J Cryptol (2020) 33:1080–1113
https://doi.org/10.1007/s00145-019-09340-0

Journal of
CRYPTOLOGY

Multilinear Maps from Obfuscation*

Martin R. Albrecht
Royal Holloway, University of London, London, United Kingdom

Pooya Farshim
DI/ENS and CNRS, PSL University, Paris, France

Shuai Han
Shanghai Jiao Tong University, Shanghai, China

Dennis Hofheinz
Karlsruhe Institute of Technology, Karlsruhe, Germany
dennis.hofheinz@kit.edu

Enrique Larraia · Kenneth G. Paterson
Royal Holloway, University of London, London, United Kingdom

Communicated by Alon Rosen.

Received 1 February 2018
Online publication 2 January 2020

Abstract. We provide constructions of multilinear groups equipped with natural hard problems from indistinguishability obfuscation, homomorphic encryption, and NIZKs. This complements known results on the constructions of indistinguishability obfuscators from multilinear maps in the reverse direction. We provide two distinct, but closely related constructions and show that multilinear analogues of the DDH assumption hold for them. Our first construction is *symmetric* and comes with a κ-linear map $\mathbf{e} : \mathbb{G}^{\kappa} \longrightarrow \mathbb{G}_T$ for prime-order groups $\mathbb{G}$ and $\mathbb{G}_T$. To establish the hardness of the κ-linear DDH problem, we rely on the existence of a base group for which the κ-strong DDH assumption holds. Our second construction is for the *asymmetric* setting, where $\mathbf{e} : \mathbb{G}_1 \times \cdots \times \mathbb{G}_{\kappa} \longrightarrow \mathbb{G}_T$ for a collection of $\kappa + 1$ prime-order groups $\mathbb{G}_i$ and $\mathbb{G}_T$, and relies only on the 1-strong DDH assumption in its base group. In both constructions, the linearity κ can be set to any arbitrary but a priori fixed polynomial value in the security parameter. We rely on a number of powerful tools in our constructions: probabilistic indistinguishability obfuscation, dual-mode NIZK proof systems (with perfect soundness, witness-indistinguishability, and zero knowledge), and additively homomorphic encryption for the group $\mathbb{Z}_N^+$. At a high level, we enable "bootstrapping" multilinear assumptions from their simpler counterparts in standard cryptographic groups and show the equivalence of PIO and multilinear maps under the existence of the aforementioned primitives.

*This paper has been handled by Ivan Bjerre Damgård as acting editor in chief and communicated by Alon Rosen.

Keywords. Multilinear map, Indistinguishability obfuscation, Homomorphic encryption, Decisional Diffie–Hellman, Groth–Sahai proofs.

1. Introduction

1.1. *Main Contribution*

In this paper, we explore the relationship between multilinear maps and obfuscation. Our main contribution is a construction of multilinear maps for groups of prime order equipped with natural hard problems, using indistinguishability obfuscation (IO) in combination with other tools, namely NIZK proofs, homomorphic encryption, and a base group $\mathbb{G}_0$ satisfying a mild cryptographic assumption. This complements known results in the reverse direction, showing that various forms of indistinguishability obfuscation can be constructed from multilinear maps [17,24,45]. The relationship between IO and multilinear maps is a very natural question to study, given the rich diversity of cryptographic constructions that have been obtained from both multilinear maps and obfuscation, and the apparent fragility of current constructions for multilinear maps. More on this below.

We provide two distinct but closely related constructions. One is for multilinear maps in the *symmetric* setting, that is, non-degenerate multilinear maps $\mathbf{e} : \mathbb{G}_1{}^{\kappa} \longrightarrow \mathbb{G}_T$ for groups $\mathbb{G}_1$ and $\mathbb{G}_T$ of prime order N. Our construction relies on the existence of a base group $\mathbb{G}_0$ in which the κ-SDDH assumption holds—this states that, given a $(\kappa+1)$-tuple of $\mathbb{G}_0$-elements $(g, g^{\omega}, \ldots, g^{\omega^{\kappa}})$, we cannot efficiently distinguish $g^{\omega^{\kappa+1}}$ from a random element of $\mathbb{G}_0$. Under this assumption, we prove that the κ-MDDH problem, a natural analogue of the DDH problem as stated below, is hard.

> **(The κ-MDDH problem, informal)** Given a generator g_1 of $\mathbb{G}_1$ and $\kappa+1$ group elements $g_1^{a_i}$ in $\mathbb{G}$ with $a_i \leftarrow\!\!\$\, \mathbb{Z}_N$, distinguish $\mathbf{e}(g_1, \ldots, g_1)^{\prod_{i=1}^{\kappa+1} a_i}$ from a random element of $\mathbb{G}_T$.

This problem can be used as the basis for several cryptographic constructions [7], including by now the classic example of multiparty non-interactive key exchange (NIKE) [23].

Our other construction is for the *asymmetric* setting; that is, for multilinear maps $\mathbf{e} : \mathbb{G}_1 \times \cdots \times \mathbb{G}_{\kappa} \longrightarrow \mathbb{G}_T$ for a collection of κ groups $\mathbb{G}_i$ and $\mathbb{G}_T$ all of prime order N. It uses a base group $\mathbb{G}_0$ in which we require only that the 1-SDDH assumption holds. For this construction, we show that a natural asymmetric analogue of the κ-MDDH assumption holds.

At a high level, then, our constructions are able to "bootstrap" from rather mild assumptions in a standard cryptographic group to much stronger multilinear assumptions in a group (or groups, in the asymmetric setting) equipped with a κ-linear map. Here, κ is fixed up-front at the time of setup, but is otherwise unrestricted. Of course, such constructions cannot be expected to come "for free," and we need to make use of powerful tools including probabilistic IO (PIO) for obfuscating randomized circuits [17], dual-mode NIZK proofs enjoying perfect soundness (for a binding CRS), perfect witness-indistinguishability (for a hiding CRS), and perfect zero knowledge, and additive homomorphic encryption for the group $(\mathbb{Z}_N, +)$ (or alternatively, a perfectly correct FHE

scheme). We note that all these tools can be constructed from a (pair of) pairing-friendly groups (in which, e.g., the SXDH assumption holds), subexponentially secure one-way functions, and subexponentially secure IO. It is an important open problem arising from our work to weaken the requirements on, or remove altogether, these additional tools.

1.2. *General Approach*

Our approach to obtaining multilinear maps in the symmetric setting is as follows (with many details to follow in the main body).[1] Let $\mathbb{G}_0$ with generator g_0 be a group of prime order N in which the κ-SDDH assumption holds.

We work with redundant encodings of elements h of the base group $\mathbb{G}_0$ of the form $h = g_0^{x_0}(g_0^{\omega})^{x_1}$ where g_0^{ω} comes from a κ-SDDH instance; we write $\mathbf{x} = (x_0, x_1)$ for the vector of exponents *representing* h. Then, $\mathbb{G}_1$ consists of all strings of the form $(h, \mathbf{c}_1, \mathbf{c}_2, \pi)$ where $h \in \mathbb{G}_0$, ciphertext $\mathbf{c}_1$ is a homomorphic encryption under public key pk_1 of a vector $\mathbf{x}$ representing h, ciphertext $\mathbf{c}_2$ is a homomorphic encryption under a second public key pk_2 of another vector $\mathbf{y}$ also representing h, and π is a NIZK proof showing consistency of the two vectors $\mathbf{x}$ and $\mathbf{y}$, i.e., a proof that the plaintexts $\mathbf{x}, \mathbf{y}$ underlying $\mathbf{c}_1, \mathbf{c}_2$ encode the *same* group element h. Note that each element of the base group $\mathbb{G}_0$ is multiply represented when forming elements in $\mathbb{G}_1$, but that equality of group elements in $\mathbb{G}_1$ is easy to test. An alternative viewpoint is to consider $(\mathbf{c}_1, \mathbf{c}_2, \pi)$ as being *auxiliary information* accompanying element $h \in \mathbb{G}_0$; we prefer the perspective of redundant encodings, and our abstraction in Sect. 3 is stated in such terms. When viewed in this way, our approach can be seen as closely related to the Naor–Yung paradigm for constructing CCA-secure PKE [37].

Addition of two elements in $\mathbb{G}_1$ is carried out by an obfuscation of a circuit C_{Add} that is published along with the groups. It has the secret keys sk_1, sk_2 hard-coded in; it first checks the respective proofs, then uses the additive homomorphic property of the encryption scheme to combine ciphertexts, and finally uses the secret keys sk_1, sk_2 as witnesses to generate a new NIZK proof showing equality of encodings. Note that the new encoding is as compact as that of the two input elements.

The multilinear map on inputs $(h_i, \mathbf{c}_{i,1}, \mathbf{c}_{i,2}, \pi_i)$ for $1 \leq i \leq \kappa$ is computed using the obfuscation of a circuit C_{Map} that has sk_1 and ω hard-coded in. This allows C_{Map} to "extract" full exponents of h_i in the form $(x_{i,1} + \omega \cdot x_{i,2})$ from $\mathbf{c}_{i,1}$ and thereby compute the element $g_0^{\prod_i (x_{i,1} + \omega \cdot x_{i,2})}$. This is defined to be the output of our multilinear map $\mathbf{e}$, and so our target group $\mathbb{G}_T$ is in fact $\mathbb{G}_0$, the base group. The multilinearity of $\mathbf{e}$ follows immediately from the form of the exponent.

In the asymmetric case, the main difference is that we work with different values ω_i in each of our input groups $\mathbb{G}_i$. However, the groups are all constructed via redundant encodings, just as above.

This provides a high-level view of our approach, but no insight into why the approach achieves our aim of building multilinear maps with associated hard problems. Let us give some intuition on why the κ-MDDH problem is hard in our setting. We transform

[1] This version fixes a flaw that we found in the proof of Theorem 1 in the conference version of the paper. The construction of Sect. 4 has been slightly modified, but it does not make use of stronger assumptions and has comparable efficiency.

a κ-MDDH tuple $\mathbf{h} = ((g_1^{a_i})_{i \leq \kappa+1}, g_T^d)$, where d is the product of the $a_i \in \mathbb{Z}_N$, g_1 is in the "encoded" form above, and thus, $g_1 = (h_1, \mathbf{c}_1, \mathbf{c}_2, \pi)$, and g_T is a generator of $\mathbb{G}_T = \mathbb{G}_0$, into another κ-MDDH tuple $\mathbf{h}'$ with exponents $a_i' = a_i + \omega$ for $i \leq \kappa + 1$. This means that the exponent of the challenge element in the target group $d' = \prod_{i=1}^{\kappa+1}(a_i + \omega)$ can be seen as a degree $\kappa + 1$ polynomial in ω. Therefore, with the knowledge of the a_i and a κ-SDDH challenge, with ω implicit in the exponent, we are able to switch $g_T^{d'}$ to a uniformly random value.

Nevertheless, in the preceding simplistic argument, we have made two assumptions. The first is that we are able to provide an obfuscation of a circuit C'_{Map} that has the same functionality as C_{Map} over $\mathbb{G}_1$ *without* the explicit knowledge of ω. We resolve this by showing a way of evaluating the κ-linear map on any elements of $\mathbb{G}_1$ using only the powers $g_0^{\omega^i}$ for $1 \leq i \leq \kappa$, and vectors extracted from the accompanying ciphertexts, and then applying IO to the two circuits.[2]

The second assumption we made is that we can indeed switch from $\mathbf{h}$ to $\mathbf{h}'$ without being noticed. In other words, that the vectors $\mathbf{x}_i$, $\mathbf{y}_i$ representing g^{a_i} can be replaced (without being noticed) with vectors $\mathbf{h}'_i$ whose second coordinate is always fixed. Intuitively, this is based on the IND-CPA security of the FHE scheme, but in order to give a successful reduction, we also have to change the circuit C_{Add} (since C_{Add} uses both decryption keys) and apply probabilistic indistinguishability obfuscation [17] to the circuit.

We note that in this work, we do not construct graded encoding schemes as in [23]. That is, we do not construct maps from $\mathbb{G}_i \times \mathbb{G}_j$ to $\mathbb{G}_{i+j}$. On the other hand, our construction is noiseless and is closer to multilinear maps as defined by Boneh and Silverberg [7].

1.3. *The Current State of Multilinear Maps Constructions*

Multilinear maps have been in a state of turmoil, with the discovery of attacks [9,13,14, 30,36] against the GGH13 [23], CLT [15], and GGH15 [26] proposals, and a sequence of countermeasures and fixes [12,16], which since have been broken, too. Hence, our confidence in constructions for graded encoding schemes (and thereby multilinear maps) has been shaken. On the other hand, recently, several constructions of IO from increasingly weaker assumptions have been proposed (see, for example, [1,3,24,33–35,45]), culminating in the construction [35] that requires only trilinear (non-graded) multilinear maps.

Hence, currently it is perhaps more plausible to assume that IO exists than it is to assume that secure (multi-level) multilinear maps exist. However, we stress that more cryptanalysis of IO constructions is required to investigate what security they provide.

Moreover, even though current constructions for IO rely on graded encoding schemes, it is not implausible that alternative routes to achieving IO without relying on multilinear maps will emerge in due course. Furthermore, multilinear maps, and more generally graded encoding schemes, have proven to be very fruitful as constructive tools in their own right (cf. [7,40], resp., [5,8,22,25,27,31,42]). This rich set of applications coupled with the current uncertainty over the status of graded encoding schemes and multilinear

[2] This is not trivial since the new method should not lead to an exponential blowup in κ.

maps provides additional motivation to ask what additional tools are needed in order to upgrade IO to multilinear maps. As an additional benefit, we upgrade (via IO) noisy graded encoding schemes to clean multilinear maps—sometimes now informally called "dream" or "ideal" multilinear maps.

1.4. *Related Work*

The work that is technically closest to ours is that of Yamakawa et al. (see [43,44]); indeed, their work was the starting point for ours. Yamakawa et al. construct a *self-pairing map*, that is, a bilinear map from $\mathbb{G} \times \mathbb{G}$ to $\mathbb{G}$; multilinear maps can be obtained by iterating their self-pairing. Their work is limited to the RSA setting. It uses the group of signed quadratic residues modulo a Blum integer N, denoted QR_N^+, to define a pairing function that, on input elements g^x, g^y in QR_N^+, outputs g^{2xy}. In their construction, elements of QR_N^+ are augmented with auxiliary information to enable the pairing computation—in fact, the auxiliary information for an element g^x is simply an obfuscation of a circuit for computing the $2x$th power modulo $\mathrm{ord}(\mathrm{QR}_N^+)$, and the pairing is computed by evaluating this circuit on an input g^y (say). The main contribution of [43] is in showing that these obfuscated circuits leak nothing about x or the group order.

A nice feature of their scheme is that the degree of linearity κ that can be accommodated is not limited up-front in the sense that the pairing output is also a group element to which further pairing operations (derived from auxiliary information for other group elements) can be applied. However, the construction has several drawbacks. First, the element output by the pairing does not come with auxiliary information.[3] Second, the size of the auxiliary information for a product of group elements grows exponentially with the length of the product, as each single product involves computing the obfuscation of a circuit for multiplying, with its inputs already being obfuscated circuits. Third, the main construction in [43] only builds hard problems for the self-pairing of the computational type. (In fact, they show the hardness of the computational version of the κ-MDDH problem in QR_N^+ assuming that factoring is hard.) Still, this is sufficient for several cryptographic applications.

In contrast, our construction is *generic* with respect to its platform group. Furthermore, the equivalent of the auxiliary information in our approach does not itself involve any obfuscation. Consequently, the description of a product of group elements stays compact. Indeed, given perfect additive homomorphic encryption for $(\mathbb{Z}_p, +)$, we can perform arbitrary numbers of group operations in each component group $\mathbb{G}_i$. It is an open problem to find a means of augmenting our construction with the equivalent of auxiliary information in the *target* group $\mathbb{G}_T$, to make our multilinear maps amenable to iteration and thereby achieve graded maps as per [15,23].

[3] The authors of [43] state that such information can be added in their construction, but what would be needed is the obfuscation of a circuit for computing $4xy$th powers. The information available for building this would be obfuscations of circuits for computing $2x$th and $2y$th powers, so an obfuscation of a *composition* of *already* obfuscated circuits would be required. Strictly speaking then, the auxiliary information associated with elements output by their pairing is of a different type to that belonging to the inputs, making it questionable whether "self-pairing" is the right description of what is constructed in [43].

Another related work is the work of Paneth and Sahai [39]. They show a near equivalence between a suitable abstraction of multilinear maps and IO. Their result requires no computational assumptions at all, but also does not consider multilinear maps in our sense. In particular, they construct an abstraction of a multilinear map that only admits restricted access to encodings, similar to the one in [24]. Beyond the group operation and the multilinear map, other procedures for, e.g., uniform sampling, comparison or rerandomization of encodings are not part of this abstraction. Our notion of a multilinear map, on the other hand, contains descriptions of efficient procedures for all of these tasks.

1.5. *Follow-Up Work*

The work [21] extends our approach from this work to *graded* encoding schemes (with multilinear maps). They use techniques similar to ours and in particular employ a suitable "switching theorem" (like our Theorem 1) to replace encodings of equivalent group elements.

On the other hand, the work [2] aims to construct groups (or, rather, encoding schemes) that support stronger computational assumptions. Specifically, [2] construct encoding schemes in which even an adaptive variant of the so-called Uber assumption [6] holds. The price that [2] pay is that their encoding scheme has no extraction algorithm (i.e., no algorithm that takes an encoding and outputs a bit string that is unique for the encoded group element). Not only such an extraction algorithm is useful to compare elements, it can also be used to transform non-unique group elements to a unique common secret in a Diffie–Hellman key exchange protocol. Observe that with non-unique group elements and without such an extraction algorithm, the two parties may end with different representations of the same shared key.

In this setting, the only means to compare two group elements (given by possibly different encodings) is an explicit comparison algorithm that takes two encodings as input and outputs whether these encodings represent the same group element. ([2] provide such a comparison algorithm.) The techniques that [2] use are again an extension of our techniques.

1.6. *Relation to Conference Version of This Work*

Erratum. After the publication of the conference version of this work at TCC 2016-A, we became aware of several technical problems in our work. Specifically, the conference version of our work (and of course a previous full version) claimed (a) the validity of the RANK assumption (a reformulation of the $\mathcal{U}_n$-matrix Diffie–Hellman assumption from [19]) in our framework and (b) a variant of our construction that only uses indistinguishability obfuscation (instead of probabilistic indistinguishability obfuscation). We encountered serious problems in both respective proofs, and we are currently not aware of a way to repair these proofs.

Furthermore, we became aware of problems in the proof of the multilinear DDH assumption in our framework (both in the symmetric and asymmetric settings). These problems can be resolved, which in fact leads to a simpler proof from a slightly stronger computational assumption.

Hence, this version of our work omits the results (a) and (b) described above and provides corrected versions of the proofs of the MDDH assumption in our framework.

Changes to conference version. Besides the corrections explained above, this version features full proofs, and in particular a detailed and modular treatment of the central *switching theorem* (Theorem 1). Our constructed group has non-unique, randomized encodings in place of group elements, and Theorem 1 allows to replace one encoding with another encoding, as long as both encodings are functionally equivalent.

2. Preliminaries

2.1. *Notation*

We denote the security parameter by $\lambda \in \mathbb{N}$ and assume that it is implicitly given to all algorithms in the unary representation 1^λ. By an algorithm, we mean a stateless Turing machine. Algorithms are randomized unless stated otherwise, and PPT as usual stands for "probabilistic polynomial-time" in the (unary) security parameter. Given a randomized algorithm $\mathcal{A}$, we denote the action of running $\mathcal{A}$ on input(s) $(1^\lambda, x_1, \ldots)$ with fresh random coins r and assigning the output(s) to $y_1, \ldots$ by $(y_1, \ldots) \leftarrow\!\$\, \mathcal{A}(1^\lambda, x_1, \ldots; r)$. For a finite set X, we denote its cardinality by $|X|$ and the action of sampling a uniformly random element x from X by $x \leftarrow\!\$\, X$. Vectors are written in boldface $\mathbf{x}$ and by slight abuse of notation, running algorithms on vectors of elements indicates component-wise operation. Throughout the paper, $\perp$ denotes a special error symbol, and $\mathrm{poly}(\cdot)$ stands for a fixed polynomial. A real-valued function $\mathrm{negl}(\lambda)$ is negligible if $\mathrm{negl}(\lambda) \in \mathcal{O}(\lambda^{-\omega(1)})$. We denote the set of all negligible functions by NEGL and use $\mathrm{negl}(\lambda)$ to denote an unspecified negligible function.

2.2. *Homomorphic Public-Key Encryption*

CIRCUITS. A polynomial-sized deterministic circuit family $\mathcal{C} := \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ is a sequence of sets of $\mathrm{poly}(\lambda)$-sized circuits for a fixed polynomial poly. We assume that for all $\lambda \in \mathbb{N}$, all circuits $C \in \mathcal{C}_\lambda$ share a common input domain $(\{0,1\}^\lambda)^{a(\lambda)}$, where $a(\lambda)$ is a the arity of the circuit family, and codomain $\{0,1\}^\lambda$. A randomized circuit family is defined similarly except that the circuits now also take random coins $r \in \{0,1\}^{r(\lambda)}$. To make the coins used by a circuit explicit (e.g., to view a randomized circuit as a deterministic one), we write $C(x; r)$.

SYNTAX AND COMPACTNESS. A tuple of PPT algorithms $\Pi :=$ (**Gen**, **Enc**, **Dec**, **Eval**) is called a homomorphic public-key encryption (HPKE) scheme for deterministic circuit family $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ of arity $a(\lambda)$ if (**Gen**, **Enc**, **Dec**) is a conventional public-key encryption scheme with message space $\{0,1\}^\lambda$ and **Eval** is a *deterministic* algorithm that on input, a public key pk, a circuit $C \in \mathcal{C}_\lambda$ and ciphertexts $c_1, \ldots, c_{a(\lambda)}$ output a ciphertext c. We require HPKE schemes to be *compact* in the sense that the outputs of **Eval** have a size that is bounded by a polynomial function of the security parameter (and independent of the size of the circuit). Without loss of generality, we assume that secret

keys of an HPKE scheme are the random coins used in key generation. This will allow us to check key pairs for validity.

CORRECTNESS. We require the following *perfect* correctness requirements from a HPKE scheme:

1. Scheme $\Pi := (\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ is perfectly correct as a PKE scheme; that is, for any $\lambda \in \mathbb{N}$, any $(sk, pk) \leftarrow_\$ \mathbf{Gen}(1^\lambda)$, any $m \in \{0,1\}^\lambda$, and any $c \leftarrow_\$ \mathbf{Enc}(m, pk)$, we have that $\mathbf{Dec}(c, sk) = m$.
2. We furthermore require that *any* ciphertext (i.e., not only honestly generated ones) uniquely determines the message it decrypts to. That is, for any $\lambda \in \mathbb{N}$, any pk in the range of $\mathbf{Gen}(1^\lambda)$, and any syntactically possible c, there is precisely one m (which may be $m = \perp$), such that for all sk with $(pk, sk) = \mathbf{Gen}(sk)$, we have $\mathbf{Dec}(c, sk) = m$.
3. The evaluation algorithm is also perfectly correct in the sense that for any $\lambda \in \mathbb{N}$, any $(sk, pk) \leftarrow_\$ \mathbf{Gen}(1^\lambda)$, any $m_i \in \{0,1\}^\lambda$ for $i \in [a(\lambda)]$, any $c_i \leftarrow_\$ \mathbf{Enc}(m_i, pk)$, any $C \in \mathcal{C}_\lambda$ and any $c \leftarrow \mathbf{Eval}(pk, C, c_1, \ldots, c_{a(\lambda)})$, we have that $\mathbf{Dec}(c, sk) = C(m_1, \ldots, m_{a(\lambda)})$.

We note that perfect correctness implies that every ciphertext (even an adversarially generated one) uniquely determines its decryption result, independently of the used secret key (for a given public key). Hence, it is reasonable to think of any ciphertext as "containing" a uniquely defined message (as long as only secret keys consistent with a given public key are used).

SECURITY. The IND-CPA security of an HPKE scheme is defined identically to a standard PKE scheme without reference to the **Dec** and **Eval** algorithms. Formally, we require that for any legitimate PPT adversary $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$,

$$\mathbf{Adv}^{\text{ind-cpa}}_{\Pi,\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[\text{IND-CPA}^{\mathcal{A}}_{\Pi}(\lambda)\right] - 1 \in \text{NEGL}\,,$$

where game $\text{IND-CPA}^{\mathcal{A}}_{\Pi}(\lambda)$ is shown in Fig. 1 (left). Adversary $\mathcal{A}$ is legitimate if it outputs two messages of equal lengths.

HPKE schemes can be constructed from rerandomizable IND-CPA secure PKE schemes, subexponentially secure IO, and subexponentially secure one-way functions [17]. The correctness properties of this construction immediately follow from those of its underlying components. Although this HPKE construction may not be perfectly correct in our sense above, when used with ElGamal (which is rerandomizable and IND-CPA secure under the DDH assumption), it *does* satisfy our notion of perfect correctness.

2.3. *Obfuscators*

SYNTAX AND CORRECTNESS. A PPT algorithm **Obf** is called an *obfuscator* for (deterministic or randomized) circuit class $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ if **Obf** on input the security parameter 1^λ and the description of a (deterministic or randomized) circuit $C \in \mathcal{C}_\lambda$ outputs a deterministic circuit $\overline{C}$. For deterministic circuits, we require **Obf** to be perfectly correct in

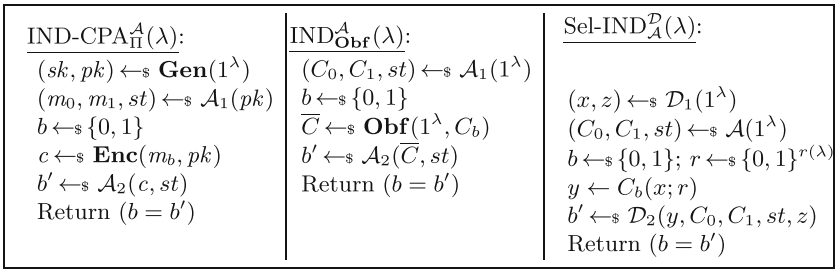

Fig. 1. *Left*: IND-CPA security of a (homomorphic) PKE scheme. *Middle*: Indistinguishability security of an obfuscator. *Right*: Static input (aka. selective) X-IND property of $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$.

the sense the circuits C and $\overline{C}$ are functionally equivalent; that is, that for all $\lambda \in \mathbb{N}$, all $C \in \mathcal{C}_\lambda$, all $\overline{C} \leftarrow\$ \mathbf{Obf}(1^\lambda, C)$, and all $m_i \in \{0,1\}^\lambda$ for $i \in [a(\lambda)]$, we have that $C(m_1, \ldots, m_{a(\lambda)}) = \overline{C}(m_1, \ldots, m_{a(\lambda)})$. For randomized circuits, the authors of [17] define correctness via computational indistinguishability of the outputs of C and $\overline{C}$. For our constructions, we do *not* rely on this property and instead require that C and $\overline{C}$ are functionally equivalent up to a change in randomness; that is, for all $\lambda \in \mathbb{N}$, all $C \in \mathcal{C}_\lambda$, all $\overline{C} \leftarrow\$ \mathbf{Obf}(1^\lambda, C)$ and all $m_i \in \{0,1\}^\lambda$ for $i \in [a(\lambda)]$, we require there is an r such that $\overline{C}(m_1, \ldots, m_{a(\lambda)}) = C(m_1, \ldots, m_{a(\lambda)}; r)$. In this paper by correctness, we refer to this latter property. We note that the construction from [17] is correct as it relies on a correct (indistinguishability) obfuscator (and a PRF to internally generate the required random coins).

Security. The security of an obfuscator **Obf** requires that for any legitimate PPT adversary $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$

$$\mathbf{Adv}^{\text{ind}}_{\mathbf{Obf},\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[\text{IND}^{\mathcal{A}}_{\mathbf{Obf}}(\lambda)\right] - 1 \in \text{Negl}\,,$$

where game IND is shown in Fig. 1 (middle). Depending on the notion of legitimacy different security notions for the obfuscator emerge, we consider two such notions below.
Functionally equivalent samplers. We call (the first phase of) $\mathcal{A}$ a *functionally equivalent sampler* if for any (possibly unbounded) distinguisher $\mathcal{D}$

$$\mathbf{Adv}^{\text{eq\$}}_{\mathcal{A},\mathcal{D}}(\lambda) := \Pr\left[C_0(x) \neq C_1(x) : (C_0, C_1, st) \leftarrow\$ \mathcal{A}_1(1^\lambda); x \leftarrow\$ \mathcal{D}(C_0, C_1, st)\right] \in \text{Negl}.$$

The security notion associated with equivalent samplers is called *indistinguishability*. We call an obfuscator meeting this level of security an *indistinguishability obfuscator* [24] and use **IO** instead of **Obf** to emphasize this.
X-IND samplers [17] Roughly speaking, $\mathcal{A}$ is an X-IND sampler if there is a set $\mathcal{X}$ of size at most X such that the circuits output by $\mathcal{A}$ are functionally equivalent outside $\mathcal{X}$ and furthermore within $\mathcal{X}$ the outputs of the two sampled circuits are indistinguishable. Formally, let $X(\cdot)$ be a function such that $X(\lambda) \leq 2^\lambda$ for all $\lambda \in \mathbb{N}$. We call $\mathcal{A}$ an X-IND *sampler* if there is a set $\mathcal{X}_\lambda$ of size at most $X(\lambda)$ such that the following two conditions hold: (1) for all (possibly unbounded) $\mathcal{D}$, the advantage function below is negligible

$$\mathbf{Adv}^{\text{eq\$}}_{\mathcal{A},\mathcal{D}}(\lambda) := \Pr\left[C_0(x;r) \neq C_1(x;r) \wedge x \notin \mathcal{X}_\lambda : (C_0, C_1, st) \leftarrow\!\!\$\, \mathcal{A}(1^\lambda);\ (x, r) \leftarrow\!\!\$\, \mathcal{D}(C_0, C_1, st)\right].$$

(2) For all non-uniform PPT distinguishers $\mathcal{D} := (\mathcal{D}_1, \mathcal{D}_2)$

$$X(\lambda) \cdot \mathbf{Adv}^{\text{sel-ind}}_{\mathcal{A},\mathcal{D}}(\lambda) := X(\lambda) \cdot \Pr\left[\text{Sel-IND}^{\mathcal{D}}_{\mathcal{A}}(1^\lambda)\right] \in \text{NEGL}\,,$$

where game $\text{Sel-IND}^{\mathcal{D}}_{\mathcal{A}}(1^\lambda)$ is shown in Fig. 1 (right). This game has a static (or selective) flavor as $\mathcal{D}_1$ chooses a differing input x *before* it gets to see the challenge circuit pair. We call an obfuscator meeting this level of security a *probabilistic indistinguishability obfuscator* [17] and use **PIO** instead of **Obf** to emphasize this.

[17] show how to construct secure probabilistic indistinguishability obfuscators for X-IND samplers from subexponentially secure indistinguishability obfuscation and subexponentially secure one-way functions.

2.4. *Dual-Mode NIZK Proof Systems*

In our constructions, we will be relying on special types of non-interactive zero-knowledge proof systems [29]. These systems have "dual-mode" common reference string (CRS) generation algorithms that produce indistinguishable CRSs in the "binding" and "hiding" modes. They also enjoy perfect completeness in both modes, are perfectly sound and extractable in the binding mode, and perfectly witness-indistinguishable (WI) and zero knowledge (ZK) in the hiding mode. The standard prototype for such schemes is pairing-based Groth–Sahai proofs [29]. These proof systems can be instantiated in any (pair of) pairing-friendly groups, under a variety of computational assumptions, including the SXDH and k-Linear assumptions. Moreover, using a generic NP reduction to the satisfiability of (systems of) quadratic equations, we can obtain a suitable proof system for any NP language.[4] We formalize the syntax and security of such proof systems next.

SYNTAX. A relation with setup is a pair of PPT algorithms $(\mathbf{S}, \mathbf{R})$ such that $\mathbf{S}(1^\lambda)$ outputs (gpk, gsk) and $\mathbf{R}(gpk, x, w)$ is a ternary relation that outputs a bit $b \in \{0, 1\}$. A dual-mode non-interactive zero-knowledge (NIZK) proof system Σ for $(\mathbf{S}, \mathbf{R})$ consists of six algorithms as follows. (1) Algorithm $\mathbf{BCRS}(gpk, gsk)$ outputs a (binding) common reference string crs and an extraction trapdoor td_{ext}; (2) $\mathbf{HCRS}(gpk, gsk)$ outputs a (hiding) common reference string crs and a simulation trapdoor td_{zk}; (3) $\mathbf{Prove}(gpk, crs, x, w)$ on input crs, an instance x, and a witness w, outputs a proof π; (4) $\mathbf{Verify}(gpk, crs, x, \pi)$ on input a bit string crs, an instance x, and a proof π, outputs accept or reject; (5) $\mathbf{WExt}(td_{ext}, x, \pi)$ on input an extraction trapdoor, an instance x, and a proof π, outputs a witness w; and (6) $\mathbf{Sim}(td_{zk}, crs, x)$ on input the simulation trapdoor td_{zk}, the CRS crs, and an instance x, outputs a simulated proof π. We require a dual-mode NIZK to meet the following requirements.

[4] We note that extraction in Groth–Sahai proofs does not recover a witness for all types of statements. (Instead, for some types of statements, only g^{w_i} for a witness variable $w_i \in \mathbb{Z}_p$ can be recovered.) Here, however, we will only be interested in witnesses $w = (w_1, \ldots, w_n) \in \{0, 1\}^n$ that are bit strings, in which case extraction always recovers w. (Specifically, extraction will recover g^{w_i} for all i and thus all w_i.)

CRS INDISTINGUISHABILITY. The common reference strings generated through **BCRS**(gpk, gsk) and **HCRS**(gpk, gsk) are computationally indistinguishable. We denote the distinguishing advantage of a PPT adversary $\mathcal{A}$ in the relevant security game by $\mathbf{Adv}^{\mathrm{crs}}_{\Sigma,\mathcal{A}}(\lambda)$.

PERFECT COMPLETENESS UNDER **BCRS/HCRS**. For any $\lambda \in \mathbb{N}$, any $(gpk, gsk) \leftarrow\!\!\$\ \mathbf{S}(1^\lambda)$, any $crs \leftarrow\!\!\$\ \mathbf{BCRS}(gpk, gsk)$, any (x, w) such that $\mathbf{R}(gpk, x, w) = 1$, and any $\pi \leftarrow\!\!\$\ \mathbf{Prove}(gpk, crs, x, w)$, we have that $\mathbf{Verify}(gpk, crs, x, \pi) = 1$. We require this property to also hold for any choice of $crs \leftarrow\!\!\$\ \mathbf{HCRS}(gpk, gsk)$.

PERFECT SOUNDNESS UNDER **BCRS**. For any $\lambda \in \mathbb{N}$, any $(gpk, gsk) \leftarrow\!\!\$\ \mathbf{S}(1^\lambda)$, any common reference string $crs \leftarrow\!\!\$\ \mathbf{BCRS}(gpk, gsk)$, any x for which for all $w \in \{0,1\}^*$, we have $\mathbf{R}(gpk, x, w) = 0$, and any $\pi \in \{0,1\}^*$, we have that $\mathbf{Verify}(gpk, crs, x, \pi) = 0$.

PERFECT EXTRACTABILITY UNDER **BCRS**. For any $\lambda \in \mathbb{N}$, any $(gpk, gsk) \leftarrow\!\!\$\ \mathbf{S}(1^\lambda)$, any $(crs, td_{ext}) \leftarrow\!\!\$\ \mathbf{BCRS}(gpk, gsk)$, any tuple (x, π) with $\mathbf{Verify}(gpk, crs, x, \pi) = 1$, and for $w \leftarrow\!\!\$\ \mathbf{WExt}(td_{ext}, x, \pi)$, we always have $\mathbf{R}(gpk, x, w) = 1$.

PERFECT WI UNDER **HCRS**. For any $\lambda \in \mathbb{N}$, any $(gpk, gsk) \leftarrow\!\!\$\ \mathbf{S}(1^\lambda)$, any $(crs, td_{zk}) \leftarrow\!\!\$\ \mathbf{HCRS}(gpk, gsk)$, any (x, w_b) such that $\mathbf{R}(gpk, x, w_b) = 1$ for $b \in \{0,1\}$, we have that $\pi_b \leftarrow\!\!\$\ \mathbf{Prove}(gpk, crs, x, w_b)$ for $b \in \{0,1\}$ are identically distributed.

PERFECT ZK UNDER **HCRS**. For any $\lambda \in \mathbb{N}$, any $(gpk, gsk) \leftarrow\!\!\$\ \mathbf{S}(1^\lambda)$, any $(crs, td_{zk}) \leftarrow\!\!\$\ \mathbf{HCRS}(gpk, gsk)$, any (x, w) such that $\mathbf{R}(gpk, x, w) = 1$, it is that $\pi_0 \leftarrow\!\!\$\ \mathbf{Prove}(gpk, crs, x, w)$ and $\pi_1 \leftarrow\!\!\$\ \mathbf{Sim}(td_{zk}, x)$ are identically distributed.

2.5. *Hard Membership Problems*

Finally, we will use languages with hard membership problems. Let $U = \{U_\lambda\}$ be a collection of universal sets, and let $\mathcal{L} = \{\mathcal{L}_\lambda\}$ be a collection of sets $\mathcal{L}_\lambda = \{L\}$ of languages with $L \subseteq U_\lambda$ for each $L \in \mathcal{L}_\lambda$. We say that $\mathcal{L}$ has a hard subset membership problem if the following holds: No PPT algorithm given $L \leftarrow\!\!\$\ \mathcal{L}_\lambda$ can efficiently distinguish between $x \leftarrow\!\!\$\ L$ and $x \leftarrow\!\!\$\ U_\lambda$.

3. Multilinear Groups with Non-Unique Encodings

Before presenting our constructions, we formally introduce what we mean by a multilinear group (MLG) scheme. Our abstraction differs from that of Garg, Gentry, and Halevi [23] in that our treatment of MLG schemes is a direct adaptation of the "dream" MLG setting (called the "cryptographic" MLG setting in [7]) to a setting where group elements have *non-unique* encodings (as in [23]). In our abstraction, on top of the procedures needed for generating, manipulating, and checking group elements, we introduce an *equality* procedure which generalizes the equality relation for groups with unique encodings.

SYNTAX. A multilinear group (MLG) scheme Γ consists of six PPT algorithms as follows.

Setup$(1^\lambda, 1^\kappa)$: This is the setup algorithm. On input the security parameter 1^λ and the multilinearity 1^κ, it outputs the group parameters pp. These parame-

ters include *generators* $g_1, \ldots, g_{\kappa+1}$, *identity elements* $\mathsf{1}_1, \ldots, \mathsf{1}_{\kappa+1}$, and integers $N_1, \ldots, N_{\kappa+1}$, which will represent group orders. Generators, identity elements, and group orders are discussed in detail below. In our constructions, we will have $N_1, \ldots, N_{\kappa+1}$ all equal to some prime N, but we work here at a greater level of generality because it may be useful in future work. We assume pp is provided to the various algorithms below.

$\mathbf{Val}_i(h)$: This is the validity testing algorithm. On input (the group parameters), a group index $1 \leq i \leq \kappa + 1$ and a string $h \in \{0,1\}^*$, it returns $b \in \{0,1\}$. We define $\mathbb{G}_i$, which is also parameterized by pp, as the set of all h for which $\mathbf{Val}_i(h) = 1$. We write $h \in \mathbb{G}_i$ when $\mathbf{Val}_i(h) = 1$ and refer to such strings as *group elements* (since we will soon impose a group structure on $\mathbb{G}_i$). Without loss of generality, we assume the $\mathbb{G}_i$ to be non-intersecting sets (since a string $h \in \mathbb{G}_i$ can always be augmented with an encoding of i). We require that the bit strings in $\mathbb{G}_i$ have lengths that are polynomial in λ and κ, a property that we refer to as *compactness*.

$\mathbf{Eq}_i(h_1, h_2)$: This is the equality algorithm. On input two valid group elements $h_1, h_2 \in \mathbb{G}_i$, it outputs a bit $b \in \{0,1\}$. We require $\mathbf{Eq}_i$ to define an equivalence relation. We say that the group has unique encodings if $\mathbf{Eq}_i$ simply checks the equality of bit strings. We write $\mathbb{G}_i(h)$ for the set of all $h' \in \mathbb{G}_i$ such that $\mathbf{Eq}_i(h, h') = 1$; for any such h, h' in $\mathbb{G}_i$, we write $h = h'$; sometimes we write $h = h'$ *in* $\mathbb{G}_i$ for clarity. Since "=" refers to equality of bit strings as well as equivalence under $\mathbf{Eq}_i$, we will henceforth write "as bit strings" when we mean equality in that sense. We require $|\mathbb{G}_i/\mathbf{Eq}_i|$, the number of equivalence classes into which $\mathbf{Eq}_i$ partitions $\mathbb{G}_i$, to be finite and equal to N_i (where N_i comes from pp). We assume throughout the paper that various algorithms below return $\perp$ when run on invalid group elements.

$\mathbf{Op}_i(h_1, h_2)$: This algorithm defines the group operation. On input two valid group elements $h_1, h_2 \in \mathbb{G}_i$, it outputs $h \in \mathbb{G}_i$. We write $h_1 h_2$ in place of $\mathbf{Op}_i(h_1, h_2)$ for simplicity. We require that $\mathbf{Op}_i$ respect the equivalence relations $\mathbf{Eq}_i$, meaning that if $h_1 = h_2$ in $\mathbb{G}_i$ and $h \in \mathbb{G}_i$, then $h_1 h = h_2 h$ in $\mathbb{G}_i$. We also demand that $h_1 h_2 = h_2 h_1$ in $\mathbb{G}_i$ (commutativity); for any third $h_3 \in \mathbb{G}_i$, we require $h_1(h_2 h_3) = (h_1 h_2) h_3$ in $\mathbb{G}_i$ (associativity), and we require $h\mathsf{1}_i = h$ in $\mathbb{G}_i$ for all $h \in \mathbb{G}_i$.

The algorithm $\mathbf{Op}_i$ gives rise to an exponentiation algorithm $\mathbf{Exp}_i(h, z)$ that on input $h \in \mathbb{G}_i$ and $z \in \mathbb{N}$ outputs an $h' \in \mathbb{G}_i$ such that $h' = h \cdots h$ in $\mathbb{G}_i$ with z occurrences of h. When no h is specified, we assume $h = g_i$. This algorithm runs in polynomial time in the length of z. We denote $\mathbf{Exp}_i(h, z)$ by h^z and define $h^0 := \mathsf{1}_i$. Note that under the definition of N_i for any $h \in \mathbb{G}_i$ we have that $\mathbf{Exp}_i(h, N_i) = \mathsf{1}_i$. This in turn leads to an inversion algorithm $\mathbf{Inv}_i(h)$ that on input $h \in \mathbb{G}_i$ outputs $h^{N_i - 1}$. We insist that g_i in fact has order N_i, so that (the equivalence class containing) g_i generates $\mathbb{G}_i/\mathbf{Eq}_i$.

The above requirements ensure that $\mathbb{G}_i/\mathbf{Eq}_i$ acts as a cyclic group of order N_i with respect to the operation induced by $\mathbf{Op}_i$, with identity (the equivalence class containing) $\mathsf{1}_i$, and inverse operation $\mathbf{Inv}_i$.

We use the *bracket* notion [19] to denote an element $h = g_i^x$ in $\mathbb{G}_i$ with $[x]_i$. When using this notation, we will write the group law additively. This notation will be convenient in the construction and analysis of our MLG schemes. For example, $[z]_i + [z']_i$ succinctly denotes $\mathbf{Op}_i(\mathbf{Exp}(g_i, z), \mathbf{Exp}(g_i, z'))$. Note that when writing $[z]_i$, it is *not* necessarily the case that z is explicitly known.

$\mathbf{e}(h_1, \ldots, h_\kappa)$: This is the multilinear map algorithm. For κ group elements $h_i \in \mathbb{G}_i$ as input, it outputs $h_{\kappa+1} \in \mathbb{G}_{\kappa+1}$. We demand that for any $1 \leq j \leq \kappa$ and any $h'_j \in \mathbb{G}_j$

$$\mathbf{e}(h_1, \ldots, h_j h'_j, \ldots, h_\kappa) = \mathbf{e}(h_1, \ldots, h_j, \ldots, h_\kappa)\mathbf{e}(h_1, \ldots, h'_j, \ldots, h_\kappa) \text{ in } \mathbb{G}_{\kappa+1}.$$

We also require the map to be *non-degenerate* in the sense that for some tuple of elements as input the multilinear map outputs an element of $\mathbb{G}_{\kappa+1}$ outside the equivalence class of $\mathbf{1}_{\kappa+1}$. We call an MLG scheme *symmetric* if the group algorithms are independent of the group index for $1 \leq i \leq \kappa$ and $\mathbf{e}$ is invariant under permutations of its inputs. That is, for any permutation $\pi : [\kappa] \longrightarrow [\kappa]$, we have

$$\mathbf{e}(h_1, \ldots, h_\kappa) = \mathbf{e}(h_{\pi(1)}, \ldots, h_{\pi(\kappa)}) \text{ in } \mathbb{G}_{\kappa+1} .$$

We refer to all the other cases as being *asymmetric*. To distinguish the target group, we frequently write $\mathbb{G}_T$ instead of $\mathbb{G}_{\kappa+1}$ (and similarly for $\mathbf{1}_T$ and g_T in place of $\mathbf{1}_{\kappa+1}$ and $g_{\kappa+1}$) as its structure in our construction will be different from that of the source groups $\mathbb{G}_1, \ldots, \mathbb{G}_\kappa$.

$\mathbf{Sam}_i(z)$: This is the sampling algorithm. On input $z \in \mathbb{N}$, it outputs some $h \in \mathbb{G}_i$. We also allow a special input ε to this algorithm, in which case the sampler is required to output some $h \in \mathbb{G}_i$ together with a uniformly distributed z such that $h \in \mathbb{G}_i(g_i^z)$. Note that for groups with unique encodings, these algorithms trivially exist. For notational convenience, for a known a, we define $[a]_i$ to be an element sampled via $\mathbf{Sam}_i(a)$.

Some applications also rely on the following algorithm which provides a canonical bit string for the group elements within a single equivalence class.

$\mathbf{Ext}_i(h)$: This is the extraction algorithm. On input $h \in \mathbb{G}_i$, it outputs a string $s \in \{0,1\}^{\mathrm{poly}(\lambda)}$. We demand that for any $h_1, h_2 \in \mathbb{G}_i$ with $h_1 = h_2$ in $\mathbb{G}_i$, we have that $\mathbf{Ext}_i(h_1) = \mathbf{Ext}_i(h_2)$ (as bit strings). We also require that for $[z]_i \leftarrow\!\$\, \mathbf{Sam}_i(\varepsilon)$, the distribution of $\mathbf{Ext}_i([z]_i)$ is uniform over $\{0,1\}^{\mathrm{poly}(\lambda)}$. For groups with unique encodings, this algorithm trivially exists.

Comparison with GGH. Our formalization differs from that of [23] which defines a *graded encoding scheme*. The main difference is that a graded encoding scheme defines bilinear maps $\mathbf{e}_{i,j} : \mathbb{G}_i \times \mathbb{G}_j \longrightarrow \mathbb{G}_{i+j}$. Using this algorithm, one can implement $\mathbf{Eq}_i$ for any $1 \leq i \leq \kappa$ from $\mathbf{Eq}_{\kappa+1}$ as follows (if $\mathbf{e}_{i,j}$ is injective). To check the equality of $h_1, h_2 \in \mathbb{G}_i$, call $\mathbf{e}_{i,\kappa+1-i}(h, g_{\kappa+1-i})$ for $h = h_1, h_2$ to map these elements to the target group and check equality there using $\mathbf{Eq}_{\kappa+1}$. Similarly, $\mathbf{Ext}_i(h)$ can be constructed from $\mathbf{Ext}_{\kappa+1}(h)$ and $\mathbf{1}_j$ for all $\mathbb{G}_j$. (Note that for extraction we need a canonical *string* rather than a canonical group element.) Moreover, the abstraction and construction of graded encodings schemes in [23] do not provide any validity algorithms; these are useful in certain adversarial situations such as CCA security and signature verification. Further, all known candidate constructions of graded encoding schemes are noisy and only permit a limited number of group operations (though parameters can be set to allow that number to be polynomial). Finally, the known candidate graded encoding schemes do not permit sampling for specific values of z, but rather only permit sampling elements with a z that is only known up to its equivalence class.

SYNTACTIC EXTENSIONS. Although our syntax does not treat the cases of graded [15,23], exponentially multilinear, or self-pairing [43] maps, it can be modified to capture these variants. We briefly outline the required modifications. For graded maps, we require the existence of a map that on input $h_i \in \mathbb{G}_i$ for indices $i = i_1, \ldots, i_\ell$ with $t := \sum_{i=1}^{\ell} i_j \leq \kappa$ outputs a group element in $\mathbb{G}_t$. This map is required to be multilinear in each component. For exponential (aka. unbounded) linearity, we provide the linearity κ in its binary representation to the **Setup** algorithm. We also include procedures for generator and identity element generation. Proper self-pairing maps correspond to a setting where the group algorithms are independent of the group index for $1 \leq i \leq \kappa + 1$ (including the target index $\kappa + 1$), and the group generators and identity elements are all identical. Observe that a proper self-pairing would induce a graded encoding scheme of unbounded linearity; recall from the introduction that the scheme of Yamakawa et al. [43] does not meet this definition because of the growth in the size of its auxiliary information.

4. The Construction

We now present our construction of an MLG scheme Γ according to the syntax introduced in Sect. 3. In the later sections, we will consider special cases of the construction and prove the hardness of analogues of the multilinear DDH problem under various assumptions.

We rely on the following building blocks in our MLG scheme. (1) A cyclic group $\mathbb{G}_0$ of some order N_0 with generator g_0 and identity 1_0; formally, we think of this as a 1-linear MLG scheme Γ_0 with unique encodings in which $\mathbf{e}$ is trivial; the algorithm $\mathbf{Val}_0$ implies that elements of $\mathbb{G}_0$ are efficiently recognizable. (2) A general-purpose obfuscator **Obf**. (3) A perfectly correct additively homomorphic public-key encryption scheme $\Pi := (\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec}, \mathbf{Eval})$ with plaintext space $\mathbb{Z}_N$.[5] (4) A dual-mode NIZK proof system. (5) A family $\mathcal{TD}$ of (families of) languages TD which has a hard subset membership problem, and such that all TD have efficiently computable witness relations with unique witnesses.[6] (See Sect. 2 for more formal definitions.)

We reserve variables and algorithms with index 0 for the base scheme Γ_0; we also write $N = N_0$. We require that the algorithms of Γ_0 except for $\mathbf{Setup}_0$ and $\mathbf{Sam}_0$ are deterministic. We will also use the bracket notation to denote the group elements in $\mathbb{G}_0$. For example, we write $[z]_0, [z']_0 \in \mathbb{G}_0$ for two valid elements of the base group and $[z]_0 + [z']_0 \in \mathbb{G}_0$ for $\mathbf{Op}_0([z]_0, [z']_0)$. Variables with nonzero indices correspond to various source and target groups. Given all of the above components, our MLG scheme Γ consists of algorithms as detailed in the sections that follow.

4.1. *Setup*

The setup algorithm for Γ samples parameters $pp_0 \leftarrow\!\!\$\ \mathbf{Setup}_0(1^\lambda)$ for the base MLG scheme generates two encryption key pairs $(pk_j, sk_j) \leftarrow\!\!\$\ \mathbf{Gen}(1^\lambda)$ $(j = 1, 2)$ of an

[5] Note that such a scheme can be constructed from any perfectly correct HPKE scheme.

[6] An example of such a language is the Diffie–Hellman language $\mathsf{TD} = \{(g_1^r, g_2^r) \mid r \in \mathbb{N}\}$ in a DDH group with generators g_1, g_2. In particular, a suitable trapdoor language imposes no additional computational assumption in our upcoming security proof.

HPKE scheme, and a matrix $\mathbf{W} = (\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_\kappa)^t \in \mathbb{Z}_N^{\kappa \times \ell}$ where κ is the linearity and $\ell = 2$ is a parameter of our construction. Although many of the upcoming results hold for more general distributions of $\mathbf{W}$, for concreteness, we set $\boldsymbol{\omega}_i = (1, \omega)$ for all i and a uniformly random ω. The setup algorithm then sets

$$gpk := (pp_0, pk_1, pk_2, [\mathbf{W}]_0, \mathsf{TD}, y) \ ,$$

where $[\mathbf{W}]_0$ denotes a matrix of $\mathbb{G}_0$ elements that entrywise is written in the bracket notation, $\mathsf{TD} \leftarrow\!\!\$\, \mathcal{TD}$, and y is *not* in TD. In our MLG scheme, we set $N_1 = \cdots = N_{\kappa+1} := N$, where N is the group order implicit in pp_0. The setup algorithm then generates a common reference string $crs = (crs', y)$ where $crs' \leftarrow\!\!\$\, \mathbf{BCRS}(gpk, gsk)$ for a relation $(\mathbf{S}, \mathbf{R})$ that will be defined in Sect. 4.2. It also constructs two obfuscated circuits $\overline{C}_{\mathrm{Map}}$ and $\overline{C}_{\mathrm{Add}}$ which we will describe in Sects. 4.3 and 4.4. For $1 \le i \le \kappa$, the identity elements $\mathsf{1}_i$ and group generators g_i are sampled using $\mathbf{Sam}_i(0)$ and $\mathbf{Sam}_i(x_i)$, respectively, for algorithm $\mathbf{Sam}_i$ described in Sect. 4.5 with $x_i \in [N-1]$. We emphasize that this approach is well defined since the operation of $\mathbf{Sam}_i$ is defined independently of the generators and the identity elements and depends only on gpk and crs. We set $\mathsf{1}_{\kappa+1} = \mathsf{1}_0$ and $g_{\kappa+1} = g_0$. The scheme parameters are

$$pp := (gpk, crs, \overline{C}_{\mathrm{Map}}, \overline{C}_{\mathrm{Add}}, g_1, \ldots, g_{\kappa+1}, \mathsf{1}_1, \ldots, \mathsf{1}_{\kappa+1}) \ .$$

We note that this algorithm runs in polynomial time in λ as long as κ is polynomial in λ.

4.2. *Validity and Equality*

The elements of $\mathbb{G}_i$ for $1 \le i \le \kappa$ are tuples of the form $h = ([z]_0, \mathbf{c}_1, \mathbf{c}_2, \pi)$ where $\mathbf{c}_1, \mathbf{c}_2$ are encryptions of vectors from $\mathbb{Z}_N^\ell$ under pk_1, pk_2, respectively (encryption algorithm $\mathbf{Enc}$ extends from plaintext space $\mathbb{Z}_N$ to $\mathbb{Z}_N^\ell$ in the obvious way) and where π is a NIZK to be defined below. We refer to $(\mathbf{c}_1, \mathbf{c}_2, \pi)$ as the *auxiliary information* for $[z]_0$. The elements of $\mathbb{G}_{\kappa+1}$ are just those of $\mathbb{G}_0$.

The NIZK proof system that we use corresponds to the following inclusive disjunctive relation $(\mathbf{S}, \mathbf{R} := \mathbf{R}_1 \vee \mathbf{R}_2)$. Algorithm $\mathbf{S}(1^\lambda)$ outputs $gpk = (pp_0, pk_1, pk_2, [\mathbf{W}]_0, \mathsf{TD})$ as defined above and sets $gsk = (sk_1, sk_2)$. Relation $\mathbf{R}_1$ on input gpk, tuple $([z]_0, \mathbf{c}_1, \mathbf{c}_2)$, and witness $(\mathbf{x}, \mathbf{y}, \mathbf{r}_1, \mathbf{r}_2, sk_1, sk_2)$ accepts iff $[z]_0 \in \mathbb{G}_0$, the *representations* of $[z]_0$ as $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_N^\ell$ are valid with respect to $[\mathbf{W}]_0$ in the sense that

$$[z]_0 = [\langle \mathbf{x}, \boldsymbol{\omega}_i \rangle]_0 \wedge [z]_0 = [\langle \mathbf{y}, \boldsymbol{\omega}_i \rangle]_0 \ ,$$

(where $\langle \cdot, \cdot \rangle$ denotes inner product) and the following ciphertext validity condition (with respect to the inputs to the relation) is met:

$$\begin{gathered} (\mathbf{c}_1 = \mathbf{Enc}(\mathbf{x}, pk_1; \mathbf{r}_1) \wedge \mathbf{c}_2 = \mathbf{Enc}(\mathbf{y}, pk_2; \mathbf{r}_2)) \\ \vee \\ \big((pk_1, sk_1) = \mathbf{Gen}(sk_1) \wedge (pk_2, sk_2) = \mathbf{Gen}(sk_2) \\ \wedge \mathbf{x} = \mathbf{Dec}(\mathbf{c}_1, sk_1) \wedge \mathbf{y} = \mathbf{Dec}(\mathbf{c}_2, sk_2))\big) \end{gathered} \tag{1}$$

Recall that we have assumed the secret key of the encryption scheme to be the random coins used in **Gen**. Note that the representation validity check can be efficiently performed "in the exponent" using $[\mathbf{W}]_0$ and the explicit knowledge of $\mathbf{x}$ and $\mathbf{y}$. Note also that for honestly generated keys and ciphertexts, the two checks in the expression above are equivalent (although this is not generally the case when public keys are malformed, i.e., not in the range of **Gen**).

Intuitively, the upper branch of the disjunction (1) checks consistency based on encryption randomness. This branch allows **Sam** to generate proofs without decryption keys. The lower branch of (1) uses decryption keys. This branch is used by the addition circuit **Add** to generate proofs without knowing the encryption randomness.

Relation $\mathbf{R}_2$ depends on the language TD, and on input gpk, tuple $([z]_0, \mathbf{c}_1, \mathbf{c}_2)$, and witness w_y accepts iff w_y is a valid witness to $y \in \mathsf{TD}$. (Note that $\mathbf{R}_2$ completely ignores $([z]_0, \mathbf{c}_1, \mathbf{c}_2)$.) Intuitively, $\mathbf{R}_2$ creates a simulation trapdoor (i.e., w_y) that allows to generate proofs for statements that are not in $\mathbf{R}_1$.

For $1 \leq i \leq \kappa$, the $\mathbf{Val}_i$ algorithm for Γ, on input $([z]_0, \mathbf{c}_1, \mathbf{c}_2, \pi)$, first checks that the first component is in $\mathbb{G}_0$ using $\mathbf{Val}_0$ and then checks the proof π; if both tests pass, it then returns $\top$, else $\bot$. Observe that for an honest choice of $crs = (crs', y)$, the perfect completeness and the perfect soundness of the proof system ensure that only those elements which pass relation $\mathbf{R}_1$ are accepted. Algorithm $\mathbf{Val}_{\kappa+1}$ just uses $\mathbf{Val}_0$.

The equality algorithm $\mathbf{Eq}_i$ of Γ for $1 \leq i \leq \kappa$ first checks the validity of the two group elements passed to it and then returns true iff their first components match, according to $\mathbf{Eq}_0$, the equality algorithm from the base scheme Γ_0. Algorithm $\mathbf{Eq}_{\kappa+1}$ just uses $\mathbf{Eq}_0$. The correctness of this algorithm follows from the perfect completeness of Σ.

4.3. *Group Operations*

We provide a procedure that, given as inputs $h = ([z]_0, \mathbf{c}_1, \mathbf{c}_2, \pi)$ and $h' = ([z']_0, \mathbf{c}'_1, \mathbf{c}'_2, \pi') \in \mathbb{G}_i$, generates a tuple representing the product $h \cdot h'$. This, in particular, will enable our multilinear map to be run on the additions of group elements whose explicit representations are not necessarily known. We exploit the structure of the base group as well as the homomorphic properties of the encryption scheme to "add together" the first three components. We then use (sk_1, sk_2) as a witness to generate a proof π'' that the new tuple is well formed. (For technical reasons, we check the validity of h and h' in two different ways: using proofs π, π', and also explicitly using (sk_1, sk_2). Note that, although useful in the analysis, the explicit check is redundant by the perfect soundness of the proof system under a binding crs'.)

In pp, we include an obfuscation of the C_{Add} circuit shown in Fig. 2 (top), and again, we emphasize that steps $5a$ or $5b$ are never reached with a binding crs' (but they may be reached with a hiding crs' later in the analysis). Note that although we have assumed the evaluation algorithm to be deterministic, algorithm **Prove** is randomized and we need to address how we deal with its coins. To this end, we use a **PIO** to obfuscate C_{Add}; the probabilistic obfuscator directly deals with the needed randomness.[7] The $\mathbf{Op}_i$ algorithm

[7] Typically, the obfuscated circuit will have a PRF key hardwired in and derives the required randomness by applying the PRF to the circuit inputs.

CIRCUIT $C_{\mathrm{Add}}[gpk, crs, sk_1, sk_2, td_{ext}; r](i, h, h')$:

1. if $\neg\mathbf{Val}_i(h) \vee \neg\mathbf{Val}_i(h')$ return $\perp$
2. parse $([z]_0, \mathbf{c}_1, \mathbf{c}_2, \pi) \leftarrow h$ and $([z']_0, \mathbf{c}'_1, \mathbf{c}'_2, \pi') \leftarrow h'$
3. $[z'']_0 \leftarrow [z]_0 + [z']_0$; $\mathbf{c}''_1 \leftarrow \mathbf{c}_1 + \mathbf{c}'_1$; $\mathbf{c}''_2 \leftarrow \mathbf{c}_2 + \mathbf{c}'_2$
4. // explicit validity check of h, h'
   - 4.1 $\mathbf{x} \leftarrow \mathbf{Dec}(\mathbf{c}_1, sk_1)$, $\mathbf{y} \leftarrow \mathbf{Dec}(\mathbf{c}_2, sk_2)$
     $\mathbf{x}' \leftarrow \mathbf{Dec}(\mathbf{c}'_1, sk_1)$, $\mathbf{y}' \leftarrow \mathbf{Dec}(\mathbf{c}'_2, sk_2)$
   - 4.2a if $([z]_0 \neq [\langle \mathbf{x}, \boldsymbol{\omega}_i \rangle]_0) \vee ([z]_0 \neq [\langle \mathbf{y}, \boldsymbol{\omega}_i \rangle]_0)$ goto 5a
   - 4.2b else if $([z']_0 \neq [\langle \mathbf{x}', \boldsymbol{\omega}_i \rangle]_0) \vee ([z']_0 \neq [\langle \mathbf{y}', \boldsymbol{\omega}_i \rangle]_0)$ goto 5b
   - 4.2c else goto 5c // h, h' are valid

5a. // h is invalid

- 5a.1 $w'_y \leftarrow_{\$} \mathbf{WExt}(td_{ext}, ([z]_0, \mathbf{c}_1, \mathbf{c}_2), \pi)$
- 5a.2 $\pi'' \leftarrow \mathbf{Prove}(gpk, crs, ([z'']_0, \mathbf{c}''_1, \mathbf{c}''_2), w'_y; r)$

5b.repeat 5a with h' // only h' is invalid

5c. $\pi'' \leftarrow \mathbf{Prove}(gpk, crs, ([z'']_0, \mathbf{c}''_1, \mathbf{c}''_2), (sk_1, sk_2); r)$

6. return $([z''], \mathbf{c}''_1, \mathbf{c}''_2, \pi'')$

CIRCUIT $C_{\mathrm{Map}}[gpk, crs, \mathbf{W}, sk_1](h_1, \ldots, h_\kappa)$:

1. for $i = 1 \ldots \kappa$
   - 1.1 if $\neg\mathbf{Val}_i(h_i)$ return $\perp$
   - 1.2 $([z_i]_0, \mathbf{c}_{i,1}, \mathbf{c}_{i,2}, \pi_i) \leftarrow h_i$
   - 1.3 $\mathbf{x}_i \leftarrow \mathbf{Dec}(\mathbf{c}_{i,1}, sk_1)$

2. $z_{\kappa+1} \leftarrow \prod_{i=1}^{k} \langle \mathbf{x}_i, \boldsymbol{\omega}_i \rangle \pmod N$
3. return $[z_{\kappa+1}]_{\kappa+1}$

Fig. 2. *Top*: Circuit for addition of group elements. Explicit randomness r is internally generated when using a **PIO**. *Bottom*: Circuit implementing the multilinear map. Recall that here $gpk = (pp_0, pk_1, pk_2, [\mathbf{W}]_0, \mathsf{TD}, y)$.

for $1 \leq i \leq \kappa$ runs the obfuscated circuit on i, the input group elements. Algorithm $\mathbf{Op}_{\kappa+1}$ just uses $\mathbf{Op}_0$ as usual. The correctness of this algorithm follows from those of Γ_0 and Π, the completeness of Σ, and the correctness, in our sense, of the probabilistic obfuscator $\mathbf{Obf} = \mathbf{PIO}$; see Sect. 2 for the definitions.

4.4. *The Multilinear Map*

The multilinear map for Γ, on input κ group elements $h_i = [z_i]_i = ([z_i]_0, \mathbf{c}_{i,1}, \mathbf{c}_{i,2}, \pi_i)$, uses sk_1 to recover the representation $\mathbf{x}_i$. It then uses the explicit knowledge of the matrix $\mathbf{W}$ to compute the output of the map as

$$\mathbf{e}([z_1]_1, \ldots, [z_\kappa]_\kappa) := \left[\prod_{i=1}^{k} \langle \mathbf{x}_i, \boldsymbol{\omega}_i \rangle \right]_{\kappa+1}.$$

Recalling that $\mathbb{G}_{\kappa+1}$ is nothing other than $\mathbb{G}_0$, and $g_{\kappa+1} = g_0$, the output of the map is just the $\mathbb{G}_0$-element $(g_0)^{\prod_{i=1}^{k} \langle \mathbf{x}_i, \boldsymbol{\omega}_i \rangle}$. The product in the exponent can be efficiently computed over $\mathbb{Z}_N$ for *any* polynomial level of linearity κ and any ℓ as it uses $\mathbf{x}_i$ and $\boldsymbol{\omega}_i$ explicitly. The multilinearity of the map follows from the linearity of each of the

multiplicands in the above product (and the completeness of Σ, the correctness of Π, and the correctness of the (possibly probabilistic) obfuscator **Obf**). An obfuscation $\overline{C}_{\text{Map}}$ of the circuit implementing this operation (see Fig. 2, bottom) will be made available through the public parameters, and **e** is defined to run this circuit on its inputs.

4.5. *Sampling and Extraction*

For sampling random group elements, we first define two vectors $\mathbf{x}$ and $\mathbf{y}$ in $\mathbb{Z}_N^\ell$ satisfying $\langle \mathbf{x}, \boldsymbol{\omega}_i \rangle = \langle \mathbf{y}, \boldsymbol{\omega}_i \rangle$. In other words, these vectors define two equivalent representations of a group element relative to a matrix $\mathbf{W}$. If $\mathbf{W}$ is explicitly known, the vectors $\mathbf{x}$ and $\mathbf{y}$ can take arbitrary forms subject to validity. However, is only implicitly known by an honest user of the system, and in order to sample random group elements, we set $\mathbf{x} = \mathbf{y} = (z, 0)$ for $\ell = 2$. (We call these the canonical representations.)

Then, we set $[z]_0 := [\langle \mathbf{y}, \boldsymbol{\omega}_i \rangle]_0$ (which can be computed using $[\mathbf{W}]_0$ and explicit knowledge of $\mathbf{x}$) and

$$[z]_i \leftarrow \big([z]_0, \mathbf{c}_1 = \mathbf{Enc}(\mathbf{x}, pk_1; \mathbf{r}_1), \mathbf{c}_2 = \mathbf{Enc}(\mathbf{y}, pk_2; \mathbf{r}_2), \\ \pi = \mathbf{Prove}(gpk, crs, ([z]_i, \mathbf{c}_1, \mathbf{c}_2), (\mathbf{x}, \mathbf{y}, \mathbf{r}_1, \mathbf{r}_2))\big) .$$

Note that the outputs of the sampler are *not* statistically uniform within $\mathbb{G}_i([z]_i)$. Indeed, not even the IND-CPA security of the encryption directly implies any form of security of the generated ciphertexts (since the addition circuit **Add** contains the corresponding decryption keys). Our upcoming "switching theorem" (Theorem 1) will, however, prove that encodings that are functionally equivalent cannot be efficiently distinguished.

Since the target group has unique encodings, as noted in Sect. 3, an extraction algorithm for all groups can be derived from one for the target group. The latter can be implemented by applying a universal hash function to the group elements in $\mathbb{G}_T$, for example.

5. Indistinguishability of Encodings

In this section, we will prove a theorem that is an essential tool in establishing the intractability of the κ-MDDH for our MLG scheme Γ constructed in Sect. 4. This theorem, roughly speaking, states that valid encodings of elements within a single equivalence class are computationally indistinguishable. We formalize this property via the κ-Switch game shown in Fig. 3. This game lets an adversary $\mathcal{A}$ choose an element $[z]_i \in \mathbb{G}_i$ by producing two valid representations $(\mathbf{x}_0, \mathbf{y}_0)$ and $(\mathbf{x}_1, \mathbf{y}_1)$ for it. The adversary is given an encoding of $[z]_i$ generated using $(\mathbf{x}_b, \mathbf{y}_b)$ for a random b and has to guess the bit b. In this game, besides access to pp, which contains the obfuscated circuits for the group operation and the multilinear map, we also provide the matrix $\mathbf{W}$ in the clear to the adversary. This strengthens the κ-Switch game and is needed for our later analysis.

To prove that the advantage of $\mathcal{A}$ in the κ-Switch game is negligible, we rely on the security of the obfuscator, the IND-CPA security of the encryption scheme, and the security of the NIZK proof system.

κ-Switch$^{\mathcal{A}}_{\Gamma}(\lambda)$:
$pp \leftarrow\!\!\$\ \mathbf{Setup}(1^\lambda, 1^\kappa)$
$((\mathbf{x}_0, \mathbf{y}_0), (\mathbf{x}_1, \mathbf{y}_1), i, st) \leftarrow\!\!\$\ \mathcal{A}_1(pp, \mathbf{W})$
$b \leftarrow\!\!\$\ \{0,1\};\ \mathbf{r}_1, \mathbf{r}_2 \leftarrow\!\!\$\ \{0,1\}^{r(\lambda)}$
$\mathbf{c}_1 \leftarrow \mathbf{Enc}(\mathbf{x}_b, pk_1; \mathbf{r}_1);\ \mathbf{c}_2 \leftarrow \mathbf{Enc}(\mathbf{y}_b, pk_2; \mathbf{r}_2)$
$\pi \leftarrow\!\!\$\ \mathbf{Prove}(gpk, crs, ([z]_0, \mathbf{c}_1, \mathbf{c}_2), (\mathbf{x}_b, \mathbf{y}_b, \mathbf{r}_1, \mathbf{r}_2, \perp, \perp))$
$b' \leftarrow\!\!\$\ \mathcal{A}_2\big(([\langle \mathbf{x}_b, \boldsymbol{\omega}_i\rangle]_0, \mathbf{c}_1, \mathbf{c}_2, \pi), st\big)$
Return $(b = b')$

Fig. 3. Game formalizing the indistinguishability of encodings with an equivalence class. This game is specific to our construction Γ. An adversary is legitimate if $z = \langle \mathbf{x}_b, \boldsymbol{\omega}_i\rangle = \langle \mathbf{y}_b, \boldsymbol{\omega}_i\rangle$ for $b \in \{0,1\}$. We note that $\mathcal{A}$ gets explicit access to matrix $\mathbf{W}$ generated during setup .

Intuitively, the IND-CPA security of the encryption scheme will ensure that the encryptions of the two representations are indistinguishable. This argument, however, does not immediately work as the parameters pp contain component $\overline{C}_{\text{Add}}$ that depends on *both* decryption keys. We deal with this by finding an alternative implementation of this circuit without the knowledge of the secret keys, in the presence of a slightly different public parameters (which are computationally indistinguishable to those described in Sect. 4). The next lemma, roughly speaking, says that *provided* parameters ppinclude an instance $y \in \mathsf{TD}$; then, there exists an alternative implementation $\widehat{C}_{\text{Add}}$ that does not use the secret keys, and whose obfuscation is indistinguishable to that of $\widetilde{C}_{\text{Add}}$ of Fig. 2 (top) for an adversary that *knows* the secret keys. It relies on the security of the obfuscator and the security of the NIZK proof system.

Lemma 1. (C_{Add} without decryption keys) *Let **PIO** be a secure obfuscator for X-IND samplers and Σ be a dual-mode NIZK proof system. Additionally, let parameters $\widetilde{pp}$ be sampled as in Sect. 4 but with $\widetilde{y} \in \mathsf{TD}$. Furthermore, let $\widehat{pp}$ be sampled as $\widetilde{pp}$, but with a hiding CRS $\widehat{crs}'$, and an obfuscation of circuit $\widehat{C}_{Add}$ of Fig. 4 (bottom). Then, for any* PPT *adversary $\mathcal{A}$, there are ppt adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ of essentially the same complexity as $\mathcal{A}$ such that for all $\lambda \in \mathbb{N}$*

$$\begin{aligned} &Pr\left[\mathcal{A}(\widetilde{pp}, sk_1, sk_2) = 1 \ :\ (sk_1, sk_2) \leftarrow\!\!\$\ \boldsymbol{Gen}(1^\lambda)\right] \\ &\quad - Pr\left[\mathcal{A}(\widehat{pp}, sk_1, sk_2) = 1 \ :\ (sk_1, sk_2) \leftarrow\!\!\$\ \boldsymbol{Gen}(1^\lambda)\right] \\ &\quad \le 2 \cdot \boldsymbol{Adv}^{ind}_{\boldsymbol{PIO}, \mathcal{B}_1}(\lambda) + \boldsymbol{Adv}^{crs}_{\Sigma, \mathcal{B}_2}(\lambda). \end{aligned}$$

Proof. The crucial observation is that a witness w_y to $\widetilde{y} \in \mathsf{TD}$ is also a witness to $x \in \mathbf{R}$, and therefore, $\widehat{C}_{\text{Add}}$ can use w_y instead of sk_1, sk_2 to produce the output proof π''. Below we provide descriptions of the transformation from C_{Add} to $\widehat{C}_{\text{Add}}$, and let W_i denote the event that $\mathcal{A}$ in Game_i outputs 1.

Game_0: We start with (a PIO obfuscation of) circuit C_{Add} of Fig. 2 and with $\widetilde{pp}$ including $\widetilde{y} \in \mathsf{TD}$ and a binding crs'.

Game_1: The circuit has witness w_y to $\widetilde{y} \in \mathsf{TD}$ hard-coded. If some input reaches the "invalid" branches, C_{Add} does not extract a witness from the corresponding

proof, but instead uses w_y to generate proof π''. [See Fig. 4 (top).] Note that Game_1 requires no extraction trapdoor td_{ext} anymore.
We claim that $|Pr[\text{W}_0(\lambda)] - Pr[\text{W}_1(\lambda)]| \leq \mathbf{Adv}^{\text{ind}}_{\mathbf{PIO},\mathcal{B}_1}(\lambda)$.
By construction, the only difference between the games is that in Game_1, proof π'', with respect to invalid (input) encodings, is generated using hard-coded witness w_y to $\widetilde{y} \in \mathsf{TD}$. Since w_y is unique, and the CRS crs' guarantees perfect soundness, this leads to *identical* behavior of C_{Add} in Game 0. Hence, this hop is justified by PIO.

Game_2: The CRS $\widehat{crs}'$ included in the public parameters is now hiding (such that the generated proofs are perfectly witness-indistinguishable). We have that

$$|Pr[\text{W}_1(\lambda)] - Pr[\text{W}_2(\lambda)]| \leq \mathbf{Adv}^{\text{crs}}_{\Sigma,\mathcal{B}_2}(\lambda),$$

where $\mathcal{B}_2$ is a PPT algorithm against the indistinguishability of binding and hiding CRS's.

Game_3: ere, output proofs π'' for those inputs entering the "valid" branch (step 5b of Fig. 4 (top)) use w_y (and not sk_1, sk_2) as witness. In particular, this game does not need to perform a explicit validity check (using sk_1, sk_2) anymore, and therefore, the addition circuit can be described as in Fig. 4 (bottom).
We claim that $|Pr[\text{W}_2(\lambda)] - Pr[\text{W}_3(\lambda)]| \leq \mathbf{Adv}^{\text{ind}}_{\mathbf{PIO},\mathcal{B}_1}(\lambda)$.
By construction, the only difference between both games is that the public parameters in Game_2 contain a PIO obfuscation of C_{Add} and in Game_3 contain a PIO obfuscation of $\widehat{C}_{\text{Add}}$ of Fig. 4. In Lemma 2, we prove that these circuit variants are given by an X-IND sampler, and therefore, their PIO obfuscations are indistinguishable. □

Lemma 2. (X-IND sampling) *Let Σ be a dual-mode NIZK proof system for the relation $(\boldsymbol{S}, \boldsymbol{R})$ defined in Sect. 4.2. Suppose Σ is perfectly witness-indistinguishable under a hiding CRS. Let $\mathcal{A}$ be a sampler which outputs circuits $(\widetilde{C}_{Add}, \widehat{C}_{Add})$ of Fig. 4. (Both circuits have the system parameters hard-coded in.) Then, $\mathcal{A}$ is X-IND for (the optimal) X, the size of the domain of the circuits. More precisely, for any (possibly unbounded) distinguisher $\mathcal{D}'$ and for any* PPT *distinguisher $\mathcal{D} = (\mathcal{D}_1, \mathcal{D}_2)$ and any $\lambda \in \mathbb{N}$,*

$$\boldsymbol{Adv}^{\text{eq\$}}_{\mathcal{A},\mathcal{D}'}(\lambda) = 0 \quad \textit{and} \quad \boldsymbol{Adv}^{\textit{sel-ind}}_{\mathcal{A},\mathcal{D}}(\lambda) = 0\,.$$

Proof. The first equality is immediate as $\mathcal{X}$ is set to be the entire domain of the circuits. The second equality follows from the perfect witness-indistinguishability property of the proof system. Indeed, the only difference between the two circuits is that, for those inputs that are valid encodings, $\widetilde{C}_{\text{Add}}$ uses decryption keys sk_1, sk_2 as witness to generate the output proof $\pi'' \leftarrow \mathbf{Prove}(gpk, crs, ([z'']_0, \mathbf{c}''_1, \mathbf{c}''_2), (sk_1, sk_2); r)$, and $\widehat{C}_{\text{Add}}$ uses witness w_y to $\widetilde{y} \in \mathsf{TD}$ (with $\widetilde{y}$ in the public parameters) to generate the proof $\widehat{\pi}'' \leftarrow \mathbf{Prove}(gpk, crs, ([z'']_0, \mathbf{c}''_1, \mathbf{c}''_2), w_y; r)$. The WI property with a hiding $\widehat{crs}'$ guarantees that π'' and $\widehat{\pi}''$ are *identically* distributed and hence so are the outputs of $\widetilde{C}_{\text{Add}}$ and $\widehat{C}_{\text{Add}}$. Note that no random coins are hardwired into these circuits—we are in the PIO setting—and fresh coins are used to compute the circuits' outputs.

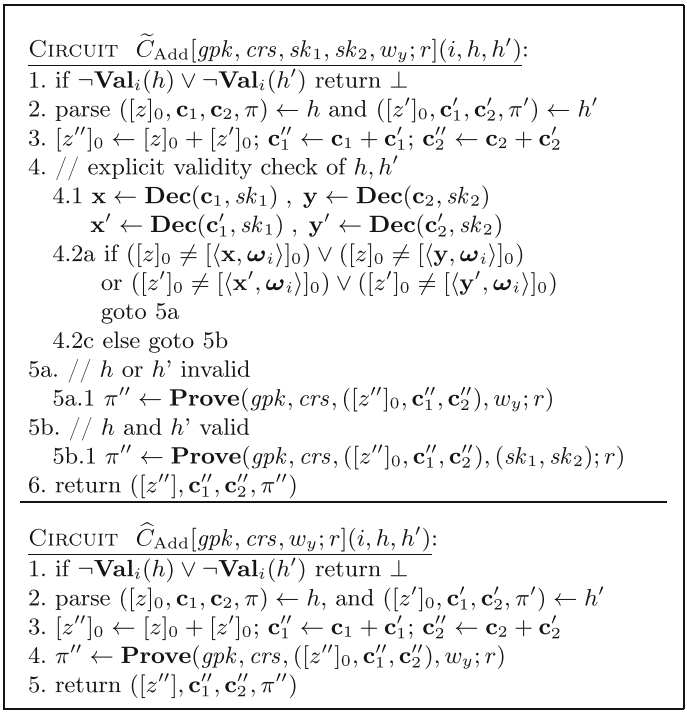

Fig. 4. Circuits for addition of group elements used in Lemma 1. $\widehat{pp}$ includes $gpk = (pp_0, pk_1, pk_2, [\mathbf{W}]_0, \mathsf{TD}, \widetilde{y})$ where $\widetilde{y} \in \mathsf{TD}$ (also includes a hiding CRS $\widehat{crs}'$). Both circuits also have hard-coded (the) witness w_y to $\widetilde{y} \in \mathsf{TD}$. **Top:** sk_1, sk_2 are used to produce π'' on valid inputs. **Bottom:** w_y is always used to produce π''.

With Lemma 1, we can invoke IND-CPA security and via a sequence of games obtain the result stated below. The proof can be found in "Appendix A.1"; we will give a high-level overview of the proof below. (See also Fig. 5.)

Theorem 1. (Switching encodings using PIO) *Let Γ be the MLG scheme constructed in Sect. 4, where **PIO** is secure for X-IND samplers, Π is an IND-CPA-secure encryption scheme, and Σ is a dual-mode NIZK proof system. Then, encodings of equivalent group elements are indistinguishable. More precisely, for any* PPT *adversary $\mathcal{A}$ and all $\lambda \in \mathbb{N}$, there are ppt adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$, and $\mathcal{B}_4$ of essentially the same complexity as $\mathcal{A}$ such that for all $\lambda \in \mathbb{N}$*

$$\boldsymbol{Adv}^{\kappa\text{-}switch}_{\Gamma,\mathcal{A}}(\lambda) \leq 3 \cdot \boldsymbol{Adv}^{sm}_{\mathsf{TD},\mathcal{B}_1} + 7 \cdot \boldsymbol{Adv}^{ind}_{\boldsymbol{PIO},\mathcal{B}_2}(\lambda) + 3 \cdot \boldsymbol{Adv}^{crs}_{\Sigma,\mathcal{B}_3}(\lambda) + 2 \cdot \boldsymbol{Adv}^{ind\text{-}cpa}_{\Pi,\mathcal{B}_4}(\lambda) \ .$$

Furthermore, $\mathcal{B}_2$ is an X-IND sampler for any function $X(\lambda)$.

Proof sketch. The proof of this theorem proceeds via a sequence of 9 games as follows.

G.	public parameters	C_{Add} knows	C_{Map} knows	$\mathbf{c}_1$ $(b=0)$ contains	$\mathbf{c}_2$ $(b=0)$ contains	**remark**
0	pp	sk_1, sk_2, td_{ext}	sk_1	$\mathbf{x}_0$	$\mathbf{y}_0$	
1	$\boxed{\widetilde{pp}}$	sk_1, sk_2, td_{ext}	sk_1	$\mathbf{x}_0$	$\mathbf{y}_0$	TD indist.
2	$\boxed{\widehat{pp}}$	$\boxed{w_y}$	sk_1	$\mathbf{x}_0$	$\mathbf{y}_0$	Lemma 1
3	$\widehat{pp}$	w_y	sk_1	$\mathbf{x}_0$	$\boxed{\mathbf{y}_1}$	IND-CPA
4	$\boxed{\widetilde{pp}}$	$\boxed{sk_1, sk_2, td_{ext}}$	sk_1	$\mathbf{x}_0$	$\mathbf{y}_1$	Lemma 1
5	$\boxed{pp}$	sk_1, sk_2, td_{ext}	sk_1	$\mathbf{x}_0$	$\mathbf{y}_1$	TD indist.
6	pp	sk_1, sk_2, td_{ext}	$\boxed{sk_2}$	$\mathbf{x}_0$	$\mathbf{y}_1$	PIO
7	$\boxed{\widetilde{pp}}$	sk_1, sk_2, td_{ext}	sk_2	$\mathbf{x}_0$	$\mathbf{y}_1$	TD indist.
8	$\boxed{\widehat{pp}}$	$\boxed{w_y}$	sk_2	$\mathbf{x}_0$	$\mathbf{y}_1$	Lemma 1
9	$\widehat{pp}$	w_y	sk_2	$\boxed{\mathbf{x}_1}$	$\mathbf{y}_1$	IND-CPA

Fig. 5. Outline of the proof steps of Theorem 1. b is the random bit of the κ-Switch game. (See Fig. 3.) Changing between pp and $\widetilde{pp}$ is justified by the hardness of deciding membership of TD, and changing between $\widetilde{pp}$ and $\widehat{pp}$ by Lemma 1. The hops relying on PIO use the perfect correctness of Π, and the perfect completeness and the perfect soundness of Σ under binding crs' to argue function equivalence of C_{Map}.

Game_0: This is the κ-Switch game. The public parameters pp contain a no-instance $y \notin \mathsf{TD}$, a binding crs', and C_{Add} is constructed using (sk_1, sk_2) and C_{Map} using sk_1. (See Fig. 2.) The ciphertexts $\mathbf{c}_1$ and $\mathbf{c}_2$ contain $\mathbf{x}_b$ and $\mathbf{y}_b$ for a random bit b.

Game_1: This game generates the public parameters $\widetilde{pp}$ so that they include a yes-instance $y \in \mathsf{TD}$. The difference to the previous game can be bounded via the hardness of deciding membership in TD.

Game_2: The public parameters $\widehat{pp}$ change so that they include a hiding $\widehat{crs}'$, and a (PIO) obfuscation of circuit $\widehat{C}_{\text{Add}}$, see Fig. 4. (Recall that this circuit uses the witness w_y to $y \in \mathsf{TD}$ to produce the output proofs $\widetilde{\pi}''$, and therefore, the *simultaneous* knowledge of decryption keys sk_1,sk_2 is not needed anymore.) Additionally, the game uses w_y to prepare the proof π in the κ-Switch challenge for $\mathcal{A}$. By Lemma 1 and the perfect witness-indistinguishability of Σ, the difference with the previous game can be bounded by PIO and CRS indistinguishability.

Game_3: This game generates $\mathbf{c}_2$ by encrypting $\mathbf{y}_1$, even when $b = 0$. We can bound the difference in any adversary's success probability via the IND-CPA advantage of Π with respect to pk_2. (The reduction will know (pk_1, sk_1) so as to be able to construct C_{Map}.)

Game_4: The public parameters are changed back to $\widetilde{pp}$, so that they include a binding crs', and a (PIO) obfuscation of circuit $\widetilde{C}_{\text{Add}}$ of Fig. 2 (top). The difference with the previous game is bounded again with Lemma 1.

Game_5: Now, a no-instance $y \notin \mathsf{TD}$ is included in the public parameters pp. This game is justified by the hardness of deciding membership in TD.

Game_6: This game uses sk_2 (in place of sk_1) in the generation of C_{Map} circuit. In this transition, we rely on the security of **Obf**, the perfect correctness of Π, and the perfect soundness of Σ. Perfect soundness of Σ implies that C_{Map} rejects

ciphertexts unless relation $\mathbf{R}_1$ holds. Together with the perfect correctness of Π, $\mathbf{R}_1$ implies that C_{Map} yields identical results with sk_1 and sk_2. We can then use the IO security of **Obf** to justify the switch from using sk_1 to using sk_2. (Note that for any function X, any obfuscator that is secure for X-IND samplers is also secure as an indistinguishability obfuscator.) Note that in this game, it is crucial that the crs' is in the binding mode.

Game$_7$: This game, similar to Game$_1$, switches to public parameters $\widetilde{pp}$ with a yes-instance $y \in \mathsf{TD}$. The analysis is as before.

Game$_8$: This game, similar to Game$_2$, includes in $\widehat{pp}$ a hiding $\widehat{crs}'$, and a (PIO) obfuscation of circuit $\widehat{C}_{\text{Add}}$. (See Fig. 4.) The analysis is as before.

Game$_9$: This game generates $\mathbf{c}_1$ by encrypting $\mathbf{x}_1$, even when $b = 0$. The analysis is as in Game$_3$.

Observe that the challenge encoding in Game$_9$ is independent of the random bit b and the advantage of any (possibly unbounded) adversary $\mathcal{A}$ is 0. Collecting bounds on the probabilities involved in the various game hops concludes the proof.

□

6. The Multilinear DDH Problem

In this section, we show that natural multilinear analogues of the decisional Diffie–Hellman (DDH) problem are hard for our MLG scheme Γ from Sect. 4. We will establish this for two specific **Setup** algorithms which give rise to symmetric and asymmetric multilinear maps in groups of prime order N. (See Sect. 3 for the formal definition.) In the symmetric case, we will base hardness on the q-strong DDH problem [4] and in the asymmetric case on the 1-strong DDH problem.

6.1. *Intractable Problems*

We start by formalizing the hard problems that we will be relying on and those whose hardness we will be proving. We do this in a uniform way using the language of group schemes of Sect. 3. Informally, the q-SDDH problem requires the indistinguishability of $g^{x^{q+1}}$ from a random element given $(g^x, g^{x^2}, \ldots, g^{x^q})$ for a random x, and the κ-MDDH problem, whose hardness we will be establishing, generalizes the standard bilinear DDH problem (and its variants) and requires this for $g_T^{a_1 \cdots a_{\kappa+1}}$ in the presence of $(g^{a_1}, \ldots, g^{a_{\kappa+1}})$ (for uniformly random a_i).

The q-SDDH problem. For $q \in \mathbb{N}$, we say that a group scheme Γ_0 is q-SDDH intractable if

$$\mathbf{Adv}^{q\text{-sddh}}_{\Gamma_0,\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[q\text{-SDDH}^{\mathcal{A}}_{\Gamma_0}(\lambda)\right] - 1 \in \text{Negl}\,,$$

where game q-SDDH$^{\mathcal{A}}_{\Gamma_0}(\lambda)$ is shown in Fig. 6 (left).

The (κ, I)-MDDH problem. We use a slight reformulation of the (generalized) MDDH problem from [23]. For $\kappa \in \mathbb{N}$ we say that an MLG scheme Γ is κ-MDDH intractable with respect to the index set I if

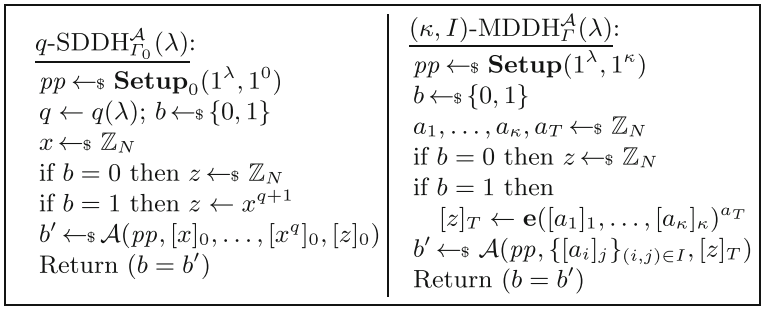

Fig. 6. *Left*: The strong DDH problem. *Right*: The multilinear DDH problem, where I specifies the available group elements. By slight abuse of notation, repeated use of $[a_i]_i$ denotes the same sample. Recall that we use the notation $[z]_T$ and $[z]_{\kappa+1}$ for elements of the target group $\mathbb{G}_{\kappa+1}$ interchangeably.

$$\mathbf{Adv}^{(\kappa,I)\text{-mddh}}_{\Gamma,\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[(\kappa, I)\text{-MDDH}^{\mathcal{A}}_{\Gamma}(\lambda)\right] - 1 \in \text{NEGL}\ ,$$

where game (κ, I)-MDDH$^{\mathcal{A}}_{\Gamma}(\lambda)$ is shown in Fig. 6 (right). Here, I is a set of ordered pairs of integers (i, j) with $1 \le i \le \kappa + 1$, $1 \le j \le \kappa$. The adversary is provided with challenge group elements $[a_i]_j$ for $(i, j) \in I$, so that its challenge elements may lie in any combination of the groups. The following example of such a set I leads to a generalization of the symmetric external Diffie–Hellman (SXDH) assumption to the multilinear case:

$$I = I^* := \{(1,1), \ldots, (\kappa,\kappa), (\kappa+1,\kappa)\}\ .$$

Of course, when generalizing SXDH, the choice of the last element of I is not canonical. Instead of $(\kappa + 1, \kappa)$, also other values $(\kappa + 1, j)$ for $j \in \{1, \ldots, \kappa\}$ seem natural.

6.2. *The Symmetric Setting*

We describe a special variant of our general construction in Sect. 4 which gives rise to a *symmetric* MLG scheme as defined in Sect. 3.

We set $\ell := 2$ and sample $\mathbf{W} = (\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_\kappa)^t$ by setting $\boldsymbol{\omega}_i = (1, \omega)$ for a random $\omega \in \mathbb{Z}_N$. The generators and identity elements for all groups are set to be a single value generated for the first group. These modifications ensure that the scheme algorithms are independent of the index for $1 \le i \le \kappa$ and that $\mathbf{e}$ is invariant under all permutations of its inputs.

The following lemma, which provides a mechanism to compute polynomial values "in the exponent," will be helpful in the security analysis of our constructions.

Lemma 3. (Horner in the exponent) *Let $\boldsymbol{\omega} = (\omega_0, \omega_1) \in \mathbb{Z}_N^2$ and $\boldsymbol{x}_i = (x_{i,0}, x_{i,1}) \in \mathbb{Z}_N^2$ for $i = 1, \ldots, \kappa$. Define $z_i := \langle \boldsymbol{x}_i, \boldsymbol{\omega} \rangle$. Then, given only the implicit values $[\omega_0^j \omega_1^k]_T$, for all j, k such that $j + k = \kappa$ and the explicit values $\boldsymbol{x}_i$, the element $[z_1 \cdots z_\kappa]_T$ can be efficiently computed.*

Proof. Let

$$P(\omega_0, \omega_1) := \prod_{i=1}^{\kappa} (x_{i,0} \cdot \omega_0 + x_{i,1} \cdot \omega_1) = \sum_{j+k=\kappa} p_{jk} \cdot \omega_0^j \omega_1^k \ .$$

Clearly, if all p_{jk} are known, then $[P(\omega_0, \omega_1)]_T$ can be computed using $[\omega_0^j \omega_1^k]_T$ with polynomially many operations. (There are $\mathcal{O}(\kappa)$ summands above.) To obtain these values, we apply Horner's rule. Define

$$P_i(\omega_0, \omega_1) := \begin{cases} 1 & \text{if } i = 0 \ ; \\ (x_{i,0} \cdot \omega_0 + x_{i,1} \cdot \omega_1) \cdot P_{i-1}(\omega_0, \omega_1) & \text{otherwise.} \end{cases}$$

The coefficients of P_κ are the required p_{jk} values. Let t_i denote the number of terms in P_i. It takes at most $2t_i$ multiplications and $t_i - 1$ additions in $\mathbb{Z}_N$ to compute the coefficients of P_i from P_{i-1} and $\mathbf{x}_i$. Since $t_i \in \mathcal{O}(\kappa)$, at most $\mathcal{O}(\kappa^2)$ many operations in total are performed. We note that the lemma generalizes to any (constant) ℓ with computational complexity $\mathcal{O}(\kappa^\ell)$. □

We prove the following result formally in "Appendix A.2" and give an overview of the proof here.

Theorem 2. *(κ-SDDH hard $\implies$ symmetric (κ, I^*)-MDDH hard) Write $I = I^* = \{(i, 1) \mid i \in [\kappa + 1]\}$ for the index set with all the second components being 1. Let Γ^* denote scheme Γ of Sect. 4 constructed using base group Γ_0 and a probabilistic indistinguishability obfuscator **PIO** with modifications as described above, and let $\kappa \in \mathbb{N}$. Then, for any* PPT *adversary $\mathcal{A}$, there are* PPT *adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, and $\mathcal{B}_3$ of essentially the same complexity as $\mathcal{A}$ such that for all $\lambda \in \mathbb{N}$*

$$\boldsymbol{Adv}^{(\kappa,I^*)\text{-}mddh}_{\Gamma^*,\mathcal{A}}(\lambda) \leq \boldsymbol{Adv}^{\kappa\text{-}sddh}_{\Gamma_0,\mathcal{B}_1}(\lambda) + \boldsymbol{Adv}^{ind}_{\boldsymbol{PIO},\mathcal{B}_2}(\lambda) + (\kappa + 1) \cdot \boldsymbol{Adv}^{\kappa\text{-}switch}_{\Gamma^*,\mathcal{B}_3}(\lambda) \ .$$

Proof. In our reduction, the value ω used to generate $\mathbf{W}$ will play the role of the implicit value in the SDDH problem instance. We therefore change the implementation of C_{Map} to one that *does not know* ω in the clear and only uses the implicit values $[\omega^i]_0$. (Recall that in our construction $\mathbb{G}_T$ is just $\mathbb{G}_0$, so these elements come from the SDDH instance.) Such a circuit C^*_{Map} can be efficiently implemented using Horner's rule above. In more detail, C^*_{Map} has $[\omega^i]_T$ hard-coded in, recovers $\mathbf{x}_i$ from its inputs using sk_1, and then applies Lemma 3 with $(\omega_0, \omega_1) := (1, \omega)$ to evaluate the multilinear map.

The proof proceeds along a sequence of $\kappa + 4$ games as follows.

Game_0:	This is the κ-MDDH problem (Fig. 6, right). We use $\mathbf{x}_i$ and $\mathbf{y}_i$ to denote the representation vectors of a_i generated within the sampler $\mathbf{Sam}_{I(i)}(a_i)$, where $(i, I(i)) \in I$.
Game_1–$\text{Game}_{\kappa+1}$:	In these games, we gradually switch the representations of $[a_i]_1$ for $i \in [\kappa + 1]$ so that they are of the form $(a_i - \omega, 1)$. Each hop can be bounded via the Switch game.

$\text{Game}_{\kappa+2}$: This game introduces a conceptual change: the a_i for $i \in [\kappa+1]$ are generated as $a_i + \omega$. Note that the distributions of these values are still uniform and that the exponent of the MDDH challenge when $b = 1$ is

$$\prod_{i=1}^{\kappa+1} (a_i + \omega) \ .$$

This game prepares us for embedding a κ-SDDH challenge and then to randomize the exponent above.

$\text{Game}_{\kappa+3}$: This game switches C_{Map} to C^*_{Map} as defined above. We use indistinguishability obfuscation and the fact that these circuits are functionally equivalent to bound this hop. We are now in a setting where ω is only implicitly known.

$\text{Game}_{\kappa+4}$: This game replaces MDDH challenge $[\omega^{\kappa+1}]_0$ with a random value $[\sigma]_0$ in case $b = 1$. (Hence, the MDDH challenge is independently uniform regardless of b.) Observe that $\text{Game}_{\kappa+3}$ and $\text{Game}_{\kappa+4}$ only require $[\omega^i]_0$ (for $i \leq \kappa+1$) and in fact require $[\omega^{\kappa+1}]_0$ only for the MDDH challenge. Hence, we can bound this hop using the κ-SDDH assumption.

In $\text{Game}_{\kappa+4}$, irrespective of the value of $b \in \{0,1\}$, the challenge is uniformly and independently distributed as σ remains outside the view of the adversary. Hence, the advantage of any (unbounded) adversary in this game is 0. This concludes the sketch proof. □

We note that in this symmetric case, C^*_{Map} can be directly used as the implementation of the multilinear map. We chose C_{Map} because it is somewhat simpler and also more in line with the upcoming asymmetric case.

6.3. *The Asymmetric Setting*

We describe a second variant of the construction in Sect. 4 that results in an asymmetric MLG scheme. We set $\ell := 2$ and choose the matrix $\mathbf{W} = (\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_\kappa)^t$ by setting $\boldsymbol{\omega}_i := (1, \omega_i)$ for random $\omega_i \in \mathbb{Z}_N$.

The following theorem shows that for index set $I = \{(i, I(i)) : 1 \leq i \leq \kappa+1\}$ given by an arbitrary function $I : [\kappa+1] \longrightarrow [\kappa]$, this construction is (κ, I)-MDDH intractable under the 1-SDDH assumption in the base group, the security of the obfuscator, and the κ-Switch game in Sect. 5. We present the proof intuition here and leave the details to "Appendix A.3."

Theorem 3. *(1-SDDH hard $\Longrightarrow$ asymmetric (κ, I)-MDDH hard) Let Γ^* denote scheme Γ of Sect. 4 constructed using base group Γ_0 and a probabilistic indistinguishability obfuscator **PIO** with modifications as described above, and let $\kappa \in \mathbb{N}$. Then, for any ppt adversary $\mathcal{A}$, there are* PPT *adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, and $\mathcal{B}_3$ such that for all λ*

$$\boldsymbol{Adv}^{(\kappa,I)\text{-}mddh}_{\Gamma^*,\mathcal{A}}(\lambda) \leq \boldsymbol{Adv}^{1\text{-}sddh}_{\Gamma_0,\mathcal{B}_1}(\lambda) + \boldsymbol{Adv}^{ind}_{\boldsymbol{PIO},\mathcal{B}_2}(\lambda) + 2 \cdot \boldsymbol{Adv}^{\kappa\text{-}switch}_{\Gamma^*,\mathcal{B}_3}(\lambda) + \frac{\kappa - 1}{N(\lambda)}.$$

Proof. The general proof strategy is similar to that of the symmetric case and proceeds along a sequence of 5 games as follows.

Game$_0$: This is the (κ, I)-MDDH problem. By the pigeon-hole principle, there must exist a pair of distinct $i, i' \in [\kappa + 1]$ such that $I(i) = I(i') \in [\kappa]$. Without loss of generality, we assume that $I(1) = I(2) = 1$.

Game$_1$–Game$_2$: In these games, we gradually switch the representation vectors of $[a_i]_1$ for $i = 1, 2$ to those of the form $(a_i - \omega_1, 1)$. Each of these hops can be bounded via the Switch game.

Game$_3$: This game introduces a conceptual change and generates a_i as $a_i + \omega_1$ for $i = 1, 2$. The exponent of the MDDH challenge when $b = 1$ is

$$(a_1 + \omega_1)(a_2 + \omega_1) \cdot \prod_{j=3}^{\kappa+1} a_j \ .$$

Game$_4$: In this game, we change the implementation of C_{Map} to one which uses all but one of the ω_i explicitly, and the remaining one implicitly via $[\omega_1]_0$. The new circuit C^*_{Map} is functionally equivalent to the original circuit used in the scheme. We invoke the IO security of the obfuscator to conclude the hop. This game prepares us to embed a 1-SDDH challenge next.

Game$_5$: This game replaces MDDH challenge $[{\omega_1}^2]_0$ with a random value $[\sigma]_0$ in case $b = 1$. Observe that Game$_4$ and Game$_5$ only require $[\omega_1]_0$ and $[{\omega_1}^2]_0$ and in fact require $[{\omega_1}^2]_0$ only for the MDDH challenge. Hence, we can bound the distinguishing advantage in this hop down to the 1-SDDH game.

In Game$_5$, irrespective of the value of $b \in \{0, 1\}$, the challenge is uniformly and independently distributed as σ remains outside the view of the adversary. Hence, the advantage of any (possibly unbounded) adversary in this game is 0. □

Acknowledgements

Albrecht, Larraia, and Paterson were supported by EPSRC grant EP/L018543/1. Farshim was supported in part by the European Research Council under the European Community's SeventhFramework Programme (FP7/2007-2013 Grant Agreement No. 339563 - CryptoCloud). Hofheinz was supported by DFG Grants HO 4534/2-2 and HO 4534/4-1 and by ERC Project 724307.

A. Full Proofs from the Main Body

A.1. *Proof of Theorem 1: Indistinguishability of encodings using PIO*

Proof. We consider a chain of 10 games, with Game_0 being the κ-Switch game, such that in the last game the challenge encoding is drawn independently of the bit b. Below we let W_i denote the event that Game_i outputs 1.

Game_0: The original Switch game.

Game_1: As Game_0 but now the public parameters $\widetilde{pp}$ are changed so that they include a yes-instance $y \in \mathsf{TD}$. We have that

$$|\Pr[\text{W}_0(\lambda)] - \Pr[\text{W}_1(\lambda)]| \leq \mathbf{Adv}^{\text{sm}}_{\mathsf{TD},\mathcal{B}_1}(\lambda),$$

where TD is a language in which membership is hard to decide.

Game_2: The public parameters $\widehat{pp}$ change so that they include a hiding $\widehat{crs}'$, and a (PIO) obfuscation of circuit $\widehat{C}_{\text{Add}}$. [See Fig. 4 (bottom).] Recall that this circuit uses the witness w_y to $y \in \mathsf{TD}$ to produce the output proofs $\widetilde{\pi}''$. Therefore, the *simultaneous* knowledge of decryption keys sk_1,sk_2 is not needed anymore. Additionally, Game_2 uses w_y to prepare the proof π in the κ-Switch challenge for $\mathcal{A}$. By Lemma 1 and the perfect witness-indistinguishability of Σ, we have that

$$|\Pr[\text{W}_1(\lambda)] - \Pr[\text{W}_2(\lambda)]| \leq 2 \cdot \mathbf{Adv}^{\text{ind}}_{\mathbf{PIO},\mathcal{B}_2}(\lambda) + \mathbf{Adv}^{\text{crs}}_{\Sigma,\mathcal{B}_3}$$

Game_3: As Game_2, but, if $b = 0$, the challenge encoding is generated by mixing the representation vectors w.r.t public key pk_2. Thus, on $\mathcal{A}$'s response $(z, (\mathbf{x}_0, \mathbf{y}_0), (\mathbf{x}_1, \mathbf{y}_1))$, in this game we set $\mathbf{c}_0 \leftarrow \mathbf{Enc}(\mathbf{x}_0, pk_1; \mathbf{r}_1)$, and $\mathbf{c}_1 \leftarrow \mathbf{Enc}(\mathbf{y}_1, pk_2; \mathbf{r}_2)$.

Claim. $|\Pr[\text{W}_2(\lambda)] - \Pr[\text{W}_3(\lambda)]| \leq \mathbf{Adv}^{\text{ind-cpa}}_{\Pi,\mathcal{B}_4}(\lambda)$.

Proof Claim A.1. Consider the following PPT distinguisher $\mathcal{B}_4$ against the IND-CPA security of the encryption scheme Π, with respect to key pair (pk_2, sk_2). The distinguisher runs experiment Game_2 using $\mathcal{A}$ as a subroutine with the following differences: When it receives $\mathcal{A}$'s vectors $(\mathbf{x}_j, \mathbf{y}_j)$ (in $\mathbb{Z}_p^\ell$ for $j = 0, 1$), it submits $(\mathbf{y}_0, \mathbf{y}_1)$ to the IND-CPA challenger. It gets back $\mathbf{c}^* = \mathbf{Enc}(\mathbf{y}_{r^*}, pk_2)$. Next, $\mathcal{B}_4$ generates $\mathbf{c}_1 \leftarrow \mathbf{Enc}(\mathbf{x}_0, pk_1)$ and sets $\mathbf{c}_2 = \mathbf{c}^*$; the proof π on instance $x = ([z]_i, \mathbf{c}_1, \mathbf{c}_2)$ is generated using the simulation trapdoor of the proof system. Namely, $\pi \leftarrow\!\$\, \mathbf{Sim}(crs, x, td_{zk})$. Finally, $\mathcal{B}_4$ outputs what $\mathcal{A}$ outputs.
Algorithm $\mathcal{B}_4$ perfectly simulates the challenger in experiment Game_2 if $r^* = 0$ and in experiment Game_3 if $r^* = 1$. This follows from the facts that (1) (x, π) is a valid encoding, indeed ciphertext $\mathbf{c}^*$ contains an encryption of $\mathbf{y}_{r^*}$, such that $[z]_i = [\langle \mathbf{y}_{r^*}, \boldsymbol{\omega}_i \rangle]_i$; and (2) real and simulated proofs are identically distributed under (the hiding) $\widehat{crs}'$ included in $\widehat{pp}$. □

Game_4: The public parameters are changed back to $\widetilde{pp}$, so that they include a binding crs', and a (PIO) obfuscation of circuit C_{Add} of Fig. 2 (top). ($\widetilde{pp}$ also include a yes-instance $y \in \mathsf{TD}$.) Again by Lemma 1, we have that

$$|\Pr[\text{W}_3(\lambda)] - \Pr[\text{W}_4(\lambda)]| \leq 2 \cdot \mathbf{Adv}^{\text{ind}}_{\mathbf{PIO},\mathcal{B}_2}(\lambda) + \mathbf{Adv}^{\text{crs}}_{\Sigma,\mathcal{B}_3}.$$

Game_5: As Game_4 but now the public parameters pp are changed back to the original one described in Sect. 4 so that they include a no-instance $y \notin \mathsf{TD}$. We have that

$$|\Pr[\mathrm{W}_4(\lambda)] - \Pr[\mathrm{W}_5(\lambda)]| \leq \mathbf{Adv}^{\mathrm{sm}}_{\mathsf{TD},\mathcal{B}_1}(\lambda),$$

where TD is a language where is hard to decide membership.

Game_6: As Game_5, but now the challenger constructs a different circuit C_{Map} with the second encryption secret key hard-coded. Thus, the extracted vector is set to $\mathbf{y}_i \leftarrow \mathbf{Dec}(\mathbf{c}_{i,1}, sk_2)$. We claim that

$$|\Pr[\mathrm{W}_5(\lambda)] - \Pr[\mathrm{W}_6(\lambda)]| \leq \mathbf{Adv}^{\mathrm{ind}}_{\mathbf{PIO},\mathcal{B}_1}(\lambda).$$

The variants of the C_{Map} circuit described in the games extract (possibly different) encoding vectors $\mathbf{x}_i^*$, $\mathbf{y}_i^*$, respectively, for any adversarial input $\mathbf{x}^* = (x_1^*, \ldots, x_\kappa^*)$. Observe that the i-th argument $x_i^* = (i, [z_i]_0, \mathbf{c}_{i,1}, \mathbf{c}_{i,2}, \pi_i)$ has a non-rejecting proof π_i iff $([z_i]_0, \mathbf{c}_{i,1}, \mathbf{c}_{i,2})$ passes relation $\mathbf{R}_1$. (In other words, the ciphertexts encrypt representation vectors of the same $[z_i]_0$.) We remark that at this point, we also use Π's perfect correctness. Indeed, observe that while $\mathbf{R}_1$ implies that there *exist* encryption random coins or secret keys that decrypt $\mathbf{c}_{i,1}$ and $\mathbf{c}_{i,2}$ to consistent representation vectors $\mathbf{x}_{i,1}$ and $\mathbf{y}_{i,2}$, the perfect correctness of Π implies that the secret keys used by C_{Map} retrieve those same representation vectors $\mathbf{x}_{i,1}$ and $\mathbf{y}_{i,2}$. By the definition of $\mathbf{R}_1$, these representation vectors lead to the same outputs of C_{Map}. It follows that these variants of C_{Map} behave identically on any (possibly malformed) input $\mathbf{x}^*$. Therefore, the variants are functionally equivalent and hence trivially drawn by an X-IND sampler, so that their PIO obfuscations are indistinguishable.

Game_7: As Game_6 but now the public parameters $\widehat{pp}$ are changed so that they include a yes-instance $y \in \mathsf{TD}$. We have that

$$|\Pr[\mathrm{W}_6(\lambda)] - \Pr[\mathrm{W}_7(\lambda)]| \leq \mathbf{Adv}^{\mathrm{sm}}_{\mathsf{TD},\mathcal{B}_1}(\lambda),$$

where TD is a language where is hard to decide membership.

Game_8: The public parameters $\widehat{pp}$ change so that they include a hiding $\widehat{crs}'$, and a (PIO) obfuscation of circuit $\widehat{C}_{\mathrm{Add}}$. [See Fig. 4 (bottom).] By Lemma 1, we have that

$$|\Pr[\mathrm{W}_7(\lambda)] - \Pr[\mathrm{W}_8(\lambda)]| \leq 2 \cdot \mathbf{Adv}^{\mathrm{ind}}_{\mathbf{PIO},\mathcal{B}_2}(\lambda) + \mathbf{Adv}^{\mathrm{crs}}_{\Sigma,\mathcal{B}_3}$$

Game_9: As Game_8, but, if $b = 0$, the challenge encoding is generated by mixing the representation vectors w.r.t public key pk_1. Thus, on $\mathcal{A}$'s response $(z, (\mathbf{x}_0, \mathbf{y}_0), (\mathbf{x}_1, \mathbf{y}_1))$, in this game, we set $\mathbf{c}_0 \leftarrow \mathbf{Enc}(\mathbf{x}_1, pk_1; \mathbf{r}_1)$, and $\mathbf{c}_1 \leftarrow \mathbf{Enc}(\mathbf{y}_1, pk_2; \mathbf{r}_2)$. Using a similar argument as in Claim A.1, we have that

$$|\Pr[\mathrm{W}_8(\lambda)] - \Pr[\mathrm{W}_9(\lambda)]| \leq \mathbf{Adv}^{\text{ind-cpa}}_{\Pi,\mathcal{B}_4}(\lambda).$$

Finally, $\Pr[\mathrm{W}_9(\lambda)] = 1/2$ because the challenge encoding is generated using the same pair of representation vectors $(\mathbf{x}_1, \mathbf{y}_1)$ regardless of the bit b. The proof of the theorem is concluded by collecting the terms above. □

A.2. *Proof of Theorem 2: Hardness of Symmetric MDDH*

Proof. We show via a chain of games, starting with the symmetric κ-MDDH problem, such that the last game chooses the challenge at random and independently of the guess bit b. Below we let W_i denote the event that Game_i outputs 1.

Game_0: The κ-MDDH problem as shown in Fig. 7. Here, there is only one source group.

(κ, I^*)-MDDH$^{\mathcal{A}}_{I^*}(\lambda)$:
$pp \leftarrow\!\!\$\ \mathbf{Setup}(1^\lambda, 1^\kappa)$
$b \leftarrow\!\!\$\ \{0,1\};\ z \leftarrow\!\!\$\ \mathbb{Z}_N$
$a_1, \ldots, a_{\kappa+1} \leftarrow\!\!\$\ \mathbb{Z}_N$
if $b = 1$ then $z \leftarrow a_1 \cdots a_{\kappa+1}$
$b' \leftarrow\!\!\$\ \mathcal{A}(pp, \{[a_i]_j\}_{(i,j)\in I^*}, [z]_T)$
Return $(b = b')$

Fig. 7. The symmetric multilinear DDH problem for our MLG scheme. Here, $I^* = \{(1,1), \ldots, (\kappa+1, 1)\}$.

Game$_s$ for $1 \le s \le \kappa + 1$: As Game$_{s-1}$, the difference is that the representation vectors $(\mathbf{x}_s, \mathbf{y}_s)$ of the sth challenge encoding $[a_s]$ are given by

$$x_{s,0} = y_{s,0} = a_s - \omega \quad \text{and} \quad x_{s,1} = y_{s,1} = 1.$$

Thus, in game $s' \ge s$, the second coordinates of the sth encoding vectors are *always* fixed to 1. Now, a straightforward reduction yields an adversary $\mathcal{B}$ that satisfies:

Claim.

$$|\Pr[\mathrm{W}_{s-1}(\lambda)] - \Pr[\mathrm{W}_s(\lambda)]| \le \mathbf{Adv}^{\kappa\text{-switch}}_{I^*,\mathcal{B}}(\lambda) \text{ for } 1 \le s \le \kappa + 1 .$$

Proof. Consider the following PPT adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ against game κ-Switch of Fig. 3. $\mathcal{B}_1$ outputs $((\mathbf{x}_{s-1}, \mathbf{y}_{s-1}), (\mathbf{x}_s, \mathbf{y}_s), s, st)$ representing a uniform value a_s in $\mathbb{Z}_N$, where $(\mathbf{x}_{s-1}, \mathbf{y}_{s-1})$ is as in Game$_{s-1}$ and $(\mathbf{x}_s, \mathbf{y}_s)$ as in Game$_s$. $\mathcal{B}_1$ can form these vectors because it knows matrix $\mathbf{W}$ and a_s explicitly. Next, $\mathcal{B}_2$ receives an encoding $[a_s]_s$ that has embedded in it vector $(\mathbf{x}_{s+b-1}, \mathbf{y}_{s+b-1})$ for a random bit b and uses $[a_s]_s$ to simulate Game$_{s+b-1}$. Last, $\mathcal{B}_2$ outputs what $\mathcal{A}$ outputs.

Game$_{\kappa+2}$: The ith source exponent is changed to $a'_i = a_i + \omega$ for randomly chosen $a_i \in \mathbb{Z}_N$ and all $i \in [\kappa + 1]$. This means that the target exponent for $b = 1$ is

$$d = (a_1 + \omega) \cdots (a_{\kappa+1} + \omega) \tag{2}$$

The distribution from which the exponents a'_i are drawn has not changed and indeed is the uniform distribution. Therefore, $\Pr[\mathrm{W}_{\kappa+1}(\lambda)] = \Pr[\mathrm{W}_{\kappa+2}(\lambda)]$.

Game$_{\kappa+3}$: The differences with the previous game are twofold.
First, for case $b = 1$, the challenge group element $[d]_T$ is generated as in Lemma 3. More precisely, we first write Eq. (2) as

$$d = P(\omega) ,$$

where P is a degree $\kappa + 1$ polynomial whose coefficients $\mathbf{p} = (p_0, \ldots, p_\kappa, p_{\kappa+1})$ are computed using the iterative rule of Lemma 3, with $(x_{i,0}, x_{i,1}) = (a_i, 1)$. Then, $[d]_T$ is obtained by evaluating P at point ω in the exponent using group elements $([1]_T, [\omega]_T, \ldots, [\omega^\kappa]_T, [\omega^{\kappa+1}]_T)$.
The other difference is that we obfuscate a different circuit C^*_{Map} which has the powers $[\omega^i]_T$ hard-coded, for $1 \le i \le \kappa$. This new circuit extracts the encoding vectors $\mathbf{x}_i$ from the inputs, as usual; then, it computes the coefficients of $Q(w) = \prod_{i=1}^{\kappa}(x_{i,0} + x_{i,1}\omega)$ by Lemma 3 and evaluates it at ω in the exponent.

$\underline{(\kappa, I^*)\text{-MDDH}^{\mathcal{A}}_{\Gamma}(\lambda)}$:
$pp \leftarrow\!\!\$\ \mathbf{Setup}(1^\lambda, 1^\kappa)$
$b \leftarrow\!\!\$\ \{0,1\};\ z \leftarrow\!\!\$\ \mathbb{Z}_N$
$a_1, \ldots, a_{\kappa+1} \leftarrow\!\!\$\ \mathbb{Z}_N$
if $b = 1$ then $z \leftarrow a_1 \cdots a_{\kappa+1}$
$b' \leftarrow\!\!\$\ \mathcal{A}(pp, \{[a_i]_j\}_{(i,I(i))}, [z]_T)$
Return $(b = b')$

Fig. 8. The asymmetric multilinear DDH problem for our MLG scheme. Here, I is a function defining the index set $I = (i, I(i))$.

Lemma 3 implies that (1) both circuits are functionally equivalent, and (2) C^*_{Map} is of size $poly(\lambda)$. We conclude that obfuscations of these two variants are indistinguishable. Or putting it differently:

$$|\Pr[\mathrm{W}_{\kappa+2}(\lambda)] - \Pr[\mathrm{W}_{\kappa+3}(\lambda)]| \leq \mathbf{Adv}^{\text{ind}}_{\mathbf{PIO},\mathcal{B}}(\lambda) .$$

Game$_{\kappa+4}$: The last game samples the challenge $[d]_T$ for case $b = 1$ as $[d]_T = [\sigma]_T$ for independently random $\sigma \in \mathbb{Z}_N$. A κ-SDDH challenge $([\omega^i]_0)_{i\leq\kappa}, [\sigma]_0)$ can be used to emulate the challenger in Game$_{\kappa+3}$ if $\sigma = \omega^{\kappa+1}$, or in Game$_{\kappa+4}$ if σ is random. The latter follows from the fact that knowing ω^i in the exponent for $i \in [\kappa + 1]$ suffices to generate $[d]_T$. (Recall that $\mathbb{G}_T = \mathbb{G}_0$.) This shows:

$$|\Pr[\mathrm{W}_{\kappa+3}(\lambda)] - \Pr[\mathrm{W}_{\kappa+4}(\lambda)]| \leq \mathbf{Adv}^{\kappa\text{-sddh}}_{\Gamma_0,\mathcal{B}}(\lambda) .$$

To conclude, to see that $\Pr[\mathrm{W}_{\kappa+4}] \leq 1/2$, it suffices to observe that the exponent target challenge d is randomly distributed, regardless of the challenge bit b. □

A.3. *Proof of Theorem 3: Hardness of Asymmetric MDDH*

Proof. Let $I : [\kappa + 1] \longrightarrow [\kappa]$ be any function. Slightly abusing notation, we set $I = (i, I(i))$ for $1 \leq i \leq \kappa + 1$. By the pigeon-hole principle, there must exist a pair of distinct $i, i' \in [\kappa + 1]$ such that $I(i) = I(i') \in [\kappa]$. For simplicity, and without loss of generality, we assume that $I(1) = I(2) = 1$.
We show a chain of games, starting with the asymmetric (κ, I)-MDDH problem, such that the last game chooses the challenge encoding at random and independently of the challenge bit b. Below we let W_i denote the event that Game$_i$ outputs 1.

Game$_0$: The asymmetric (κ, I)-MDDH problem as shown in Fig. 8.

Game$_s$ for $s = 1, 2$: Similar to Game$_{s-1}$ with the difference that the representation vectors $(\mathbf{x}_s, \mathbf{y}_s)$ of the source encoding $[a_s]_1$ are given by

$$x_{s,0} = y_{s,0} = a_s - \omega_1 \quad \text{and} \quad x_{s,1} = y_{s,1} = 1 .$$

Thus, in game $s' \geq s$, the second coordinates of the sth encoding vectors are *always* fixed to 1. Using a similar argument as Claim A.2, we have that

$$|\Pr[\mathrm{W}_{s-1}(\lambda)] - \Pr[\mathrm{W}_s(\lambda)]| \leq \mathbf{Adv}^{\kappa\text{-switch}}_{\Gamma^*,\mathcal{B}}(\lambda) .$$

Game$_3$: We change the first two source exponents to $a'_i = a_i + \omega_1$ for randomly chosen $a_i \in \mathbb{Z}_N$. This means that the target exponent for $b = 1$ is

$$d = (a_1 + \omega_1)(a_2 + \omega_1) \cdot a_3 \cdots a_{\kappa+1} .$$

The first two elements a'_i are drawn from the uniform distribution, and their respective representation vectors are $(a_i, 1)$ so $\Pr[\mathrm{W}_2(\lambda)] = \Pr[\mathrm{W}_3(\lambda)]$.

Game$_4$: The implementation of C_{Map} is changed. Now, it has hard-coded

$$[\omega_1]_0, \omega_2, \omega_3, \ldots, \omega_\kappa \ .$$

The polynomial $P(\omega_1, \ldots, \omega_\kappa) = \prod_{i=1}^{\kappa}(x_{i,0}+x_{i,1}\omega_i)$ on point $(\omega_1, \ldots, \omega_\kappa)$ can be evaluated in the exponent knowing $[\omega_1]_0$ and explicit ω_i for $i \geq 2$. Since the output of the original C_{Map} is exactly $[P(\omega_1, \ldots, \omega_\kappa)]_T$, we conclude that

$$|\Pr[\mathrm{W}_3(\lambda)] - \Pr[\mathrm{W}_4(\lambda)]| \leq \mathbf{Adv}^{\text{ind}}_{\mathbf{PIO},\mathcal{B}}(\lambda) \ .$$

Game$_5$: The challenge target d is set to

$$d = (a_1 a_2 + \omega_1 a_2 + \omega_1 a_1 + \sigma) \cdot a_3 \cdots a_{\kappa+1} \ , \tag{3}$$

where σ is a fresh random value in $\mathbb{Z}_N$.
Note that if $\sigma = {\omega_1}^2$, then this is precisely the challenge target d in the previous game. Thus, a 1-SDDH challenge $([\omega_1]_0, [\sigma]_0)$ can be used to generate the pair $([d]_T, \overline{C^*}_{\text{Map}})$ as in Game$_4$ if $\sigma = {\omega_1}^2$, or as in Game$_5$ if σ is random. This shows:

$$|\Pr[\mathrm{W}_4(\lambda)] - \Pr[\mathrm{W}_5(\lambda)]| \leq \mathbf{Adv}^{\text{1-sddh}}_{\Gamma_0,\mathcal{B}}(\lambda) \ .$$

To conclude, we have $\Pr[\mathrm{W}_5(\lambda)] \leq 1/2 + \mathrm{negl}(\lambda)$. To see this, we argue that d is randomly distributed in $\mathbb{Z}_N$ for challenge bit $b = 1$ with overwhelming probability in λ as follows: If N is prime, then $\prod_{j=3}^{\kappa+1} a_j$ has an inverse in $\mathbb{Z}_N$, and therefore, d in Eq. (3) seen as a function of σ and parametrized by a_j defines a bijection in $\mathbb{Z}_N$ with overwhelming probability. Thus, if σ is uniform so is d. □

References

[1] B. Applebaum, Z. Brakerski, Obfuscating circuits via composite-order graded encoding. in *Dodis and Nielsen [18]*, pp. 528–556

[2] T. Agrikola, D. Hofheinz, Interactively secure groups from obfuscation. in *Proc. PKC 2018*, volume 10770 of *Lecture Notes in Computer Science*, (Springer, 2018), pp. 341–370

[3] P. Ananth, A. Sahai. Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. in J.-S. Coron, J.B. Nielsen, editors, *EUROCRYPT 2017, Part I*, LNCS, vol. 10210 (Springer, Heidelberg, May 2017), pp. 152–181

[4] D. Boneh, X. Boyen, H. Shacham, Short group signatures, in M. Franklin, editor, *CRYPTO 2004*, LNCS, vol. 3152, (Springer, Heidelberg, August 2004), pp. 41–55

[5] D. Boneh, K. Lewi, M. Raykova, A. Sahai, M. Zhandry, J. Zimmerman, Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation. in *Oswald and Fischlin [38]*, pp. 563–594

[6] X. Boyen. The uber-assumption family (invited talk). in S.D. Galbraith, K.G. Paterson, editors, *PAIRING 2008*, LNCS, vol. 5209 (Springer, Heidelberg, September 2008), pp. 39–56

[7] D. Boneh, A. Silverberg, Applications of multilinear forms to cryptography. *Contemporary Mathematics*, **324**, 71–90 (2003)

[8] D. Boneh, B. Waters, M. Zhandry, Low overhead broadcast encryption from multilinear maps. in J.A. Garay, R. Gennaro, editors, *CRYPTO 2014, Part I*, LNCS, vol. 8616, (Springer, Heidelberg, August 2014), pp. 206–223

[9] J.H. Cheon, P.-A. Fouque, C. Lee, B. Minaud, H. Ryu, Cryptanalysis of the new CLT multilinear map over the integers. in *Fischlin and Coron [20]*, pp. 509–536.

[10] R. Canetti, J.A. Garay, editors, *CRYPTO 2013, Part I*, LNCS, vol. 8042. (Springer, Heidelberg, August 2013)

[11] R. Canetti, J.A. Garay, editors, *CRYPTO 2013, Part II*, LNCS, vol. 8043 (Springer, Heidelberg, August 2013)
[12] J.-S. Coron, C. Gentry, S. Halevi, T. Lepoint, H.K. Maji, E. Miles, M. Raykova, A. Sahai, M. Tibouchi, Zeroizing without low-level zeroes: New MMAP attacks and their limitations. in *Gennaro and Robshaw [28]*, pp. 247–266
[13] J.H. Cheon, K. Han, C. Lee, H. Ryu, D. Stehlé. Cryptanalysis of the multilinear map over the integers. in E. Oswald, M. Fischlin, editors, *EUROCRYPT 2015, Part I*, LNCS, vol. 9056 (Springer, Heidelberg, April 2015), pp. 3–12
[14] J.-S. Coron, M.S. Lee, T. Lepoint, M. Tibouchi, Cryptanalysis of GGH15 multilinear maps. in *Robshaw and Katz [41]*, pp. 607–628
[15] J.-S. Coron, T. Lepoint, M. Tibouchi, Practical multilinear maps over the integers. in *Canetti and Garay [10]*, pp. 476–493
[16] J.-S. Coron, T. Lepoint, M. Tibouchi. New multilinear maps over the integers. in *Gennaro and Robshaw [28]*, pp. 267–286
[17] R. Canetti, H. Lin, S. Tessaro, V. Vaikuntanathan, Obfuscation of probabilistic circuits and applications. in *Dodis and Nielsen [18]*, pp. 468–497
[18] Y. Dodis, J.B. Nielsen, editors. *TCC 2015, Part II*, LNCS, vol. 9015 (Springer, Heidelberg, March 2015)
[19] A. Escala, G. Herold, E. Kiltz, C. Ràfols, J. Villar. An algebraic framework for Diffie-Hellman assumptions. in *Canetti and Garay [11]*, pp. 129–147
[20] M. Fischlin, J.-S. Coron, editors. *EUROCRYPT 2016, Part I*, LNCS, vol. 9665 (Springer, Heidelberg, May 2016)
[21] P. Farshim, J. Hesse, D. Hofheinz, E. Larraia. Graded encoding schemes from obfuscation. in *Proc. PKC 2018*, LNCS, vol. 10770 (Springer, 2018), pp. 371–400
[22] E.S.V. Freire, D. Hofheinz, K.G. Paterson, C. Striecks. Programmable hash functions in the multilinear setting. in *Canetti and Garay [10]*, pp. 513–530
[23] S. Garg, C. Gentry, S. Halevi. Candidate multilinear maps from ideal lattices. in T.J. Phong, Q. Nguyen, editors, *EUROCRYPT 2013*, LNCS, vol. 7881 (Springer, Heidelberg, May 2013), pp. 1–17
[24] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, B. Waters, Candidate indistinguishability obfuscation and functional encryption for all circuits. in *54th FOCS*, (IEEE Computer Society Press, October 2013), pp. 40–49
[25] S. Garg, C. Gentry, S. Halevi, A. Sahai, B. Waters, Attribute-based encryption for circuits from multilinear maps. in *Canetti and Garay [11]*, pp. 479–499
[26] C. Gentry, S. Gorbunov, S. Halevi, Graph-induced multilinear maps from lattices. in *Dodis and Nielsen [18]*, pp. 498–527
[27] S. Garg, C. Gentry, A. Sahai, B. Waters, Witness encryption and its applications. in D. Boneh, T. Roughgarden, J. Feigenbaum, editors, *45th ACM STOC* (ACM Press, June 2013), pp. 467–476
[28] R. Gennaro, M.J.B. Robshaw, editors, *CRYPTO 2015, Part I*, LNCS, vol. 9215 (Springer, Heidelberg, August 2015)
[29] J. Groth, A. Sahai. Efficient non-interactive proof systems for bilinear groups. in N.P. Smart, editor, *EUROCRYPT 2008*, LNCS, vol. 4965 (Springer, Heidelberg, April 2008), pp. 415–432
[30] Y. Hu, H. Jia, Cryptanalysis of GGH map. in *Fischlin and Coron [20]*, pp. 537–565
[31] S. Hohenberger, A. Sahai, B. Waters. Full domain hash from (leveled) multilinear maps and identity-based aggregate signatures. in *Canetti and Garay [10]*, pp. 494–512
[32] J. Katz, H. Shacham, editors. *CRYPTO 2017, Part I*, LNCS, vol. 10401 (Springer, Heidelberg, August 2017)
[33] H. Lin, Indistinguishability obfuscation from constant-degree graded encoding schemes. in *Fischlin and Coron [20]*, pp. 28–57
[34] H. Lin, Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. in *Katz and Shacham [32]*, pp. 599–629
[35] H. Lin, S. Tessaro, Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. In *Katz and Shacham [32]*, pp. 630–660
[36] E. Miles, A. Sahai, M. Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. in *Robshaw and Katz [41]*, pp. 629–658
[37] M. Naor, M. Yung, Public-key cryptosystems provably secure against chosen ciphertext attacks. in *22nd ACM STOC* (ACM Press, May 1990), pp. 427–437

[38] E. Oswald, M. Fischlin, editors. *EUROCRYPT 2015, Part II*, LNCS, vol. 9057 (Springer, Heidelberg, April 2015)
[39] O. Paneth, A. Sahai, On the equivalence of obfuscation and multilinear maps. Cryptology ePrint Archive, Report 2015/791, (2015). http://eprint.iacr.org/2015/791
[40] C. Papamanthou, R. Tamassia, N. Triandopoulos, Optimal authenticated data structures with multilinear forms. in M. Joye, A. Miyaji, A. Otsuka, editors, *PAIRING 2010*, LNCS, vol. 6487 (Springer, Heidelberg, December 2010), pp. 246–264
[41] M. Robshaw, J. Katz, editors. *CRYPTO 2016, Part II*, LNCS, vol. 9815 (Springer, Heidelberg, August 2016)
[42] F. Tang, H. Li, B. Liang, Attribute-based signatures for circuits from multilinear maps. in S.S.M. Chow, J. Camenisch, L.C.K. Hui, S.-M. Yiu, editors, *ISC 2014*, LNCS, vol. 8783 (Springer, Heidelberg, October 2014), pp. 54–71
[43] T. Yamakawa, S. Yamada, G. Hanaoka, N. Kunihiro. Self-bilinear map on unknown order groups from indistinguishability obfuscation and its applications. in J.A. Garay, R. Gennaro, editors, *CRYPTO 2014, Part II*, LNCS, vol. 8617(Springer, Heidelberg, August 2014), pp. 90–107
[44] T. Yamakawa, S. Yamada, G. Hanaoka, N. Kunihiro. Self-bilinear map on unknown order groups from indistinguishability obfuscation and its applications. Cryptology ePrint Archive, Report 2015/128, (2015). http://eprint.iacr.org/2015/128
[45] J. Zimmerman. How to obfuscate programs directly. in *Oswald and Fischlin [38]*, pp. 439–467