{0,1\}^{\ell_K}$ and $b \xleftarrow{\$} \{0,1\}$.

Our construction of multiparty NIKE scheme is as follows. Let $\mathcal{H}$ be a family of hash functions.

$\mathsf{Setup}_{\mathsf{NIKE}}(1^\lambda)$: $\mathsf{Setup}_{\mathsf{NIKE}}$ runs $\mathsf{params} = (G, e, g) \leftarrow \mathsf{InstGen}(1^\lambda)$ and chooses $H \xleftarrow{\$} \mathcal{H}$. It outputs $\mathsf{PP} = (\mathsf{params}, H)$ as the public parameter.

$\mathsf{Publish}_{\mathsf{NIKE}}(\mathsf{PP})$**:** It chooses $x \leftarrow [\mathrm{Approx}(G)]$ and sets $\tau_x \leftarrow \mathsf{AIGen}(\mathsf{params}, 1, x)$. It sets $pk := (g^x, \tau_x)$ and $sk := x$, and outputs (pk, sk).

$\mathsf{KeyGen}_{\mathsf{NIKE}}(\mathsf{PP}, sk, \{pk_j\}_{j=1,\ldots,n})$: Let i be the index such that sk corresponds to pk_i, $x_i := sk$ and $(g^{x_j}, \tau_{x_j}) := pk_j$ for $j \in [n]$. Without loss of generality, we assume that $i \neq 1$. It computes recursively as follows. It first sets $k_1 := g^{x_1}$. For $j = 2, \ldots, n$, it computes

$$k_j := \begin{cases} \mathsf{Map}(\mathsf{params}, k_{j-1}, \tau_{x_j}) & \text{if } j \neq i \\ k_{i-1}^{x_i} & \text{if } j = i. \end{cases}$$

Finally, it outputs $K := H(k_n)$ as its derived key.

Remark 3. *If we instantiate H by the Goldreich-Levin hardcore function, then the output length of H is 1-bit, and therefore the length of a key derived by the above NIKE scheme is only 1-bit. A scheme for multi-bit keys can be obtained by running our scheme in parallel. Alternatively, one can use variant of our map which is based on the idea of the Blum-Blum-Shub pseudo-random number generator [6]. The details are given in Appendix A.*

The correctness can be seen easily since a derived key K for a set S of users satisfies $K = H(e_{n-1}(g, \ldots, g)^{\prod_{j=1}^n x_j})$ where $\{(g^{x_j}, \tau_{x_j})\}_{j=1,\ldots,n}$ and $\{x_j\}_{j=1,\ldots,n}$ are public keys and secret keys of users in S.

The security of our NIKE scheme is as follows.

Theorem 3. *This multiparty NIKE scheme is statically secure if the MHDHAI assumption holds with respect to the underlying $\mathcal{SBP}$ and $\mathcal{H}$.*

This is clear from the definitions of the MHDHAI assumption and the static security of multiparty NIKE. Thus if we use our construction of $\mathcal{SBP}$ given in Section 4, then this multiparty NIKE scheme is statically secure if the factoring assumption holds and $i\mathcal{O}$ is an indistinguishability obfuscator for *P/poly*.

[4]We do not include the number of users in the input of Setup. This means that the number of users is unbounded in the setup phase.

Comparison to [11] Boneh and Zhandry [11] also constructed a multiparty key exchange scheme based on an indistinguishability obfuscation, whose construction is totally different from ours. Our scheme is superior to their scheme in regard to the size of public parameter: it is $O(1)$ in our scheme whereas it is $O(poly(n))$ in their scheme where n is the number of users. Moreover, in their scheme, the number of users is fixed in setup phase whereas it is not in our scheme. On the other hand, their scheme has a useful property ours does not have. In their scheme, $\mathsf{Publish}$ can be run independently of Setup. This property enable one to make the scheme"no setup" by letting the "master user" publish the public parameter in addition to his own public key. We note that in their "no setup" scheme, a "master user" has to know the number of users before publishing his public parameter unlike usual multiparty NIKE schemes. Actually, this is not allowed in our formulation because KeyGen does not take the number of users as input. We also note that our scheme is based on slightly stronger assumption than they are. That is, ours is based on the factoring assumption and the existence of an indistinguishability obfuscation whereas theirs is based on the existence of a one-way function and an indistinguishability obfuscation. Thus, these schemes are incomparable.

Distributed broadcast encryption. It is known that a multiparty NIKE scheme can be converted to a *distributed broadcast encryption* scheme [9, 11], where a user can join the system by himself without the assistance of a (semi) trusted third party holding a master key.

The conversion is very simple: The setup algorithm runs $\mathsf{Setup}_{\mathsf{NIKE}}(1^\lambda)$ to obtain PP and publishes it. A user who wants to join the system runs $\mathsf{Publish}_{\mathsf{NIKE}}(\mathsf{PP})$ to obtain (pk, sk), publishes pk as his public key and keeps sk as his secret key. A sender who wants to send a message M to a set S of users plays the role of a user of the underlying NIKE, shares a derived key K with users in S and encrypts M to obtain a ciphertext Ψ by a symmetric key encryption scheme using the key K. A ciphertext consists of S, the sender's public key and Ψ. It is proven that the resulting broadcast encryption scheme is CPA secure if the underlying multiparty NIKE scheme is statically secure. See [11] for more detail.

In our scheme, as in the multiparty NIKE scheme, all algorithms can be run independently of the number of users. In particular, the size of both the public parameters and a ciphertext overhead are independent of the number of users. This is the first distributed broadcast encryption scheme with this property. Note that [11] also constructed distributed broadcast encryption schemes based on indistinguishability obfuscation. However, in their schemes, the setup algorithm takes the number of users as input and a public parameter depends on the number of users unlike ours.

Attribute based encryption for circuits. Here, we construct an attribute based encryption scheme for general circuits. Our construction is an analogue of [21].

First define attribute-based encryption (ABE) and its security. An ABE scheme consists of four algorithms $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$. Setup takes the security parameter 1^λ, the length n of the index as input and upper bound d of circuit depth, and outputs the public parameters PP and a master secret key MSK. Enc takes the public parameters PP, an index $x \in \{0,1\}^n$ and a message M as input, and outputs a ciphertext CT. KeyGen takes a master secret key MSK and a circuit f with a single output gate, and outputs a secret key SK. Dec takes a secret key SK and a ciphertext CT as input, and outputs a message M or $\perp$. For correctness, we require that for all M, $x \in \{0,1\}^n$ and f with depth lower than d such that $f(x) = 1$, $\mathsf{Dec}(SK, CT) = M$ always holds, where $(PP, MSK) \leftarrow \mathsf{Setup}(1^\lambda, n, d)$, $SK = \mathsf{KeyGen}(MSK, f)$ and $CT = \mathsf{Enc}(PP, x, M)$.

Next, we define the security of ABE. Here, we only define the selective security since we only consider it in this paper. For an adversary $\mathcal{A}$, we consider the following game between $\mathcal{A}$ and a challenger. $\mathcal{A}$ first declares the target index x^*. Then the challenger computes $(PP, MSK) \leftarrow$

$\mathsf{Setup}(1^\lambda)$ and gives PP to $\mathcal{A}$. Then $\mathcal{A}$ declares M_0 and M_1. The challenger chooses $b \overset{\$}{\leftarrow} \{0,1\}$ and computes $CT \leftarrow \mathsf{Enc}(PP, x^*, M_b)$. Then it gives CT to $\mathcal{A}$. In the game, $\mathcal{A}$ can query a circuit f such that $f(x^*) = 0$ for key generation oracle, and the oracle returns $\mathsf{KeyGen}(MSK, f)$ to $\mathcal{A}$. Finally, $\mathcal{A}$ outputs b'. We say that $\mathcal{A}$ wins if $b' = b$. We say that an ABE scheme is selectively secure if for any efficient adversary $\mathcal{A}$, tha probability that $\mathcal{A}$ wins is negligibly close to 1/2.

It is known that any general Boolean circuit can be converted to an equivalent monotone layered Boolean circuit [21]. Therefore we only consider ABE for monotone layered circuits. Here, monotone circuit is a circuit where all gates are either AND or OR gates of two inputs, and layered circuit is a circuit where a gate at depth j receive both of its inputs from wires at depth $j-1$. Then we give our construction of an ABE scheme. Let $\mathcal{H}$ be a family of hash functions $H : G \to \{0,1\}^{\ell_H}$.

$\mathsf{Setup}(1^\lambda, n, d)$: It runs $\mathsf{params} = (G, e, g) \leftarrow \mathsf{InstGen}(1^\lambda)$ and chooses $\alpha \leftarrow [\mathrm{Approx}(G)]$, $h_1, \ldots, h_n \overset{\$}{\leftarrow} G$ and $H \overset{\$}{\leftarrow} \mathcal{H}$. Then it outputs $PP := (\mathsf{params}, H, e_{d+1}(g, \ldots, g)^\alpha, h_1, \ldots, h_n)$ and $MSK := (\alpha, PP)$.

$\mathsf{Enc}(\mathsf{PP}, x \in \{0,1\}^n, M \in \{0,1\}^{\ell_H})$: It chooses $s \leftarrow [\mathrm{Approx}(G)]$, sets $\tau_s \leftarrow \mathsf{AIGen}(\mathsf{params}, 1, s)$ and $CT := (M \oplus H((e_{d+1}(g, \ldots, g)^\alpha)^s), \tau_s, h_i^s$ for all i such that $x_i = 1)$, and outputs CT.

$\mathsf{KeyGen}(MSK, f)$: It chooses $r_1, \ldots, r_v \overset{\$}{\leftarrow} [\mathrm{Approx}(G)]$ and sets $K_H := e_d(g, \ldots, g)^{\alpha - r_v}$ where v is the number of wires of f. Next, it generates key component for each wire of f as follows.

- *Input Wire*: If w is an input wire (i.e., depth is 1), then it chooses $z_w \overset{\$}{\leftarrow} [\mathrm{Approx}(G)]$ and sets $K_w := g^{r_w} h_w^{-z_w}$ and $\tau_{z_w} \leftarrow \mathsf{AIGen}(\mathsf{params}, 2, z_w)$. We let (K_w, τ_{z_w}) be the key component for wire w.
- *OR Gate*: If w is an output wire of an OR gate with depth j, then it chooses $a_w, b_w \overset{\$}{\leftarrow} [\mathrm{Approx}(G)]$ and sets $K_{w,1} := e_j(g, \ldots, g)^{r_w - a_w r_{A(w)}}$, $K_{w,2} := e_j(g, \ldots, g)^{r_w - b_w r_{B(w)}} \tau_{a_w} := \mathsf{AIGen}(\mathsf{params}, 2, a_w)$ and $\tau_{b_w} := \mathsf{AIGen}(\mathsf{params}, 2, b_w)$. We let $(K_{w,1}, K_{w,2}, \tau_{a_w}, \tau_{b_w})$ be the key component for wire w.
- *AND Gate*: If w is an output wire of an AND gate with depth j, then it chooses $a_w, b_w \overset{\$}{\leftarrow} [\mathrm{Approx}(G)]$ and sets $K_w := e_j(g, \ldots, g)^{r_w - a_w r_{A(w)} - b_w r_{B(w)}}, \tau_{a_w} := \mathsf{AIGen}(\mathsf{params}, 2, a_w)$ and $\tau_{b_w} := \mathsf{AIGen}(\mathsf{params}, 2, b_w)$. We let $(K_w, \tau_{a_w}, \tau_{b_w})$ be the key component for wire w.

It outputs SK which consists of description of f, K_H and key components for each wire.

$\mathsf{Dec}(PP, SK, CT)$: Let SK be a secret key corresponding to f and CT be a ciphertext corresponding to x. We can correctly decrypt it if $f(x) = 1$. First, it computes $E' := e(g^s, K_H) = e_{d+1}(g, \ldots, g)^{s(\alpha - r_v)}$ by using τ_s. Then it computes as follows for all wires from wires with lower depth.

- *Input Wire*: Let w be an input wire. If x_w=1 holds, then it computes $E_w := e(g^s, K_w) e(g^{z_w}, h_w^s) = e(g^s, g^{r_w} h_w^{-z_w}) e(g^{z_w}, h_w^s) = e(g, g)^{s r_w}$ by using τ_s and τ_{z_w}.
- *OR gate*: Let w be an output wire of an OR gate with depth j. If $f_w(x) = 1$ holds, then it works as follows. In this case we have $f_{A(w)} = 1$ or $f_{B(w)} = 1$. If we have $f_{A(w)} = 1$, then it computes $E_w := e(g^{a_w}, E_{A(w)}) e(g^s, K_{w,1}) = e(g^{a_w}, e_j(g, \ldots, g)^{s r_{A(w)}})$ $e(g^s, e_j(g, \ldots, g)^{r_w - a_w r_{A(w)}}) = e_{j+1}(g, \ldots, g)^{s r_w}$ by using τ_{a_w} and τ_s. If we have $f_{A(w)} \neq 1$, (in this case, we have $f_{B(w)} = 1$,) then it computes $E_w := e(g^{b_w}, E_{B(w)}) e(g^s, K_{w,2})$

$= e(g^{b_w}, e_j(g,\dots,g)^{sr_{B(w)}})e(g^s, e_j(g,\dots,g)^{r_w - b_w r_{B(w)}}) = e_{j+1}(g,\dots,g)^{sr_w}$ by using τ_{b_w} and τ_s.

- *AND gate*: Let w be an output gate of an AND gate with depth j. If $f_w(x) = 1$ holds, i.e, we have $f_{A(w)} = 1$ and $f_{B(w)} = 1$, then it works as follows. It computes $E_w := e(E_{A(w)}, g^{a_w})e(E_{B(w)}, g^{b_w})e(g^s, K_w) = e(e_j(g,\dots,g)^{sr_{A(w)}}, g^{a_w})e(e_j(g,\dots,g)^{sr_{B(w)}}, g^{b_w})$ $e(g^s, e_j(g,\dots,g)^{r_w - a_w r_{A(w)} - b_w r_{B(w)}}) = e_{j+1}(g,\dots,g)^{sr_w}$ by using τ_{a_w}, τ_{b_w} and τ_s.

If $f(x) = 1$ holds, then for output wire v with depth d, it can compute $E_v = e_{d+1}(g,\dots,g)^{sr_v}$. Next, it computes $E'' := E'E_v = e_{d+1}(g,\dots,g)^{s(\alpha - r_v)}e_{d+1}(g,\dots,g)^{sr_v} = e_{d+1}(g,\dots,g)^{s\alpha}$. Finally, it outputs $M := C_M \oplus H(E'')$.

The correctness of the scheme is already checked in the above description. The security of the scheme is as follows.

Theorem 4. *Our ABE scheme is selectively secure if the MHDHAI assumption holds with respect to the underlying multilinear map and $\mathcal{H}$.*

Proof. We construct an algorithm $\mathcal{D}$ that breaks the $d+1$-MHDHAI assumption by using $\mathcal{A}$ that breaks the ABE scheme. The construction of $\mathcal{D}$ is as follows. We note that the label of an instance of the MHDHAI problem is different from that in Definition 3 for notational convenience.

$\mathcal{D}(\mathsf{params}, g^s, g^{c_1}, \dots, g^{c_{d+1}}, \tau_s, \tau_{c_1}, \dots, \tau_{c_{d+1}}, H, T)$:

Setup. Let $x^* = (x_1^*, \dots, x_n^*)$ be a target input declared by $\mathcal{A}$. $\mathcal{D}$ sets $\alpha := \Pi_{i=1}^{d+1} c_i$. Then it can computes $e_{d+1}(g,\dots,g)^{\alpha}$ by iterated usage of Map by using $\tau_{c_1}, \dots, \tau_{c_{d+1}}$. It chooses $y_i \xleftarrow{\$} [\text{Approx}(G)]$ for $i = 1, \dots, n$, and sets

$$h_i := \begin{cases} g^{y_i} \text{ if } x_i^* = 1 \\ g^{y_i + c_1} \text{ if } x_i^* = 0 \end{cases}$$

and $PP := (\mathsf{params}, H, e_{d+1}(g,\dots,g)^{\alpha}, h_1, \dots, h_n)$. Then it gives PP to $\mathcal{A}$.

Challenge Ciphertext. For messages M_0, M_1 which are declared by $\mathcal{A}$, $\mathcal{D}$ chooses $b \xleftarrow{\$} \{0,1\}$, and sets $CT := (M_b \oplus T, \tau_s, (g^s)^{y_i}$ for $i \in [d+1]$ such that $x_i^* = 1)$. It gives CT to $\mathcal{A}$ as a challenge ciphertext.

Key Generation. For $\mathcal{A}$'s key query f such that $f(x^*) = 0$, $\mathcal{D}$ computes as follows for all wires from wires with lower depth.

- *Input Wire* Let w be an input wire. If $x_w^* = 1$, then $\mathcal{D}$ chooses $z_w, r_w \xleftarrow{\$} [\text{Approx}(G)]$, and computes $K_w := g^{r_w} h_w^{-z_w}$ and $\tau_{z_w} := \mathsf{AIGen}(\mathsf{params}, 2, z_w)$. If $x_w^* = 0$, then it chooses $\eta_w, \nu_w \xleftarrow{\$} [\text{Approx}(G)]$, and computes $z_w := c_2 + \nu_w$, $r_w := c_1 c_2 + \eta_w$, $K_w := g^{-c_2 y_w + \eta_w - (y_w + c_1)\nu_w}$ and $\tau_{z_w} := \mathsf{AIMult}(\mathsf{params}, 2, \tau_{c_2}, \mathsf{AIGen}(G, 1, \nu_w))$.
- *OR Gate*: Let w be an output wire of an OR gate with depth j. If $f_w(x^*) = 1$, then $\mathcal{D}$ chooses $a_w, b_w \xleftarrow{\$} [\text{Approx}(G)]$ and computes $K_{w,1} := e_j(g,\dots,g)^{r_w - a_w r_{A(w)}}$, $K_{w,2} := e_j(g,\dots,g)^{r_w - b_w r_{B(w)}}$, $\tau_{a_w} := \mathsf{AIGen}(\mathsf{params}, 2, a_w)$ and $\tau_{b_w} := \mathsf{AIGen}(\mathsf{params}, 2, b_w)$.
  If $f_w(x^*) = 0$, then it chooses $\psi_w, \phi_w, \eta_w \xleftarrow{\$} [\text{Approx}(G)]$, sets $a_w := c_{j+1} + \psi_w$, $b_w :=$

$c_{j+1} + \phi_w$ and $r_w := \Pi_{i=1}^{j+1} c_i + \eta_w$, and computes

$$\begin{aligned} K_{w,1} &:= e_j(g,\ldots,g)^{\eta_w - \psi_w \eta_{A(w)} - c_{j+1}\eta_{A(w)} - \psi_w \Pi_{i=1}^{j} c_i} \\ K_{w,2} &:= e_j(g,\ldots,g)^{\eta_w - \phi_w \eta_{B(w)} - c_{j+1}\eta_{B(w)} - \phi_w \Pi_{i=1}^{j} c_i} \\ \tau_{a_w} &:= \mathsf{AIMult}(\mathsf{params}, 2, \tau_{c_{j+1}}, \mathsf{AIGen}(\mathsf{params}, 1, \psi_w)) \\ \tau_{b_w} &:= \mathsf{AIMult}(\mathsf{params}, 2, \tau_{c_{j+1}}, \mathsf{AIGen}(\mathsf{params}, 1, \phi_w)). \end{aligned}$$

- *AND Gate*: Let w be an output wire of an AND gate with depth j. If $f_w(x^*) = 1$, then $\mathcal{D}$ chooses $a_w, b_w \xleftarrow{\$} [\mathrm{Approx}(G)]$, computes $K_w := e_j(g,\ldots,g)^{r_w - a_w r_{A(w)} - b_w r_{B(w)}}$, $\tau_{a_w} := \mathsf{AIGen}(\mathsf{params}, 2, a_w)$, $\tau_{b_w} := \mathsf{AIGen}(\mathsf{params}, 2, b_w)$. If $f_w(x^*) = 0$, then it works as follows. If $f_{A(w)}(x^*) = 0$, then it chooses $\psi_w, \phi_w, \eta_w \xleftarrow{\$} [\mathrm{Approx}(G)]$, sets $a_w := c_{j+1} + \psi_w$, $b_w := \phi_w$ and $r_w := \Pi_{i=1}^{j+1} c_i + \eta_w$, and computes

$$\begin{aligned} K_w &:= e_j(g,\ldots,g)^{\eta_w - \psi_w \eta_{A(w)} - \phi_w r_{B(w)} - c_{j+1}\eta_{A(w)} - \psi_w \Pi_{i=1}^{j} c_i} \\ \tau_{a_w} &:= \mathsf{AIMult}(\mathsf{params}, 2, \tau_{c_{j+1}}, \mathsf{AIGen}(\mathsf{params}, 1, \psi_w)) \\ \tau_{b_w} &:= \mathsf{AIGen}(\mathsf{params}, 2, \phi_w). \end{aligned}$$

  If $f_{A(w)}(x^*) = 1$ and $f_{B(w)}(x^*) = 0$, it works symmetric to what is above, with the roles of a_w and b_w reversed.

Remark 4. *$\mathcal{D}$ can actually simulate the key generation oracle as the above since $e_j(g,\ldots,g)^{\Pi_{i=1}^{j} c_i}$ can be computed by iterated usage of* Map *by using $\tau_{c_1},\ldots,\tau_{c_j}$. Note that it need not compute $e_j(g,\ldots,g)^{\Pi_{i=1}^{j+1} c_i}$ thanks to the cancellation technique.*

Since we have $f(x^*) = 0$, for the output wire v, r_v is defined as $\Pi_{i=1}^{d+1} c_i + \eta_v$. Therefore it can generate $K_H := e_d(g,\ldots,g)^{-\eta_v}$ by the cancellation. Thus, it can simulate the key generation oracle.

Guess. Finally, when $\mathcal{A}$ outputs b', $\mathcal{D}$ outputs 1 if $b = b'$, and otherwise 0.

The above completes the description of $\mathcal{D}$. We can easily see that if $\beta = 1$, then the CPA game and the simulated environment are computationally indistinguishable from the view of $\mathcal{A}$ by the indistinguishablity of auxiliary information. (Recall that β is a random coin that determines T is random or not.) On the other hand, if $\beta = 0$, then information of b is completely hidden and therefore the probability that $\mathcal{A}$ predicts b is equal to 1/2. Therefore $\Pr[1 \leftarrow \mathcal{D}|\beta = 1] - \Pr[1 \leftarrow \mathcal{D}|\beta = 0]$ is non-negligible if $\mathcal{A}$ breaks the CPA security of the scheme. □

6 Homomorphic Encryption

In this section, we construct a somewhat homomorphic encryption scheme by using an indistinguishability obfuscator. This is not a direct application of our self-bilinear map. However, the idea behind the construction is similar.

6.1 Definition of Homomorphic Encryption

Here, we recall some definitions for homomorphic encryption. A homomorphic encryption scheme HE consists of the four algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$. KeyGen takes the security parameter 1^λ

as input and outputs a public key pk and a secret key sk. Enc takes a public key pk and a massage $m \in \{0,1\}$ as input, and outputs a ciphertext c. Eval takes a public key pk, a circuit f with input length ℓ and a set of ℓ ciphertexts $c_1, \ldots, c_\ell$ as input, and outputs a ciphertext c_f. Dec takes a secret key sk and a ciphertext c as input, and outputs a message m. For correctness of the scheme, we require that for all $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and all $m \in \{0,1\}$, we have $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m$ with overwhelming probability.

Next, we define some properties of homomorphic encryption such as the CPA security, $\mathcal{C}$-homomorphism, and compactness.

Definition 5. *(CPA security) We say that a scheme* HE *is CPA secure if for any efficient adversary* $\mathcal{A}$,

$$|\Pr[1 \leftarrow \mathcal{A}(pk, \mathsf{Enc}(pk, 0))] - \Pr[1 \leftarrow \mathcal{A}(pk, \mathsf{Enc}(pk, 1))]|$$

is negligible, where $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$.

Definition 6. *($\mathcal{C}$-homomorphism) Let* $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ *be a class of circuits. A scheme* HE *is* $\mathcal{C}$-*homomorphic if for any family of circuits* $\{f_\lambda\}_{\lambda \in \mathbb{N}}$ *such that* $f_\lambda \in \mathcal{C}$ *whose input length is* ℓ *and any messages* $m_1, \ldots, m_\ell \in \{0,1\}$,

$$\Pr[\mathsf{Dec}(sk, \mathsf{Eval}(pk, C, c_1, \ldots, c_\ell)) \neq C(m_1, \ldots, m_\ell)]$$

is negligible, where $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and* $c_i \leftarrow \mathsf{Enc}(pk, m_i)$.

Remark 5. *We can also consider the additional property that an output of* Eval *can be used as input of another homomorphic evaluation. This is called "multi-hop" homomorphism, and many fully homomorphic encryption schemes have this property. However, our scheme does not.*

Definition 7. *(Compactness) A homomorphic encryption scheme* HE *is compact if there exists a polynomial* poly *such that the output length of* Eval *is at most* $\mathsf{poly}(\lambda)$*-bit.*

6.2 Φ-hiding Assumption

Here, we give the definition of the Φ-hiding assumption [35] as follows. Let $\mathsf{RSA}[p \equiv 1 \bmod e]$ be an efficient algorithm which takes the security parameter 1^λ as input and outputs (N, P, Q) where $N = PQ$ is an ℓ_N-bit Blum integer such that $P \equiv 1 \bmod e$ and $\mathbb{QR}_N^+$ is cyclic. Let $\mathcal{P}_\ell$ be the set of all ℓ-bit primes.

Definition 8. *For a constant c, we consider the following distributions.*

$$\mathcal{R} = \{(e, N) : e, e' \xleftarrow{R} \mathcal{P}_{c\ell_N}; N \leftarrow \mathsf{RSA}[p \equiv 1 \bmod e'](1^\lambda)\}$$
$$\mathcal{L} = \{(e, N) : e \xleftarrow{R} \mathcal{P}_{c\ell_N}; N \leftarrow \mathsf{RSA}[p \equiv 1 \bmod e](1^\lambda)\}$$

We say that the Φ-hiding assumption holds with respect to RSA *if for any efficient adversary* $\mathcal{A}$, $|\Pr[1 \leftarrow \mathcal{A}(\mathcal{L})] - \Pr[1 \leftarrow \mathcal{A}(\mathcal{R})]|$ *is negligible.*

Parameters. According to [35], N can be factorized in time $O(N^\epsilon)$ where $e \xleftarrow{R} \mathcal{P}_{c\ell_N}; N \leftarrow \mathsf{RSA}[p \equiv 1 \bmod e](1^k)$ and $c = 1/4 - \epsilon$. In our scheme, we set c to be the value such that $c\ell_N = \lambda$. This setting avoids the above mentioned attack in a usual parameter setting (e.g., $\ell_N = 1024$ for 80-bit security).

6.3 Our Construction

Here, we construct a somewhat homomorphic encryption scheme by using indistinguishability obfuscation. We use the notation for circuits on $\mathbb{QR}_N^+$ which is given in Sec. 4. In addition to that, here, we use the following notation. For circuits C_1 and C_2 such that an output of C_1 can be interpreted as input for C_2, $C_1 \circ C_2$ denotes the composition of C_1 and C_2, i.e, $C_1 \circ C_2$ is a circuit that computes $C_2(C_1(x))$ for input x. The construction of our homomorphic encryption $\mathsf{HE}_{\mathsf{Ours}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ is as follows.

$\mathsf{KeyGen}(1^\lambda)$**:** Choose $e \xleftarrow{\$} \mathcal{P}_\lambda$ and $(N, P, Q) \leftarrow \mathsf{RSA}[p \equiv 1 \bmod e](1^\lambda)$. Choose $g \xleftarrow{\$} \mathbb{QR}_N^+$ and compute an integer ρ such that $\rho \equiv 0 \bmod \mathrm{ord}(\mathbb{QR}_N^+)/e$ and $\rho \equiv 1 \bmod e$. It outputs a public key $pk = (N, e, g)$ and a secret key $sk = (\rho, pk)$.

$\mathsf{Enc}(pk, m \in \{0,1\})$**:** Choose $r \xleftarrow{\$} [(N-1)/4]$, set $c \leftarrow i\mathcal{O}(\mathsf{Max}, \tilde{C}_{N,m+re})$ and output c, where Max is defined as an integer larger than $\max_{m\in\{0,1\}, r\in[(N-1)/4]}\{|\tilde{C}_{N,m+re}|\}$.

$\mathsf{Eval}(pk, f, c_1, \ldots, c_\ell)$**:** Work only if $c_1, \ldots, c_\ell$ are circuits (i.e., generated by Enc). Convert f into an arithmetic circuit f' on $\mathbb{Z}_e$. (That is, each gate of f' is addition, multiplication or negation on $\mathbb{Z}_e$.)[5] Compute as follows for all wires of f' from wires with lower depth.

- *Input*: Let w be the i-th input wire. Then c_i is assigned to this wire.
- *Addition*: Let w be an output wire of an addition gate. Set $c_w := \mathsf{Mult}(c_{A(w)}, c_{B(w)})$.
- *Multiplication*: Let w be an output wire of a multiplication gate. Set $c_w := c_{A(w)} \circ c_{B(w)}$.
- *Negation*: Let w be an output wire of a negation gate. Set $c_w := C_{N,inv} \circ c_{A(w)}$ where $C_{N,inv}$ is a circuit that computes an inverse on $\mathbb{QR}_N^+$.

Let v be the output wire. Compute $c_{\mathsf{eval}} = c_v(g)$ and output it. Note that it is a group element and not a circuit. Therefore we cannot evaluate it again.

$\mathsf{Dec}(sk, c)$**:** Work differently depending on whether c is an output of Enc or Eval. If c is an output of Enc, then compute $M = c(g)$. If $M^\rho = 1$, then output 0, and otherwise output 1. If c is an output of Eval, then output 0 if $c^\rho = 1$, and otherwise output 1.

First, we prove the correctness of the scheme. We have $e | \mathrm{ord}(\mathbb{QR}_N^+)$ by the choice of N. Therefore, there exists a subgroup G_e^+ of order e of $\mathbb{QR}_N^+$. We can see that for any element $h \in \mathbb{QR}_N^+$, h^ρ is the G_e^+ component of h. In the decryption, we have $M = i\mathcal{O}(\mathsf{Max}, C_{N,m+re})(g) = g^{m+re}$. Therefore M^ρ is the G_e^+ component of g^m. We can see that G_e^+ component of g is not 1 with overwhelming probability since e is a λ-bit prime. Therefore $M^\rho = 1$ is equivalent to $m = 0$ and $M^\rho \neq 1$ is equivalent to $m = 1$ with overwhelming probability. Thus the correctness follows.

The security of $\mathsf{HE}_{\mathsf{Ours}}$ relies on the Φ-hiding assumption. Specifically, it satisfies the following property.

Theorem 5. $\mathsf{HE}_{\mathsf{Ours}}$ *is NC*^{1}*-homomorphic, compact and CPA secure if the* Φ*-hiding assumption holds with respect to* RSA *and* $i\mathcal{O}$ *is an indistinguishability obfuscator for P/poly.*

Proof. **NC^1-homomorphism.** We show that $\mathsf{HE}_{\mathsf{Ours}}$ is NC^1-homomorphic. Here, NC^1 is the class of circuits with depth $O(\log(\lambda))$. Note that if f is in NC^1, then the depth of the corresponding arithmetic circuit f' is also $O(\log(\lambda))$. First, we show the correctness of Eval. We can see that

[5] This can be done since we have $a \wedge b = a \cdot b \bmod e$ and $a \vee b = a + b - a \cdot b \bmod e$ if $a, b \in \{0,1\}$.

output c_{eval} of Eval satisfies $c_{\mathsf{eval}} = g^m$ by an easy induction where m is a corresponding message. Therefore we can prove that c_{eval} is correctly decrypted similarly as the above. Next, we show that Eval is computed efficiently if the depth of f' is $O(\log(\lambda))$. We can easily see that Eval is computed efficiently if $|c_v|$ is polynomially bounded in the security parameter. Let M_j be the maximum value of $|c_w|$ for a wire w with depth j. M_1 is constant in $|f'|$. For $j \geq 2$, we have $M_j \leq 2M_{j-1} + \max\{|C_{N,mult}|, |C_{N,inv}|\}$. Therefore if the depth of f' is $O(\log \lambda)$, then $|c_v|$ is polynomially bounded by $|f'|$ (and therefore λ). Therefore Eval is efficiently computable.

Compactness. $\mathsf{HE}_{\mathsf{Ours}}$ is compact since output of Eval is always an element of $\mathbb{QR}_N^+$.

Security. To prove the security, we consider the following sequence of games.

Game **1:** This game is the original CPA game. More formally, it is as follows.

$e \xleftarrow{\$} \mathcal{P}_\lambda$
$(N, P, Q) \leftarrow \mathsf{RSA}[p \equiv 1 \bmod e](1^\lambda)$
$g \xleftarrow{\$} \mathbb{Z}_N^*$
$b \xleftarrow{\$} \{0, 1\}$
$r \xleftarrow{\$} [(N-1)/4]$
$c \leftarrow i\mathcal{O}(\mathsf{Max}, \tilde{C}_{N,b+re})$
$b' \xleftarrow{\$} \mathcal{A}(N, e, g, c)$

Game **2:** This is the same game as Game 1 except that N and e are set differently as follows.

$e, e' \xleftarrow{\$} \mathcal{P}_\lambda$
$(N, P, Q) \leftarrow \mathsf{RSA}[p \equiv 1 \bmod e'](1^\lambda)$

Game **3:** This is the same game as Game 2 except that c is set differently as follows.

$c \leftarrow i\mathcal{O}(\mathsf{Max}, \tilde{C}_{N,(b+re \bmod \mathrm{ord}(\mathbb{QR}_N^+))})$

Game **4:** This is the same game as Game 3 except that c is set differently as follows.

$r' \xleftarrow{\$} [\mathrm{ord}(\mathbb{QR}_N^+)]$
$c \leftarrow i\mathcal{O}(\mathsf{Max}, C_{N,r'})$

Let T_i be the event that $b' = b$ in Game i. What we want to prove is $|\Pr[T_1] - 1/2|$ is negligible. We prove it by the following lemmas.

Lemma 4. $\Pr[T_1] - \Pr[T_2]$ *is negligible if the* Φ*-hiding assumption holds.*

Proof. It is easy to see that an adversary that distinguishes Game 1 and Game 2 is reduced to an adversary that breaks the Φ-hiding assumption. □

Lemma 5. $\Pr[T_2] - \Pr[T_3]$ *is negligible if* $i\mathcal{O}$ *is an indistinguishability obfuscator for P/poly.*

Proof. $\tilde{C}_{N,b+re}$ and $\tilde{C}_{N,(b+re \bmod \mathrm{ord}(\mathbb{QR}_N^+))}$ compute identically for all input. Therefore the lemma follows from the property of $i\mathcal{O}$. □

Lemma 6. $\Pr[T_3] - \Pr[T_4]$ *is negligible.*

Proof. In Game 3, $\text{ord}(\mathbb{QR}_N^+)$ is coprime to e with overwhelming probability. Therefore the distribution of $r \bmod \text{ord}(\mathbb{QR}_N^+)$ where $r \overset{\$}{\leftarrow} [(N-1)/4]$ is negligibly close to the uniform distribution on $\mathbb{Z}_{\text{ord}(\mathbb{QR}_N^+)}$ since $(N-1)/4$ is negligibly close to $\text{ord}(\mathbb{QR}_N^+)$. □

Lemma 7. $\Pr[T_4] = 1/2$

Proof. In Game 4, $\mathcal{A}$ obtain no information of b, therefore the probability that $\mathcal{A}$ predicts b is 1/2. □

□

Acknowledgment

We would like to thank the anonymous reviewers of CRYPTO 2014 and members of the study group "Shin-Akarui-Angou-Benkyou-Kai" for their helpful comments. Especially, we would like to thank Satsuya Ohata for his instructive comment on self-bilinear maps, and Takahiro Matsuda and Jacob Schuldt for their detailed proofreading. We also thank Kenny Paterson for his valuable comments.

References

[1] Prabhanjan Vijendra Ananth, Divya Gupta, Yuval Ishai, and Amit Sahai. Optimizing obfuscation: Avoiding barrington's theorem. In *ACM Conference on Computer and Communications Security*, pages 646–658, 2014.

[2] Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Protecting obfuscation against algebraic attacks. In *EUROCRYPT*, pages 221–238, 2014.

[3] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *CRYPTO*, pages 1–18, 2001.

[4] Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In *ACM Conference on Computer and Communications Security*, pages 784–796, 2012.

[5] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334, 2007.

[6] Lenore Blum, Manuel Blum, and Mike Shub. A simple unpredictable pseudo-random number generator. *SIAM J. Comput.*, 15(2):364–383, 1986.

[7] Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity-based encryption without random oracles. In *EUROCRYPT*, pages 223–238, 2004.

[8] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, pages 213–229, 2001.

[9] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *Contemporary Mathematics*, 324:71–90, 2002.

[10] Dan Boneh, David J. Wu, and Joe Zimmerman. Immunizing multilinear maps against zeroizing attacks. *IACR Cryptology ePrint Archive*, 2014:930, 2014.

[11] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In *CRYPTO*, 2014.

[12] Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In *TCC*, pages 1–25, 2014.

[13] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. *IACR Cryptology ePrint Archive*, 2014:906, 2014.

[14] Jung Hee Cheon and Dong Hoon Lee. A note on self-bilinear maps. *Bulletin of the Korean Mathematical Society*, 46(2):303–309, 2009.

[15] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In *CRYPTO (1)*, pages 476–493, 2013.

[16] Jean-Sebastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Cryptanalysis of two candidate fixes of multilinear maps over the integers. Cryptology ePrint Archive, Report 2014/975, 2014. `http://eprint.iacr.org/`.

[17] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[18] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT*, pages 1–17, 2013.

[19] Sanjam Garg, Craig Gentry, Shai Halevi, and Mariana Raykova. Two-round secure MPC from indistinguishability obfuscation. In *TCC*, pages 74–94, 2014.

[20] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, pages 40–49, 2013.

[21] Sanjam Garg, Craig Gentry, Shai Halevi, Amit Sahai, and Brent Waters. Attribute-based encryption for circuits from multilinear maps. In *CRYPTO (2)*, pages 479–499, 2013.

[22] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *STOC*, pages 467–476, 2013.

[23] Craig Gentry, Shai Halevi, Hemanta K. Maji, and Amit Sahai. Zeroizing without zeroes: Cryptanalyzing multilinear maps without encodings of zero. *IACR Cryptology ePrint Archive*, 2014:929, 2014.

[24] Craig Gentry, Allison Lewko, Amit Sahai, and Brent Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. Cryptology ePrint Archive, Report 2014/309, 2014. `http://eprint.iacr.org/`.

[25] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32, 1989.

[26] Shafi Goldwasser, S. Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. In *Eurocrypt*, 2014.

[27] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In *STOC*, pages 545–554, 2013.

[28] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM Conference on Computer and Communications Security*, pages 89–98, 2006.

[29] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero knowledge for NP. In *EUROCRYPT*, pages 339–358, 2006.

[30] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT*, pages 415–432, 2008.

[31] Dennis Hofheinz. Fully secure constrained pseudorandom functions using random oracles. Cryptology ePrint Archive, Report 2014/372, 2014. `http://eprint.iacr.org/`.

[32] Dennis Hofheinz and Eike Kiltz. The group of signed quadratic residues and applications. In *CRYPTO*, pages 637–653, 2009.

[33] Susan Hohenberger, Amit Sahai, and Brent Waters. Replacing a random oracle: Full domain hash from indistinguishability obfuscation. *Eurocrypt*, 2014.

[34] Antoine Joux. A one round protocol for tripartite Diffie-Hellman. In *ANTS*, pages 385–394, 2000.

[35] Eike Kiltz, Adam O'Neill, and Adam Smith. Instantiability ofRSA-OAEP under chosen-plaintext attack. In *CRYPTO*, pages 295–313, 2010.

[36] Qixiang Mei, Bao Li, Xianhui Lu, and Dingding Jia. Chosen ciphertext secure encryption under factoring assumption revisited. In *Public Key Cryptography*, pages 210–227, 2011.

[37] Alfred Menezes, Tatsuaki Okamoto, and Scott A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions on Information Theory*, 39(5):1639–1646, 1993.

[38] Omkant Pandey, Manoj Prabhakaran, and Amit Sahai. Obfuscation-based non-black-box simulation and four message concurrent zero knowledge for np. In *TCC*, 2015.

[39] Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In *CRYPTO*, pages 500–517, 2014.

[40] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473, 2005.

[41] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: Deniable encryption, and more. In *STOC*, 2014.

[42] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing (in japanese). In *SCIS*, 2000.

[43] Yannick Seurin. New constructions and applications of trapdoor DDH groups. In *Public Key Cryptography*, pages 443–460, 2013.

[44] Brent Waters. Efficient identity-based encryption without random oracles. In *EUROCRYPT*, pages 114–127, 2005.

A Multi-bit Variant

In the construction of a self-bilinear map with auxiliary information which is given in Sec. 4, we can obtain only 1-bit hardcore function for the MHDH assumption. Here, we modify the construction so that we can obtain k-bit hardcore function for any integer k (which is polynomially bounded in λ). The idea of our multi-bit variant is similar to the Blum-Blum-Shub pseudorandom number generator [6].

A.1 Construction

The construction of our multi-bit variant $\mathcal{SBP}_{\mathsf{Mult}}$ is as follows.

$\mathsf{InstGen}(1^\lambda) \to \mathsf{params} = (N, g)$: It runs $\mathsf{RSAGen}(1^\lambda)$ to obtain (N, P, Q), chooses $g \overset{\$}{\leftarrow} G$ and outputs $\mathsf{params} := (N, g)$. params defines the underlying group $G := \mathbb{QR}_N^+$ the self-bilinear map as $e(g^x, g^y) = g^{2^k xy}$ and $\mathrm{Approx}(G) = (N-1)/4$. For an integer x and $\ell \in \mathbb{N}$, a set T_x^ℓ is defined as $T_x^\ell = \{i\mathcal{O}(M_\ell, C_{N,2^k x}; r) : C_{N,2^k x} \in \mathcal{C}_{N,2^k x} \text{ such that } |C_{N,2^k x}| \leq M_\ell, r \in \{0,1\}^*\}$, where M_ℓ is defined later.

$\mathsf{AIGen}(\mathsf{params}, \ell, x) \to \tau_x$: It takes the canonical circuit $\tilde{C}_{N,2^k x} \in \mathcal{C}_{N,2^k x}$, sets $\tau_x \leftarrow i\mathcal{O}(M_\ell, \tilde{C}_{N,2^k x})$ and outputs τ_x.

$\mathsf{Map}(\mathsf{params}, g^x, \tau_y) \to e(g^x, g^y)$: It computes $\tau_y(g^x)$ and outputs it. (Recall that τ_y is a circuit that computes the $2^k y$-th power for an element of $\mathbb{QR}_N^+$.)

$\mathsf{AIMult}(\mathsf{params}, \ell, \tau_x, \tau_y) \to \tau_{x+y}$: It computes $\tau_{x+y} \leftarrow i\mathcal{O}(M_\ell, \mathsf{Mult}(\tau_x, \tau_y))$ and outputs it.

Definition of M_ℓ. M_ℓ represents an upper bound of the size of a circuit which is obfuscated by $i\mathcal{O}$ when auxiliary information with level ℓ is generated (by AIGen or AIMult). It can be defined recursively as follows. We let $M_1 := \max_{x \in [(N-1)/4]}\{|\tilde{C}_{N,2^k x}|\}$ and $M_{\ell+1} := 2\mathsf{poly}(M_\ell, \lambda) + |C_{\mathsf{Mult}}|$ for $\ell \geq 1$ where poly is a polynomial that satisfies $|i\mathcal{O}(M, C)| < \mathsf{poly}(M, \lambda)$ for any integer M and circuit C such that $|C| < M$.

Indistinguishability of auxiliary information. If we have $z \equiv x + y \bmod \mathrm{ord}(\mathbb{QR}_N^+)$, then $C_{N,2^k z}$ and $\mathsf{Mult}(\tau_x, \tau_y)$ have exactly the same functionality. Therefore if we obfuscate these circuits by $i\mathcal{O}$, then the resulting circuits are computationally indistinguishable.

Indistinguishability of auxiliary information easily follows from the security of the indistinguishability obfuscator since $\tau_x \in \cup_{\ell \in \mathbb{N}} T_x^\ell$ is an obfuscation of a circuit that computes the $2x$-th power for an element of $\mathbb{QR}_N^+$ regardless of whether τ_x is generated by AIGen or AIMult.

We also define the BBS generator which we will use as a hardcore function.

Definition 9. *For ℓ_N-bit Blum integer N, $g \in \mathbb{QR}_N^+$ and $r \in \{0,1\}^{\ell_N}$, we define the BBS generator as*

$$BBS_r(g) := (\mathsf{GL}_r(g), \ldots, \mathsf{GL}_r(g^{k-1}))$$

where GL denote the Goldreich-Levin hardcore bit function [25]. That is, $\mathsf{GL}_r(x) := \bigoplus_{i=1}^{\ell_N} r_i x_i$ where r_i and x_i are i-th bit of r and x which is represented as an integer in $\{1, \ldots, (N-1)/2\}$. We write $\mathcal{BBS}$ to denote the family of functions $\{BBS_r\}_{r \in \{0,1\}^{\ell_N}}$.

A.2 Hardness Assumption

The following hardness assumption holds with respect to our construction.

Theorem 6. *The MHDH assumption holds with respect to $\mathcal{SBP}_{\mathsf{Mult}}$ and $\mathcal{BBS}$ if the factoring assumption holds for* RSAGen *and $i\mathcal{O}$ is an indistinguishability obfuscator for P/poly.*

Proof. For an algorithm $\mathcal{A}$, we consider the following games.

Game 1**.** This game is the original n-MHDH game. More precisely, it is as follows.

$$
\begin{aligned}
&(N, P, Q) \leftarrow \mathsf{RSAGen}(1^\lambda)\\
&g \xleftarrow{\$} \mathbb{QR}_N^+\\
&r \xleftarrow{\$} \{0,1\}^{\ell_N}\\
&x_0, \ldots, x_n \xleftarrow{\$} [(N-1)/4]\\
&\tau_{x_i} \leftarrow i\mathcal{O}(M_{\ell_i}, \tilde{C}_{N, 2^k x_i}) \text{ for } i = 0, \ldots, n\\
&T := BBS_r(g^{2^{k(n-1)}\Pi_{i=0}^n x_i})\\
&b \leftarrow \mathcal{A}(N, g, g^{x_0}, \ldots, g^{x_n}, \tau_{x_0} \ldots, \tau_{x_n}, r, T)
\end{aligned}
$$

Game $1'$ This game is the same as Game 1 except that $x_0, \ldots, x_n$ are chosen from $[\mathrm{ord}(\mathbb{QR}_N^+)]$.

Game $2'$**.** This game is the same as Game 1 except that g, $x_0, \ldots, x_n$, $\tau_{x_0}, \ldots, \tau_{x_n}$ are set differently. More precisely, it is as follows.

$$
\begin{aligned}
&(N, P, Q) \leftarrow \mathsf{RSAGen}(1^\lambda)\\
&h \xleftarrow{\$} \mathbb{QR}_N^+\\
&g := h^{2^k}\\
&x'_0, \ldots, x'_n \xleftarrow{\$} [\mathrm{ord}(\mathbb{QR}_N^+)]\\
&g^{x_i} := g^{x'_i} h \text{ for } i = 0, \ldots, n\\
&\text{(This implicitly defines } x_i \equiv x'_i + 1/2^k \bmod \mathrm{ord}(\mathbb{QR}_N^+))\\
&\tau_{x_i} \leftarrow i\mathcal{O}(M_{\ell_i}, \tilde{C}_{N, 2^k x'_i + 1}) \text{ for } i = 0, \ldots, n\\
&T := BBS_r(g^{2^{k(n-1)}\Pi_{i=0}^n x_i})\\
&b \leftarrow \mathcal{A}(N, g, g^{x_0}, \ldots, g^{x_n}, \tau_{x_0} \ldots, \tau_{x_n}, r, T)
\end{aligned}
$$

Game 2 This game is the same as Game $2'$ except that $x'_0, \ldots, x'_n$ are chosen from $[(N-1)/4]$.

Game 3**.** This game is the same as Game 2 except that T is set as a random k-bit string.

Let T_i be the event that $\mathcal{A}$ outputs 1 in Game i and T'_i be the event that $\mathcal{A}$ outputs 1 in Game i'. What we want to prove is $|\Pr[T_1] - \Pr[T_3]|$ is negligible. We prove this by the following lemmas.

Lemma 8. $|\Pr[T_i] - \Pr[T'_i]|$ *is negligible for $i = 1, 2$*

Proof. This follows since $(N-1)/4$ is negligibly close to $\mathrm{ord}(\mathbb{QR}_N^+)$. □

Lemma 9. $|\Pr[T'_1] - \Pr[T'_2]|$ *is negligible if $i\mathcal{O}$ is an indistinguishability obfuscator for P/poly.*

Proof. We define hybrid games $H_{1,0},\ldots H_{1,n+1}$. A hybrid game $H_{1,i}$ is the same as $\mathsf{Game}\ 1'$ except that the first i auxiliary information (i.e, $\tau_{x_0},\tau_{x_1},\ldots,\tau_{x_{i-1}}$) are generated as in $\mathsf{Game}\ 2'$. Let $T_{1,i}$ be the event that $\mathcal{A}$ outputs 1 in the hybrid $H_{1,i}$. It is clear that $H_{1,0}$ is $\mathsf{Game}\ 1'$ and $H_{1,n+1}$ is $\mathsf{Game}\ 2'$. Let $T_{1,i}$ be the event that $\mathcal{A}$ wins in $\mathsf{Game}\ H_{1,i}$. Since we have $x_i \equiv x_i' + 1/2^k \bmod \mathrm{ord}(\mathbb{QR}_N^+)$, $C_{N,2^k x_i'+1}$ computes exactly the same as $C_{N,2^k x_i}$ for any input for $i = 0,\ldots n$. (Recall that these circuits computes the exponentiation only for an element of $\mathbb{QR}_N^+$.) Then we can see that $|\Pr[T_{1,i}] - \Pr[T_{1,i-1}]$ is negligible for $i \in [n+1]$ from the security of $i\mathcal{O}$. (Note that a reduction algorithm here may know the factorization of N.) □

Lemma 10. $|\Pr[T_2] - \Pr[T_3]|$ *is negligible if the factoring assumption holds for* RSAGen *and* $i\mathcal{O}$ *is an indistinguishability obfuscator for P/poly.*

Proof. We define hybrid games $H_{2,0},\ldots H_{2,k}$. For $i = 0,1,\ldots,k$, a hybrid game $H_{2,i}$ is the same as $\mathsf{Game}\ 2$ except that the first i-bit of T are set as in $\mathsf{Game}\ 2$ and other bits are set as in $\mathsf{Game}\ 3$, i.e, $T := U_1||\ldots||U_i||\mathsf{GL}_r(g^{2^{k(n-1)+i}\Pi_{j=0}^n x_j})||\ldots||\mathsf{GL}_r(g^{2^{kn-1}\Pi_{j=0}^n x_j})$, where $U_1\ldots U_i \xleftarrow{\$} \{0,1\}$. In the following, we write $\mathsf{GL}(r,i)$ to denote $\mathsf{GL}_r(g^{2^{k(n-1)+i}\Pi_{j=0}^n x_j})$ for notational simplicity. It is clear that $H_{2,0}$ is the same as $\mathsf{Game}\ 2$ and $H_{2,k}$ is the same as $\mathsf{Game}\ 3$. Let $T_{2,i}$ be the event that $\mathcal{A}$ outputs 1 in the hybrid $H_{2,i}$. We prove that $|\Pr[T_{2,i-1}] - \Pr[T_{2,i}]|$ is negligible for all $i \in [k]$. To do so, we assume that there exists an algorithm $\mathcal{A}$ that distinguishes $H_{2,i}$ and $H_{2,i-1}$, and construct a factoring algorithm by using $\mathcal{A}$. Without loss of generality, we can assume that there exists a negligible function ϵ such that $\Pr[T_{2,i-1}] - \Pr[T_{2,i}] > \epsilon$. This is because given $\mathcal{A}$, the sign of $\Pr[T_{2,i-1}] - \Pr[T_{2,i}]$ can be checked efficiently, and if $\Pr[T_{2,i-1}] - \Pr[T_{2,i}] < 0$ then we can modify $\mathcal{A}$ to output inverse of the original output so that $\Pr[T_{2,i-1}] - \Pr[T_{2,i}] > 0$. In the following, we use a similar argument as in [36].

Hardcore Predictor $\mathcal{P}$. First, we construct an algorithm $\mathcal{P}$ that predicts $\mathsf{GL}(r,i-1)$ with non-negligible advantage when it is given $(r,N,g,g^{x_0},\ldots,g^{x_n},\tau_{x_0}\ldots,\tau_{x_n},g^{2^{k(n-1)+i}\Pi_{j=0}^n x_j})$ where r, N, g, $x_0,\ldots,x_n$ and $\tau_{x_0}\ldots,\tau_{x_n}$ are defined as in $\mathsf{Game}\ 2$. The construction of $\mathcal{P}$ is as follows.

$\mathcal{P}(N,g,g^{x_0},\ldots,g^{x_n},\tau_{x_0}\ldots,\tau_{x_n},g^{2^{k(n-1)+i}\Pi_{j=0}^n x_j},r)$**:** $\mathcal{D}'$ picks $b \xleftarrow{\$} \{0,1\}$, sets $T := U_1||\ldots||U_{i-1}||b$ $||\mathsf{GL}(r,i)||\ldots||\mathsf{GL}(r,k-1)$ and runs $\mathcal{A}(N,g,g^{x_0},\ldots,g^{x_n},\tau_{x_0}\ldots,\tau_{x_n},r,T)$. Note that $\mathcal{D}$' can generate $\mathsf{GL}(r,i),\ldots,\mathsf{GL}(r,k-1)$ since it knows $g^{2^{k(n-1)+i}\Pi_{j=0}^n x_j})$. If $\mathcal{A}$ outputs 1, then $\mathcal{P}$ outputs b, and otherwise it picks an independently random bit $b' \xleftarrow{\$} \{0,1\}$ and outputs it.

For notational simplicity, we define $Y' := (N,g,g^{x_0},\ldots,g^{x_n},\tau_{x_0}\ldots,\tau_{x_n},g^{2^{k(n-1)+i}\Pi_{i=0}^n x_i})$, i.e., Y' denotes input of $\mathcal{P}$ except r and define $Y := (N,g,g^{x_0},\ldots,g^{x_n},\tau,\tau_{x_0}\ldots,\tau_{x_n})$, i.e., Y denotes input of $\mathcal{P}$ except r and T. We prove that with the probability at least $\epsilon/2$ over the choice of Y', $\mathcal{P}$ predicts $\mathsf{GL}(r,i-1)$ with advantage $\epsilon/4$. By the standard averaging argument, with at least $\epsilon/2$ fraction of the choice of Y, we have

$$\begin{aligned}&\Pr[1 \leftarrow \mathcal{A}(Y,r,U_1||\ldots||U_{i-1}||\mathsf{GL}(r,i-1)||\ldots||\mathsf{GL}(r,k-1))]\\ -&\Pr[1 \leftarrow \mathcal{A}(Y,r,U_1||\ldots||U_i||\mathsf{GL}(r,i)||\ldots||\mathsf{GL}(r,k-1))] > \epsilon/2.\end{aligned}$$

over the choice of r and randomness of $\mathcal{A}$. Conditioned on such Y is fixed, we have

$$\begin{aligned}
&\Pr[\mathsf{GL}(r,i-1) \leftarrow \mathcal{P}(Y',r)] \\
&= \Pr[\mathsf{GL}(r,i-1) \leftarrow \mathcal{P}(Y',r) | b = \mathsf{GL}(r,i-1)] \Pr[b = \mathsf{GL}(r,i-1)] \\
&\quad + \Pr[\mathsf{GL}(r,i-1) \leftarrow \mathcal{P}(Y',r) | b \neq \mathsf{GL}(r,i-1)] \Pr[b \neq \mathsf{GL}(r,i-1)] \\
&= (1 \cdot \Pr[1 \leftarrow \mathcal{A}(Y, r, U_1 || \dots || U_{i-1} || \mathsf{GL}(r,i-1) || \dots || \mathsf{GL}(r,k-1)] \\
&\quad + 1/2 \cdot (1 - \Pr[1 \leftarrow \mathcal{A}(Y, r, U_1 || \dots || U_{i-1} || \mathsf{GL}(r,i-1) || \dots || \mathsf{GL}(r,k-1)])) \cdot 1/2 \\
&+ (0 \cdot \Pr[1 \leftarrow \mathcal{A}(Y, r, U_1 || \dots || U_{i-1} || 1 - \mathsf{GL}(r,i-1) || \mathsf{GL}(r,i) || \dots || \mathsf{GL}(r,k-1))] \\
&\quad + 1/2 \cdot (1 - \Pr[1 \leftarrow \mathcal{A}(Y, r, U_1 || \dots || U_{i-1} || 1 - \mathsf{GL}(r,i-1) || \mathsf{GL}(r,i) || \dots || \mathsf{GL}(r,k-1))])) \cdot 1/2 \\
&= 1/2 + 1/2 \cdot (\Pr[1 \leftarrow \mathcal{A}(Y, r, U_1 || \dots || U_{i-1} || \mathsf{GL}(r,i-1) || \dots || \mathsf{GL}(r,k-1)] \\
&\quad - \Pr[1 \leftarrow \mathcal{A}(Y, r, U_1 || \dots || U_{i-1} || 1 - \mathsf{GL}(r,i-1) || \mathsf{GL}(r,i) || \dots || \mathsf{GL}(r,k-1))]) \\
&> 1/2 + \epsilon/4
\end{aligned}$$

Reconstruction Algorithm. We obtained an algorithm $\mathcal{P}$ that distinguishes $\mathsf{GL}(r,i-1) = \mathsf{GL}_r(g^{2^{k(n-1)+i-1}\Pi_{j=0}^n x_j})$ from a random bit with the advantage larger than ϵ when it is given Y', r for at least $\epsilon/2$ fraction of Y'. Here, we use the Goldreich-Levin theorem.

Theorem 7. *(Goldreich-Levin Theorem [25]) Let x be an n-bit string. If there exists a PPT algorithm $\mathcal{P}$ such that*

$$|\Pr[\mathsf{GL}_r(x) \leftarrow \mathcal{P}(r)] - 1/2|$$

is non-negligible where $r \xleftarrow{\$} \{0,1\}^n$, then there exists a PPT algorithm $\mathcal{R}$ such that

$$\Pr[x \leftarrow \mathcal{R}(1^n)]$$

is non-negligible.

By using this theorem, we obtain an algorithm $\mathcal{R}$ that computes $g^{2^{k(n-1)+i-1}\Pi_{j=0}^n x_j}$ with non-negligible probability when it is given $(N, g, g^{x_0}, \dots, g^{x_n}, \tau_{x_0} \dots, \tau_{x_n}, g^{2^{k(n-1)+i}\Pi_{j=0}^n x_j})$ for non-negligible fraction of its input.

Factoring Algorithm Then we construct an algorithm $\mathcal{B}$ that given an RSA modulus N factorizes NThe construction of $\mathcal{B}$ is as follows.

$\mathcal{B}(N)$**:** $\mathcal{B}$ chooses $h' \xleftarrow{\$} \mathbb{Z}_N^* \setminus \mathbb{QR}_N^+$ sets $h := |h'^2 \mod N| \in \mathbb{QR}_N^+$, $g := h^{2^k}$, chooses $x'_0, \dots, x'_n \xleftarrow{\$} [(N-1)/4]$, sets $g^{x_0} := g^{x'_0} h^{2^{k-i}}$, $\tau_{x_0} \leftarrow i\mathcal{O}(M'_1, C_{N, 2^k x'_0 + 2^{k-i}})$, $g^{x_i} := g^{x'_i} h$, $\tau_{x_i} \leftarrow i\mathcal{O}(M'_1, C_{N, 2^k x'_i + 1})$ for $i \in [n]$. Then $\mathcal{B}$ can compute $g^{2^{k(n-1)+i}\Pi_{j=0}^n x_j} = h^{2^{kn+i}\Pi_{j=0}^n x_j} = h^{(2^i x'_0 + 1)\Pi_{j=1}^n (2^k x'_j + 1)}$. $\mathcal{B}$ runs $\mathcal{R}(N, g, g^{x_0}, \dots, g^{x_n}, \tau_{x_0}, \dots, \tau_{x_n}, g^{2^{k(n-1)+i}\Pi_{j=0}^n x_j})))$. Let U be the output of $\mathcal{R}$. Then $\mathcal{B}$ computes $X := \Pi_{j=1}^n (2^k x'_j + 1)$ and computes $V = U h^{-(2^{i-1} x'_0 X + (X-1)/2)}$. (Note that X is odd and therefore $(X-1)/2$ is an integer.) Then it outputs $\gcd(h', V)$.

First, we consider the distribution of input for $\mathcal{R}$. Clearly, all components except g^{x_0} and τ_{x_0} are distributed as in Game 2. In the above algorithm, g^{x_0} is distributed almost uniformly on $\mathbb{QR}_N^+$ as in Game 2 and therefore this difference causes a negligible difference on the behavior of $\mathcal{R}$. τ_{x_0} is set as an obfuscation of a circuit that computes $2^k x_0$-th power both in the above algorithm and in Game 2, and this causes a negligible difference by the property of indistinguishability obfuscation.

Therefore $\mathcal{R}$ outputs $g^{2^{k(n-1)+i-1}\Pi_{j=0}^{n} x_j}$ with non-negligible probability for non-negligible fraction of its input. In this case, we have

$$\begin{aligned} U &= g^{2^{k(n-1)+i-1}\Pi_{j=0}^{n} x_j} = h^{2^{kn+i-1}(x_0'+1/2^i)\Pi_{j=1}^{n}(x_j'+1/2^k)} \\ &= h^{(2^{i-1}x_0'+1/2)\Pi_{j=1}^{n}(2^k x_j'+1)} = h^{2^{i-1}x_0' X+(X-1)/2+1/2}. \end{aligned}$$

Therefore we have $V = h^{1/2}$. Thus h' and V are distinct square roots of h in $\mathbb{Z}_N^*$ and therefore $\gcd(h', V)$ is a non-trivial factor of N. □

Theorem 6 is proven by the above lemmas. □