Interactively Secure Groups from Obfuscation

Thomas Agrikola(✉) and Dennis Hofheinz

Karlsruhe Institute of Technology, Karlsruhe, Germany
{Thomas.Agrikola,Dennis.Hofheinz}@kit.edu

Abstract. We construct a mathematical group in which an interactive variant of the very general Uber assumption holds. Our construction uses probabilistic indistinguishability obfuscation, fully homomorphic encryption, and a pairing-friendly group in which a mild and standard computational assumption holds. While our construction is not practical, it constitutes a feasibility result that shows that under a strong but generic, and a mild assumption, groups *exist* in which very general computational assumptions hold. We believe that this grants additional credibility to the Uber assumption.

Keywords: Indistinguishability obfuscation · Uber assumption

1 Introduction

Cyclic groups in cryptography. Cyclic groups (such as subgroups of the multiplicative group of a finite field, or certain elliptic curves) are a popular mathematical building block in cryptography. Countless cryptographic constructions are formulated in a cyclic group setting. Usually these constructions are accompanied by a security reduction that transforms any adversarial algorithm that breaks the scheme into an algorithm that solves a computational problem in that group. Among the more popular computational problems are the (computational or decisional) Diffie-Hellman problem [25], or the discrete logarithm problem.

The currently known security reductions of several relevant cryptographic schemes require somewhat more exotic computational assumptions, however. For instance, the security of the Digital Signature Algorithm is only proven in a generic model of computation [14] (see also [15]). Moreover, the semi-adaptive (i.e., IND-CCA1) security of the ElGamal encryption scheme requires a "one-more type" assumption [33]. The currently most efficient structure-preserving signature schemes require complex interactive assumptions [1,2]. Finally, some proofs (e.g., [6,23,24,31]) even require "knowledge assumptions" that essentially state that the only way to generate new group elements is as linear combinations of given group elements (with extractable coefficients).

T. Agrikola—Supported by ERC Project PREP-CRYPTO 724307.
D. Hofheinz—Supported by ERC Project PREP-CRYPTO 724307, and by DFG grants HO 4534/4-1 and HO 4534/2-2.

© International Association for Cryptologic Research 2018
M. Abdalla and R. Dahab (Eds.): PKC 2018, LNCS 10770, pp. 341–370, 2018.
https://doi.org/10.1007/978-3-319-76581-5_12

While more exotic assumptions can thus be very helpful for constructing cryptographic schemes, their use also has a downside: reductions to more exotic (and less investigated) assumptions tend to lower our confidence in the corresponding scheme. (See [12,32] for two very different views on this matter).

The Uber-assumption family. An example of a somewhat exotic but very general and strong class of computational assumptions in a cyclic group setting is the "Uber" assumption family ([10], see also [12]). Essentially, this assumption states that no efficient adversary $\mathcal{A}$ can win the following guessing game significantly better than with probability 1/2. The game is formulated in a group $\mathcal{G} = \langle g \rangle$ of order q, and is parameterized over polynomials $P_1, \ldots, P_l, P^* \in \mathbb{Z}_q[X_1, \ldots, X_m]$. Initially, the game chooses secret exponents $s_1, \ldots, s_m \in \mathbb{Z}_q$ uniformly, and hands $\mathcal{A}$ the group elements $g^{P_i(s_1,\ldots,s_m)}$, and a challenge element $Z \in \mathcal{G}$ with either $Z = g^{P^*(s_1,\ldots,s_m)}$ or independently random Z. Given these elements, $\mathcal{A}$ has to guess if Z is random or not.[1]

Depending on the number m of variables, and the concrete polynomials P_i and P^*, the Uber assumption generalizes many popular existing assumptions, such as the Decisional Diffie-Hellman assumption, the k-Linear family of assumptions, and so-called "q-type assumptions". However, it is a priori not at all clear how plausible such general assumptions are. In fact, there are indications that, e.g., q-type assumptions are indeed easier to break than, say, the discrete log assumption [21].

Fortunately, a number of cryptographic constructions that rely on q-type assumptions can be transported into composite-order groups, with the advantage that now their security holds under a simpler, subgroup indistinguishability assumption [19,20]. However, this change of groups will not work for every cryptographic construction, and currently we only know how to perform this technique for a subclass of q-type assumptions.

Our contribution. In this work, we shed new light on the plausibility of Uber-style assumptions. Concretely, we construct a group in which an interactive variant of Uber-style assumptions (in which the adversary may choose the P_i and P^* adaptively) holds. We believe that this lends additional credibility to the Uber assumption itself, and also strengthens plausibility results obtained from the Uber assumption (see [12] for an overview).

Our construction assumes subexponentially secure indistinguishability obfuscation (iO, a very strong but generic assumption), a perfectly correct additively homomorphic encryption scheme for addition modulo a given prime, and a pairing-friendly group in which a standard assumption (SXDH, the symmetric external Diffie-Hellman assumption) holds. We stress that we consider our result as a feasibility result. Indeed, due to the use of indistinguishability obfuscation, our construction is far from practical. Still, our result shows that even interactive generalizations of the Uber assumption family are no less plausible than

[1] Owing to the original application, the Uber assumption family was formulated in [10] in a setting with a pairing-friendly group, with a final challenge in the target group.

indistinguishability obfuscation (plus a standard assumption in cyclic groups and additively homomorphic encryption).

Before describing our results in more detail, we remark that the group we construct actually has non-unique element encodings (much like in a "graded encoding scheme" [26], only without any notion of multilinear map). It is hence possible to compare and operate with group elements, but it is not directly possible to use, e.g., the encoding of group elements to hide an encrypted message. (In particular, it is not immediately possible to implement, say, the ElGamal encryption scheme with our group as there is no obvious way to decrypt ciphertexts. Signature schemes, however, do not require unique encodings of group elements and can hence be implemented using our group.) Furthermore, due to technical reasons our construction requires the maximum degree of the adversarially chosen polynomials to be bounded a priori.

Related work. Pass et al. [36] introduce semantically secure multilinear (and graded) encoding schemes (of groups). A semantically secure encoding scheme guarantees security of a class of algebraic decisional assumptions. On a high level, the security property requires that encodings are computationally indistinguishable whenever there is no way to distinguish the corresponding elements using only generic operations. The generic multilinear encoding model implies semantic security of a multilinear encoding scheme. Furthermore, Pass et al. show that many existing iO candidates [5,13,27] that are proven secure in the generic multilinear encoding model can also be proven secure assuming semantically secure encoding schemes. Hence, this result relaxes the necessary assumptions to prove the security of certain iO constructions. Bitansky et al. [8] slightly strengthen the security property of encoding schemes formulated in [36]. Assuming the resulting security property allows to prove that existing obfuscation candidates [5] provide virtual grey-box security[2].

In [4] Albrecht et al. construct a group scheme providing a multilinear map from iO. This result complements earlier results that construct iO from multilinear maps [27,38]. The notion of encoding schemes used in [4] is a direct adaption of the "cryptographic" multilinear group setting from [11]. In contrast to [8,36], the encoding scheme of Albrecht et al. provides an extraction algorithm producing a unique string for all encodings that are equal with respect to the equality relation of the scheme. Furthermore, [4] requires a publicly available sampling algorithm that produces encodings for given exponents. Hence, the encoding scheme of [4] grants adversaries slightly more power.

In this paper we use a similar notion of encoding schemes as in [4]. Furthermore, [8,36] define the security property for encoding schemes implicitly. We, in contrast, consider a concrete strong interactive hardness assumption that holds in our encoding scheme.

[2] An obfuscator $\mathcal{O}$ satisfies virtual grey-box security for a class of circuits $\mathcal{C}$ if for any circuit $C \in \mathcal{C}$, a PPT adversary given $\mathcal{O}(C)$ can not compute significally more about C than a simulator given unbounded computational resources and polynomially many queries to the circuit C.

Technical approach. The assumption we consider is defined similarly to the Uber assumption above, only with an interactive and adaptive choice of arbitrary (multivariate) polynomials P_i, P^* over $\mathbb{Z}_q$, where q is the order of the group. That is, there is a secret point $\boldsymbol{s} := (s_1, \ldots, s_m) \in \mathbb{Z}_q^m$, and $\mathcal{A}$ may freely and adaptively choose the P_i and P^* during the course of the security game. To avoid trivialities, we require that P^* does not lie in the linear span of the polynomials P_i. We call this assumption the *Interactive Uber assumption*. For convenience only, we will describe our approach assuming only univariate polynomials in the Interactive Uber assumption. However, we will see that similar techniques yield security even for multivariate polynomials.

Our starting point is a recent work by Albrecht et al. [4], which constructs a group with a multilinear map from (probabilistic) iO, an additively homomorphic encryption scheme, a dual mode NIZK proof system, and a group $\mathcal{G}$ in which (a variant of) the Strong Diffie-Hellman assumption [9] holds. For our purposes, we are not interested in obtaining a multilinear map, however, and we would also like to avoid relying on a strong (i.e., q-type) assumption to begin with. Moreover, [4] only proves relatively mild computational assumptions in the constructed group.

In a nutshell, a group element in the construction of [4] has the form

$$(g^z,\, C = \textsc{Enc}(z),\, \pi), \tag{1}$$

where $z \in \mathbb{Z}$ is the discrete logarithm of that group element, $g \in \mathcal{G}$ is a generator of the used existing group $\mathcal{G}$, Enc is the encryption algorithm of an additively homomorphic encryption scheme, and π is a non-interactive zero-knowledge proof of consistency. Concretely, π proves that C encrypts the discrete logarithm z of g^z, *or* that C encrypts a polynomial f with $f(w) = z$, for a fixed value w committed to in the public parameters.

In their security analysis, Albrecht et al. [4] crucially use a "switching lemma" that states that different encodings $(g^z, \textsc{Enc}(z), \pi)$ and $(g^{f(w)}, \textsc{Enc}(f), \pi')$ are computationally indistinguishable whenever $f(w) = z$. This allows to switch to, and argue about encodings with higher-degree f. Note, however, that any such encoding must also carry a valid $g^z = g^{f(w)}$. Hence, changing the values $z = f(w)$ in such encodings with higher-degree f (as is often required to prove security) would seem to already necessitate Uber-style assumptions. Indeed, Albrecht et al. require a variant of the Strong Diffie-Hellman assumption, a q-type assumption.

Group elements in our group. To avoid making Uber-style assumptions in the first place, we simply omit the initial g^z value in encodings of group elements, and modify the consistency proof from Eq. (1). That is, group elements in our group are of the form

$$(C = \textsc{Enc}(z),\, \pi), \tag{2}$$

where Enc is the encryption algorithm of an additively homomorphic encryption scheme, and π is a proof of knowledge of some (potentially constant) polynomial f' with $f'(w) = z$ or $f'(w) = f(w)$ (in case C encrypts a polynomial f). The value w is some point in $\mathbb{Z}_q$ that is fixed, but hidden, in the public parameters

of our group, where q is the group order. The proof of knowledge is realized through an additional encryption C' that contains the polynomial f'. Hence, group elements are actually of the form

$$(C = \textsc{Enc}(z),\, C' = \textsc{Enc}(f'),\, \pi). \tag{3}$$

In a nutshell, such an encoding implicitly represents the group element $g^{f(w)} = g^{f'(w)}$, where f and f' are the polynomials defined by C and C' respectively. For clarity, we sometimes omit the component C' in this overview.

More precisely, C and C' contain representation vectors $\vec{f}$ and $\vec{f'}$ of the polynomials f and f' with respect to a basis $\{a_1, \ldots, a_d\}$ of $\mathbb{Z}_q^d$. That is, given a vector $\vec{f}$ that is encrypted in C, the coefficients of the corresponding polynomial f are defined as follows

$$(a_1 \mid \ldots \mid a_d)^{-1} \cdot \vec{f} \tag{4}$$

using the homomorphic mapping between polynomials over $\mathbb{Z}_q$ and vectors in $\mathbb{Z}_q^d$. This basis is not public, but committed to in the public parameters. The reason for using a hidden basis is that we need to deal with adaptive queries. We postpone the details to a subsequent paragraph. In this overview, however, we will pretend the ciphertexts C and C' contain mere polynomials.

Intuitively, the crux of the matter for the proof of security will be to remove the dependency on the point w. This changes the group structure to be isomorphic to $\mathbb{Z}_q^d$ which makes it possible to argue with linear algebra.

A public sampling algorithm allows to produce arbitrary encodings of group elements. Given an exponent z, the sampling algorithm produces the ciphertexts C and C' using the constant polynomials $f := f' := z$ and produces the consistency proof accordingly. We remark that our group allows for re-randomization of encodings assuming some natural additional properties of the homomorphic encryption scheme.

The group operation is performed in a similar way to [4]. Namely, suppose we want to add two encodings $(\textsc{Enc}(f_1), \pi_1)$ and $(\textsc{Enc}(f_2), \pi_2)$. The resulting $(\textsc{Enc}(f_3), \pi_3)$ should satisfy $f_3 = f_1 + f_2$ as abstract polynomials. Hence, $\textsc{Enc}(f_3)$ can be computed homomorphically from $\textsc{Enc}(f_1)$ and $\textsc{Enc}(f_2)$. To compute the proof π_3, however, we require an obfuscated circuit C_{Add} that extracts f_1, f_2, and generates a fresh proof using the knowledge of $f_3 = f_1 + f_2$ as witness. Thus, the implementation of C_{Add} needs to know both decryption keys for C and C'. (The details are somewhat technical and similar to [4], so we omit them in this overview.) We prove that it is possible to implement a circuit C''_{Add} that has almost the same functionality as C_{Add} but produces a simulated proof of consistency that is identically distributed to a real one. Hence, the implementation of C''_{Add} does not need to know the decryption keys. Therefore, exploiting the security of the used obfuscator, we are able to unnoticeably replace the obfuscation of C_{Add} with an obfuscation of C''_{Add}.

We note that our modification to omit the entry g^z from the encodings in Eq. (1) makes it nontrivial to decide whether two given encodings represent the same group element, or, equivalently, to decide whether a given encoding

represents the identity element of the group. Recall that an encoding $(C = \text{Enc}(f), \pi)$ represents the group element $g^{f(w)}$. (This operation is trivial in the setting of Albrecht et al., since their encodings carry a value $g^z = g^{f(w)}$.) Thus, our construction needs to provide a public algorithm that tests whether a given encoding $(C = \text{Enc}(f), \pi)$ represents the identity element of the group, i.e. that tests whether $f(w) = 0$.

At this point two problems arise. First, this public algorithm must be able to obtain at least one of the polynomials that are encrypted in C and C' respectively. Second, the value w must not be explicitly known during the proof of security as our strategy is to remove the dependency on w. We solve both problems by using an *obfuscated* circuit C_{Zero} for testing whether a given encoding represents the identity element. More precisely, given an encoding $(C = \text{Enc}(f), \pi)$, C_{Zero} decrypts C (using one fixed decryption key) to obtain the polynomial f. In order to avoid the necessity to explicitly know the value w, C_{Zero} factors the univariate polynomial f (in $\mathbb{Z}_q[X]$), and obtains the small set $\{x_1, \ldots, x_n\}$ of all zeros of f.[3] As mentioned above, the value w is fixed but hidden inside the public parameters. Particularly, we store the value w in form of a point function obfuscation (i.e., in form of a publicly evaluable function $\mathsf{po}\colon \mathbb{Z}_q \to \{0,1\}$ with $\mathsf{po}(x) = 1 \Leftrightarrow x = w$, such that it is hard to determine the value w given only the function description po). The zero testing circuit C_{Zero} treats an encoding as the identity element if f is the zero polynomial or $w \in \{x_1, \ldots, x_n\}$.

Observe that this implementation of C_{Zero} only requires one decryption key allowing to apply the Naor-Yung strategy [35]. Furthermore, C_{Zero} does not need to know the value w in the clear. Hence, using an obfuscation of this implementation of C_{Zero} avoids both problems described above.

Switching of encodings. Similarly to Albrecht et al. [4] we prove a "switching lemma" that states that encodings $(C_1 = \text{Enc}(f_1), \pi_1)$ and $(C_2 = \text{Enc}(f_2), \pi_2)$ are computationally indistinguishable whenever $f_1(w) = f_2(w)$. In other words, encodings of the same group element are computationally indistinguishable. To prove this lemma, we exploit the security of the used double-encryption in a similar way as in the IND-CCA proof of Naor and Yung [35]. Particularly, when using an obfuscation of the circuit C''_{Add}, it is not necessary to know both decryption keys to produce public parameters for the group. We recall that the circuit C_{Zero} only knows the decryption key to decrypt the first component of encodings. Furthermore, it is possible to produce a consistency proof without knowing the content of the ciphertexts C and C' by simply simulating it in the same way C''_{Add} does. Therefore, we can reduce to the IND-CPA security of the encryption scheme. In order to apply the same argument for the first component of encodings, we need the circuit C_{Zero} to forget about the first decryption key. We accomplish that by replacing the obfuscation of C_{Zero} with an obfuscation of the circuit $\overline{C}_{\mathsf{Zero}}$ that uses only the second decryption key instead of the first one.

[3] We note that there are probabilistic polynomial time algorithms that factor univariate polynomials over finite fields, for instance the Cantor-Zassenhaus algorithm [18].

This is possible due to the security of the obfuscator and the soundness of the proof system. Then, we can use the same argument as above to reduce to the IND-CPA security of the encryption scheme.

Obtaining the *Interactive Uber assumption* in our group. We recall that the Interactive Uber assumption (in one variable) generates one secret point $s \in \mathbb{Z}_q$ uniformly at random at which all queried polynomials are evaluated. To show that the Interactive Uber assumption holds in our group, we first set up that secret point s as $c \cdot w$ for some independent random c from $\mathbb{Z}_q^\times$, where w is the secret value of our group introduced above. Hence, a polynomial P that is evaluated at $s = c \cdot w$ can be interpreted as a (different) polynomial in w. Particularly, given a polynomial $P(X)$, the polynomial $\overline{P}(X) := P(c \cdot X)$ satisfies the equation $P(s) = \overline{P}(w)$. Thus, an encoding that contains the polynomial $\overline{P}(X)$ determines the exponent of the represented group element to equal $\overline{P}(w) = P(c \cdot w) = P(s)$. This observation paves the way for using higher-degree polynomials $\overline{P}(X)$ to produce encodings for oracle answers and the challenge. As the resulting group elements (i.e. the corresponding exponents) remain the same, the "switching lemma" described above justifies that this modification is unnoticeable. Furthermore, by a similar argument as above, we simulate the proofs of consistency π for every produced encoding, in particular for the encodings that are produced by the addition circuit.[4] As the consistency proof can now be produced independently of the basis $\{a_1, \ldots, a_d\}$, we are able to unnoticeably "erase" this basis from the commitment in the public parameters.

Our goal now is to alter the structure of the group in the following sense. By definition, our group is isomorphic to the additive group $\mathbb{Z}_q$. We aim to alter that structure such that our group is isomorphic to the additive group of polynomials in $\mathbb{Z}_q[X]$ (of bounded degree). Particularly, we alter the equality relation that is defined on the set of encodings such that two encodings are considered equal only if the thereby defined polynomials are equal as abstract polynomials. For that purpose, we remove the dependency on the point w by altering the point function obfuscation po such that it maps all inputs to 0. Therefore, the zero testing circuit C_{Zero} only treats an encoding that contains the zero polynomial as an encoding of the identity element of the group. As the value w is never used explicitly in the game (as all the proofs of consistency are simulated), this modification is unnoticeable due to the security property of the point function obfuscation po. This is a crucial step paving the way for employing arguments from linear algebra to enable randomization.

The final step requires to randomize the challenge encoding such that there is no detectable difference between a real challenge and a randomly sampled one. First, we recall that encodings do not encrypt polynomials in the plain. The encodings contain the representation of polynomials with respect to some basis $\{a_1, \ldots, a_d\}$. That is, given a polynomial $P(X)$, the encoding corresponding to $g^{P(s)}$ encrypts the vectors

[4] More precisely, we again use an obfuscation of C''_{Add} instead of an obfuscation of C_{Add} as described above.

$$\vec{f} = \vec{f'} = (a_1 \mid \ldots \mid a_d) \cdot \underbrace{P(c \cdot X)}_{=\overline{P}(X)}, \tag{5}$$

where $P(c \cdot X)$ is interpreted as a vector of coefficients in the natural way. Therefore, the only information about the matrix $(a_1 | \ldots | a_d)$ is given by matrix vector products. To avoid trivialities, the challenge polynomial P^* can be assumed not to lie in the span of the queries $P_1, \ldots, P_l$, which is why $P^*(c \cdot X)$ does not lie in the span of $P_1(c \cdot X), \ldots, P_l(c \cdot X)$. Hence, we may resort to an information-theoretic argument. More precisely, an adversary that is able to adaptively ask for matrix vector products, information-theoretically learns nothing about matrix vector products that are linearly independent of its queries. Therefore, the polynomial that is contained in the real challenge encoding information-theoretically looks like a randomly sampled polynomial (with bounded degree) given that the matrix $(a_1 | \ldots | a_d)$ is uniformly distributed.

Obtaining the *multivariate* Interactive Uber assumption. The main difficulty that arises from generalizing our results to the multivariate Interactive Uber assumption is that we do not have a polynomial-time algorithm that computes all zeros of a multivariate polynomial. Hence, the zero testing circuit C_{Zero} needs to know the point $\boldsymbol{\omega} := (\omega_1, \ldots, \omega_m) \in \mathbb{Z}_q^m$ in the clear to explicitly evaluate the polynomial f that is defined by a given encoding. Our previous proof strategy, however, crucially relies on removing the dependency on w such that C_{Zero} only treats encodings containing the zero polynomial as encodings of the identity element. This is equivalent to altering the group structure such that it is isomorphic to the additive group of polynomials over $\mathbb{Z}_q$ (of bounded degree).

Although the zero testing circuit C_{Zero} knows $\boldsymbol{\omega}$ in the clear, it is nevertheless possible to pursue a similar strategy. Our solution is to gradually alter C_{Zero} such that it "forgets" the components ω_i of $\boldsymbol{\omega}$ one by one. Particularly, we define intermediate circuits $C_{\mathsf{Zero}}^{(i)}$ that test if the polynomial

$$F_i^{(f)}(X_1, \ldots, X_i) := f(X_1, \ldots, X_i, \omega_{i+1}, \ldots, \omega_m) \tag{6}$$

equals the zero polynomial in $\mathbb{Z}_q[X_1, \ldots, X_i]$. Observe that the original circuit C_{Zero} tests whether $F_0^{(f)} \equiv 0$. Our goal is to unnoticeably establish $C_{\mathsf{Zero}}^{(m)}$ as zero testing circuit, as it realizes the stricter equality relation we aim for.

In order to unnoticeably replace an obfuscation of $C_{\mathsf{Zero}}^{(i)}$ with an obfuscation of $C_{\mathsf{Zero}}^{(i+1)}$, we first alter the implementation of $C_{\mathsf{Zero}}^{(i)}$ such that it performs the test whether $F_i^{(f)}$ is the zero polynomial by evaluating it at a randomly sampled point $\boldsymbol{r} \in \mathbb{Z}_q^i$. Applying the Schwartz-Zippel lemma upper bounds the statistical distance of the output distributions of the two circuits enabling to reduce this step to the security of the obfuscator.

Furthermore, the condition that $F_i^{(f)}(\boldsymbol{r}) = F_{i+1}^{(f)}(\boldsymbol{r}, \omega_{i+1}) = 0$ is equivalent to the condition that the univariate polynomial $F_{i+1}^{(f)}(\boldsymbol{r}, X_{i+1})$ is zero at the point ω_{i+1}. This can be implemented in a similar manner as in the univariate case using a point function obfuscation of ω_{i+1}. In addition, this circuit contains a

conceptional logical or statement testing whether the polynomial $F_{i+1}^{(f)}(\boldsymbol{r}, X_{i+1})$ equals the zero polynomial. Using a similar argument as above we are able to alter the point function obfuscation for ω_{i+1} to a point function obfuscation that never triggers.

Hence, our zero testing circuit effectively only tests whether $F_{i+1}^{(f)}(\boldsymbol{r}, X_{i+1})$ equals the zero polynomial in $\mathbb{Z}_q[X_{i+1}]$. Applying the Schwartz-Zippel lemma again, we are able to unnoticeably alter the implementation of the zero testing circuit such that it tests whether $F_{i+1}^{(f)}$ equals the zero polynomial in $X_1, \ldots, X_{i+1}$ concluding the argument.

Roadmap. After fixing notation and recalling some basic definitions in Sect. 2, we present our main group construction in Sect. 3. Our main theorem, Theorem 1, states the validity of (our variant of) the Interactive Uber assumption relative to the group construction from Sect. 3. For the detailed proofs we refer the reader to the full version [3].

2 Preliminaries

2.1 Notation

For $n \in \mathbb{N}$, let 1^n denote the string consisting of n times the digit 1. For a probabilistic algorithm A, let $y \leftarrow A(x)$ denote that y is the output of A on input x. The randomness which A uses during the computation can be made explicit by $y \leftarrow A(x; r)$, where r denotes the randomness. Let λ denote the security parameter. We assume that the security parameter is implicitly given to all algorithms as 1^λ.

Let $\mathcal{G}$ be a group and let h be a fixed generator of $\mathcal{G}$. Then, $[n]$ denotes the group element h^n.

Let $n \in \mathbb{N}$ be a number, let $\mathbb{K}$ be a field, and let $\mathbb{K}^n$ denote the vector space of n-tuples of elements of $\mathbb{K}$. Further, for any $i \in \{1, \ldots, n\}$, let $e_i \in \mathbb{K}^n$ be the vector such that the i-th entry of e_i equals 1 and any remaining entry equals 0. Then, the set $\{e_1, e_2, \ldots, e_n\}$ denotes the *standard basis* of $\mathbb{K}^n$. Let $b_1, \ldots, b_i \in \mathbb{K}^n$, then $\langle b_1, \ldots, b_i \rangle \subseteq \mathbb{K}^n$ denotes the span of those vectors.

2.2 Assumptions

Let $(\mathcal{G}_\lambda)_{\lambda \in \mathbb{N}}$ be a family of finite cyclic groups. If it is clear from the context, we write $\mathcal{G}$ instead of $\mathcal{G}_\lambda$. We assume that the order $q := |*|\mathcal{G}$ of the group is known and *prime*. Let $\mathsf{Gens}_\mathcal{G}$ be the set of generators of $\mathcal{G}$. We assume that we can efficiently sample elements uniformly at random from $\mathsf{Gens}_\mathcal{G}$.

A very basic and well-established cryptographic assumption is the decisional Diffie-Hellman (DDH) assumption. The DDH assumption states that the distributions $([x], [y], [x \cdot y])$ and $([x], [y], [z])$ are computationally indistinguishable for $x, y, z \leftarrow \mathbb{Z}_q$.

Definition 1 (Decisional Diffie-Hellman (DDH) assumption). *For any PPT adversary $\mathcal{A}$, the advantage $Adv^{ddh}_{\mathcal{G},\mathcal{A}}(\lambda)$ is negligible in λ, where*

$$\begin{aligned} Adv^{ddh}_{\mathcal{G},\mathcal{A}}(\lambda) := &\Pr\left[\mathcal{A}(1^\lambda, [x], [y], [x \cdot y]) = 1 | x, y \leftarrow \mathbb{Z}_q\right] \\ &- \Pr\left[\mathcal{A}(1^\lambda, [x], [y], [z]) = 1 | x, y, z \leftarrow \mathbb{Z}_q\right] \end{aligned}$$

and q is the order of the group $\mathcal{G}$.

Let $(\mathcal{G}_1, \mathcal{G}_2, e)$ be finite cyclic groups of prime order $|*|\mathcal{G}_1 = |*|\mathcal{G}_2$ and let $e\colon \mathcal{G}_1 \times \mathcal{G}_2 \to \mathcal{G}_T$ be a pairing (i.e. a non-degenerate and bilinear map). The groups $\mathcal{G}_1$, $\mathcal{G}_2$, $\mathcal{G}_T$, as well as the pairing e depend on the security parameter. For greater clarity, we omit this dependency in this setting.

A natural extension of the DDH assumption to the bilinear setting is the symmetric external Diffie-Hellman (SXDH) assumption. The SXDH assumption states that the DDH assumption holds in both groups $\mathcal{G}_1$ and $\mathcal{G}_2$.

2.3 Point Obfuscation

In our construction we employ a cryptographic primitive that is called *point obfuscation* [16,37]. A point obfuscation serves the purpose to hide a certain point, but to enable a test whether a given value is hidden inside. Equivalently, this notion can be seen as an "obfuscation" of a point-function that evaluates to 1 at exactly this given point and to 0 everywhere else. We require that it is infeasible to distinguish a point obfuscation that triggers at a randomly sampled point from a point obfuscation that never triggers. This security requirement is rather weak compared to similar notions [7].

Definition 2 (Point obfuscation). *A* point obfuscation *for message space $\mathcal{M}_\lambda$ is a PPT algorithm* $\textsc{POBF}$.

$\textsc{POBF}(1^\lambda, x) \to \mathsf{po}$ *On input a message $x \in \mathcal{M}_\lambda \cup \{\bot\}$,* $\textsc{POBF}$ *produces a description of the point function*

$$\mathsf{po}\colon \mathcal{M}_\lambda \to \{0,1\}, y \mapsto \begin{cases} 1 & \text{if } y = x \\ 0 & \text{otherwise} \end{cases}.$$

We require the following two properties to hold:

Correctness: *For any x, $y \in \mathcal{M}_\lambda$ and any $\mathsf{po} \leftarrow \textsc{POBF}(1^\lambda, x)$, $\mathsf{po}(y) \mapsto 1$ if and only if $x = y$.*

Soundness: *For any PPT adversary $\mathcal{A}$, the advantage $Adv^{po}_{\textsc{POBF},\mathcal{A}}(\lambda)$ is negligible in λ, where*

$$\begin{aligned} Adv^{po}_{\textsc{POBF},\mathcal{A}}(\lambda) := &\Pr\left[\mathcal{A}(1^\lambda, \mathsf{po}) = 1 | \mathsf{po} \leftarrow \textsc{POBF}(1^\lambda, x), x \leftarrow \mathcal{M}_\lambda\right] \\ &- \Pr\left[\mathcal{A}(1^\lambda, \mathsf{po}) = 1 | \mathsf{po} \leftarrow \textsc{POBF}(1^\lambda, \bot)\right]. \end{aligned}$$

An adaption of a construction proposed in [16] yields a point obfuscation POBF with message space $\mathbb{Z}_p$ based on the DDH assumption. Furthermore, a point obfuscation with message space $\mathbb{Z}_p$ can be used to construct a point obfuscation for message space $\mathbb{Z}_q$, where q is a prime such that $\frac{p}{q}$ is negligible in λ. For further details, we refer the reader to the full version [3].

Remark 1. According to a reviewer of TCC 2017, a point obfuscation with message space $\{0,1\}^{\mathsf{poly}(\lambda)}$ can be constructed from an injective one-way function F together with a corresponding hardcore bit B.

Given a string x, the tuple $(F(x), B(x))$ is the obfuscation of x. The tuple $(F(y), 1 - B(y))$ is an obfuscation of $\perp$, where y is a random element from the message space.

2.4 Subset Membership Problems

The notion of subset membership problems was introduced in [22]. Informally, a hard subset membership problem specifies a set, such that it is intractable to decide whether a value is inside this set or not. Let $\mathcal{L} = (\mathcal{L}_\lambda)_{\lambda\in\mathbb{N}}$ be a family of families of languages $L \subseteq \mathcal{X}_\lambda$ in a universe $\mathcal{X}_\lambda = \mathcal{X}$. Further, let $\mathcal{R}$ be an efficiently computable witness relation, such that $x \in L$ if and only if there exists a witness $w \in \{0,1\}^{\mathsf{poly}(|x|)}$ with $\mathcal{R}(x, w) = 1$, where poly is a fixed polynomial. We assume that we are able to efficiently and uniformly sample elements from L together with a corresponding witness, and that we are able to efficiently and uniformly sample elements from $\mathcal{X} \setminus L$.

Definition 3 (Hard subset membership problem). *The* subset membership problem *(SMP) $L \subseteq \mathcal{X}$ is hard, if for any PPT adversary $\mathcal{A}$, the advantage*

$$Adv^{smp}_{\mathcal{L},\mathcal{A}}(\lambda) := \Pr\left[\mathcal{A}(1^\lambda, x) = 1 | x \leftarrow L\right] - \Pr\left[\mathcal{A}(1^\lambda, x) = 1 | x \leftarrow \mathcal{X} \setminus L\right]$$

is negligible in λ.

For our construction we need a family $\mathcal{L} = (\mathcal{L}_\lambda)_{\lambda\in\mathbb{N}}$ such that for any $L \in \mathcal{L}_\lambda$ and any $x \in L$, there exists exactly one witness $r \in \{0,1\}^*$ with $\mathcal{R}(x, w) = 1$.

Let $\mathcal{G} = \{\mathcal{G}_\lambda\}$ be a family of finite cyclic groups of prime order such that the DDH assumption holds. A possible instantiation of a hard SMP meeting our requirements is the Diffie-Hellman language $\mathcal{L}^{\mathrm{dh}} := (\mathcal{L}^{\mathrm{dh}}_\lambda)_{\lambda\in\mathbb{N}}$. For any $\lambda \in \mathbb{N}$, $\mathcal{L}^{\mathrm{dh}}_\lambda := \{L_{g,h} \,|\, g, h \in \mathsf{Gens}_\mathcal{G}\}$, $\mathcal{X}_\lambda = \mathsf{Gens}_\mathcal{G} \times \mathsf{Gens}_\mathcal{G}$, and $L_{g,h} := \{(g^r, h^r) \,|\, r \in \mathbb{Z}_q\}$, where $q = |*|\mathcal{G}_k$. The SMP $L_{g,h} \subseteq \mathcal{X}$ is hard for randomly chosen generators $g, h \leftarrow \mathsf{Gens}_\mathcal{G}$. Given $(g^r, h^r) \in L_{g,h}$, the corresponding unique witness is $r \in \mathbb{Z}_q$.

2.5 Non-interactive Commitments

Non-interactive commitment schemes are a commonly used cryptographic primitive [29]. They enable to commit to a chosen value without revealing this value. Additionally, once committed to a value, this value cannot be changed. In contrast to the notion of point obfuscations, a commitment scheme prevents to test whether a particular value is hidden inside a commitment.

Definition 4 (Perfectly binding non-interactive commitment scheme (syntax and security)). *A* perfectly binding non-interactive commitment scheme *for message space* $\mathcal{M}_\lambda$ *is a triple of PPT algorithms* $\textsc{Com} = (\textsc{ComSetup}, \textsc{Commit}, \textsc{Open})$.

$\textsc{ComSetup}(1^\lambda) \to ck$ *On input the unary encoded security parameter, the algorithm* ComSetup *outputs a commitment key* ck.

$\textsc{Commit}_{ck}(m) \to (com, op)$ *On input the commitment key* ck *and a message* $m \in \mathcal{M}_\lambda$, Commit *outputs a tuple* (com, op).

$\textsc{Open}_{ck}(com, op) \to \widetilde{m}$ *On input the commitment key* ck *and a commitment-opening pair* (com, op), Open *outputs the committed message* m *if* op *is a valid opening for* com. *Otherwise,* Open *outputs* $\perp$.

We require Com *to be* perfectly correct, perfectly binding, *and* computationally hiding.

Correctness Com *is* correct *if for any* $\lambda \in \mathbb{N}$, *any* $ck \leftarrow \textsc{ComSetup}(1^\lambda)$, *and any* $m \in \mathcal{M}_\lambda$, $\textsc{Open}_{ck}(\textsc{Commit}_{ck}(m)) = m$.

Perfectly binding Com *is* perfectly binding *if it is not possible to find a commitment that has valid openings for more than one message, i.e. for any (possibly unbounded) adversary* $\mathcal{A}$, $Adv^{binding}_{\textsc{Com},\mathcal{A}}(\lambda) = 0$, *where*

$$Adv^{binding}_{\textsc{Com},\mathcal{A}}(\lambda) := \Pr\left[Exp^{binding}_{\textsc{Com},\mathcal{A}}(\lambda) = 1\right].$$

Computationally hiding Com *is* computationally hiding *if commitments for different messages are computationally indistinguishable, i.e. for any PPT adversary* $\mathcal{A}$, $Adv^{hiding}_{\mathcal{A}}(\lambda)$ *is negligible, where*

$$Adv^{hiding}_{\textsc{Com},\mathcal{A}}(\lambda) := \Pr\left[Exp^{hiding}_{\textsc{Com},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}.$$

The games $Exp^{binding}_{\textsc{Com},\mathcal{A}}(\lambda)$ *and* $Exp^{hiding}_{\textsc{Com},\mathcal{A}}(\lambda)$ *are defined in Fig. 1.*

Such a commitment scheme can be obtained from a group in which the DDH assumption holds.

Experiment $Exp^{\text{binding}}_{\textsc{Com},\mathcal{A}}(\lambda)$
$ck \leftarrow \textsc{ComSetup}(1^\lambda)$
$(c, o_1, o_2) \leftarrow \mathcal{A}(1^\lambda, ck)$
$m_1 \leftarrow \textsc{Open}_{ck}(c, o_1)$, $m_2 \leftarrow \textsc{Open}_{ck}(c, o_2)$
if $m_1 \neq \perp \wedge m_2 \neq \perp \wedge m_1 \neq m_2$ **then**
return 1
return 0

Experiment $Exp^{\text{hiding}}_{\textsc{Com},\mathcal{A}}(\lambda)$
$ck \leftarrow \textsc{ComSetup}(1^\lambda)$
$(m_0, m_1, st) \leftarrow \mathcal{A}(1^\lambda, ck, \mathsf{find})$
$b \leftarrow \{0,1\}$, $(c, o) \leftarrow \textsc{Commit}_{ck}(m_b)$
$b' \leftarrow \mathcal{A}(1^\lambda, c, st, \mathsf{attack})$
if $b = b'$ **then return** 1
return 0

Fig. 1. The description of the Binding game $Exp^{\text{binding}}_{\textsc{Com},\mathcal{A}}(\lambda)$ (left) and the Hiding game $Exp^{\text{hiding}}_{\textsc{Com},\mathcal{A}}(\lambda)$ (right).

2.6 Dual Mode NIWI Proof System

The notion of dual mode NIWI proof systems abstracts from the NIWI proof system proposed in [30]. A similar abstraction was used in [4].

Definition 5 (Dual mode NIWI proof system (syntax and security)). *A* dual mode non-interactive witness-indistinguishable (NIWI) proof system *for a relation $\mathcal{R}$ is a tuple of PPT algorithms $\Pi = (\mathsf{Setup}_\Pi, \mathsf{K}, \mathsf{S}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Extract})$.*

- $\mathsf{Setup}_\Pi(1^\lambda) \to (gpk, gsk)$ *On input the unary encoded security parameter,* Setup_Π *outputs a group key gpk and, additionally, may output some related information gsk. The relation $\mathcal{R}$ is an efficiently computable ternary relation consisting of triplets of the form (gpk, x, w) and defines a group-dependent language L. The language L consists of the statements x, such that there exists a witness w with $(gpk, x, w) \in \mathcal{R}$.*
- $\mathsf{K}(gpk, gsk) \to (crs, td_{ext})$ *On input the group keys gpk and gsk,* K *outputs a* binding *common reference string (CRS) crs and a corresponding extraction trapdoor td_{ext}.*
- $\mathsf{S}(gpk, gsk) \to (crs, \perp)$ *On input the group keys gpk and gsk,* S *outputs a* hiding *CRS crs.*
- $\mathsf{Prove}(gpk, crs, x, w) \to \pi$ *On input the public group key gpk, the CRS crs, a statement x, and a corresponding witness w,* Prove *produces a proof π.*
- $\mathsf{Verify}(gpk, crs, x, \pi) \to \{0, 1\}$ *On input the public group key gpk, the CRS crs, a statement x, and a proof π,* Verify *outputs 1 if the proof is valid and 0 if the proof is rejected.*
- $\mathsf{Extract}(td_{ext}, x, \pi) \to w$ *On input the extraction trapdoor td_{ext}, a statement x, and a proof π,* $\mathsf{Extract}$ *outputs a witness w.*

We require Π to meet the following requirements:

CRS indistinguishability. *Common reference strings generated via* $\mathsf{K}(gpk, gsk)$ *and* $\mathsf{S}(gpk, gsk)$ *are computationally indistinguishable, i.e.*

$$Adv^{crs}_{\Pi,\mathcal{A}}(\lambda) := \Pr\left[Exp^{crs}_{\Pi,\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}$$

is negligible in λ, where $Exp^{crs}_{\Pi,\mathcal{A}}(\lambda)$ is defined as in Fig. 2.

Perfect completeness under K and S. *For any $\lambda \in \mathbb{N}$, any $(gpk, gsk) \leftarrow \mathsf{Setup}_\Pi(1^\lambda)$, any CRS $(crs, \cdot) \leftarrow \mathsf{K}(gpk, gsk)$, any (x, w) such that $(gpk, x, w) \in \mathcal{R}$, and any $\pi \leftarrow \mathsf{Prove}(gpk, crs, x, w)$, $\mathsf{Verify}(gpk, crs, x, \pi) \to 1$. The same holds for any $(crs, \cdot) \leftarrow \mathsf{S}(gpk, gsk)$.*

Perfect soundness under K. *For any $\lambda \in \mathbb{N}$, any $(gpk, gsk) \leftarrow \mathsf{Setup}_\Pi(1^\lambda)$, any $(crs, \cdot) \leftarrow \mathsf{K}(gpk, gsk)$, any statement x such that there exists no witness w with $(gpk, x, w) \in \mathcal{R}$, and any $\pi \in \{0, 1\}^*$, $\mathsf{Verify}(gpk, crs, x, \pi) \to 0$.*

Perfect extractability under K. *For any $\lambda \in \mathbb{N}$, any key pair $(gpk, gsk) \leftarrow \mathsf{Setup}_\Pi(1^\lambda)$, any $(crs, td_{ext}) \leftarrow \mathsf{K}(gpk, gsk)$, any (x, π) such that $\mathsf{Verify}(gpk, crs, x, \pi) \to 1$, and for any $w \leftarrow \mathsf{Extract}(td_{ext}, x, \pi)$, w is a satisfying witness for the statement x, i.e. $(gpk, x, w) \in \mathcal{R}$.*

$\underline{\text{Experiment } Exp^{\text{crs}}_{\Pi,\mathcal{A}}(\lambda)}$

$(gpk, gsk) \leftarrow \mathsf{Setup}_{\Pi}(1^{\lambda})$
$(crs_0, \cdot) \leftarrow \mathsf{K}(gpk, gsk)$, $(crs_1, \cdot) \leftarrow \mathsf{S}(gpk, gsk)$
$b \leftarrow \{0,1\}$, $b' \leftarrow \mathcal{A}(1^{\lambda}, gpk, crs_b)$
if $b' = b$ **then return** 1
return 0

Fig. 2. The description of the CRS inistinguishability game $Exp^{\text{crs}}_{\Pi,\mathcal{A}}(\lambda)$.

Perfect witness-indistinguishability under S. *For any $\lambda \in \mathbb{N}$, any $(gpk, gsk) \leftarrow \mathsf{Setup}_{\Pi}(1^{\lambda})$, any $(crs, \cdot) \leftarrow \mathsf{S}(gpk, gsk)$, any (x, w_0) and (x, w_1) with $(gpk, x, w_0), (gpk, x, w_1) \in \mathcal{R}$, the output of $\mathsf{Prove}(gpk, crs, x, w_0)$ and the output of $\mathsf{Prove}(gpk, crs, x, w_1)$ are identically distributed.*

An exemplary dual mode NIWI proof system satisfying computational CRS indistinguishability, perfect completeness, perfect soundness, perfect extractability, and perfect witness-indistinguishability is the proof system proposed by Groth and Sahai in [30]. The soundness, in particular the indistinguishability of common reference strings, of this construction can for instance be based on the SXDH assumption. The Groth-Sahai proof system allows perfect extractability for group elements, however, does not provide a natural way to extract scalars. Nevertheless, perfect extractability can be achieved by using the proof system for the bit representation of the particular scalars [34].

2.7 Probabilistic Indistinguishability Obfuscation

The notion of probabilistic circuit obfuscation was proposed in [17]. Informally, probabilistic circuit obfuscation enables to conceal the implementation of probabilistic circuits while preserving their functionality. Let $\mathcal{C} = (\mathcal{C}_\lambda)_{\lambda \in \mathbb{N}}$ be a family of sets $\mathcal{C}_\lambda$ of probabilistic circuits. The set $\mathcal{C}_\lambda$ contains circuits of polynomial size in λ. A *circuit sampler* for $\mathcal{C}$ is defined as a set of (efficiently samplable) distributions $S = (S_\lambda)_{\lambda \in \mathbb{N}}$, where S_λ is a distribution over triplets (C_0, C_1, z) with $C_0, C_1 \in \mathcal{C}_\lambda$ such that C_0 and C_1 take inputs of the same length and $z \in \{0,1\}^{\mathsf{poly}(\lambda)}$.

Definition 6 (Probabilistic indistinguishability obfuscation for a class of samplers $\mathcal{S}$, [4,17]). *A* probabilistic indistinguishability obfuscator *(pIO) for a class of samplers $\mathcal{S}$ over the probabilistic circuit family $\mathcal{C} = (\mathcal{C}_\lambda)_{\lambda \in \mathbb{N}}$ is a uniform PPT algorithm $pi\mathcal{O}$, such that the following properties hold:*

Correctness. *On input the unary encoded security parameter 1^{λ} and a circuit $C \in \mathcal{C}_\lambda$, $pi\mathcal{O}$ outputs a deterministic circuit Λ of polynomial size in $|*|C$ and λ. For any $\lambda \in \mathbb{N}$, any $C \in \mathcal{C}_\lambda$, any $\Lambda \leftarrow pi\mathcal{O}(1^{\lambda}, C)$, and any inputs $m \in \{0,1\}^*$ (of matching length), there exists a randomness r, such that $C(m; r) = \Lambda(m)$.*

Experiment $Exp^{\text{pio-c}}_{C,z,\mathcal{D}}(\lambda)$	Experiment $Exp^{\text{pio-ind}}_{pi\mathcal{O},S,\mathcal{A}}(\lambda)$	Experiment $Exp^{\text{sel-ind}}_{S,\mathcal{A}}(\lambda)$
$C_0 := C$	$(C_0, C_1, z) \leftarrow S_\lambda$	$(x, st) \leftarrow \mathcal{A}_1(1^\lambda)$
$C_1 := pi\mathcal{O}(1^\lambda, C)$	$b \leftarrow \{0,1\}$	$(C_0, C_1, z) \leftarrow S_\lambda$, $b \leftarrow \{0,1\}$
$b \leftarrow \{0,1\}$	$\Lambda \leftarrow pi\mathcal{O}(1^\lambda, C_b)$	$y \leftarrow C_b(x; r)$ // for fresh randomness r
$b' \leftarrow \mathcal{A}^{C_b(\cdot)}(1^\lambda, C, z)$	$b' \leftarrow \mathcal{A}(1^\lambda, C_0, C_1, \Lambda, z)$	$b' \leftarrow \mathcal{A}_2(1^\lambda, C_0, C_1, z, y, st)$
if $b' = b$ **then return** 1	**if** $b' = b$ **then return** 1	**if** $b' = b$ **then return** 1
return 0	**return** 0	**return** 0

Fig. 3. The descriptions of the games $Exp^{\text{pio-c}}_{C,z,\mathcal{D}}(\lambda)$ (left), $Exp^{\text{pio-ind}}_{pi\mathcal{O},S,\mathcal{A}}(\lambda)$ (middle), and $Exp^{\text{sel-ind}}_{S,\mathcal{A}}(\lambda)$ (right). In $Exp^{\text{pio-c}}_{C,z,\mathcal{D}}(\lambda)$, $\mathcal{D}$ has oracle access to either a probabilistic circuit C_0 using fresh randomness for every oracle query or to a deterministic circuit C_1. $\mathcal{D}$ can make an unbounded number of oracle queries with the restriction that no input is queried twice.

Furthermore, for every non-uniform PPT distinguisher $\mathcal{D}$, every $\lambda \in \mathbb{N}$, every $C \in \mathcal{C}_\lambda$, and every auxiliary input $z \in \{0,1\}^{poly(\lambda)}$, the advantage

$$Adv^{pio\text{-}c}_{C,z,\mathcal{D}}(\lambda) := \Pr\left[Exp^{pio\text{-}c}_{C,z,\mathcal{D}}(\lambda) = 1\right] - \frac{1}{2}$$

is negligible in λ, where $Exp^{pio\text{-}c}_{C,z,\mathcal{D}}(\lambda)$ is defined as in Fig. 3.

Security with respect to S. *For any circuit sampler $S = \{S_\lambda\}_{\lambda\in\mathbb{N}}$, for any non-uniform PPT adversary $\mathcal{A}$, the advantage*

$$Adv^{pio\text{-}ind}_{pi\mathcal{O},S,\mathcal{A}}(\lambda) := \Pr\left[Exp^{pio\text{-}ind}_{pi\mathcal{O},S\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}$$

is negligible in λ, where $Exp^{pio\text{-}ind}_{pi\mathcal{O},S\mathcal{A}}(\lambda)$ is defined as in Fig. 3.

We remark that the construction proposed in [17] also satisfies our definition of correctness.

Let $X\colon \mathbb{N} \to \mathbb{N}$ be a function. For our purposes we use a class of circuit samplers, such that the sampled circuits are functionally equivalent for all inputs outside of a set $\mathcal{X}$, and the outputs of the circuits are indistinguishable for inputs inside of this set $\mathcal{X}$. The set $\mathcal{X}$ is a subset of the circuits' domain of cardinality at most $X(\lambda)$. Two circuits C_0 and C_1 are functionally equivalent if for any input x of matching length and any randomness r, $C_0(x; r) = C_1(x; r)$.

Definition 7 (X-Ind sampler, [4,17]). *Let $X\colon \mathbb{N} \to \mathbb{N}$ be a function with $X(\lambda) \leq 2^\lambda$, for all $\lambda \in \mathbb{N}$. The class $\mathcal{S}^{X\text{-}ind}$ of* X-Ind samplers *for a circuit family $\mathcal{C}$ contains all circuit samplers S for $\mathcal{C}$ satisfying, that for any $\lambda \in \mathbb{N}$, there exists a set $\mathcal{X} = \mathcal{X}_\lambda \subseteq \{0,1\}^*$ with $|*|_{\mathcal{X}} \leq X(\lambda)$, such that the following two properties hold:*

***X*-differing inputs.** *For any (possibly unbounded) deterministic adversary $\mathcal{A}$, the advantage*

$$Adv^{eq\$}_{S,\mathcal{A}}(\lambda) := \Pr\left[C_0(x;r) \neq C_1(x;r) \wedge x \notin \mathcal{X} \;\middle|\; \begin{matrix}(C_0, C_1, z) \leftarrow S_\lambda,\\ (x, r) \leftarrow \mathcal{A}(C_0, C_1, z)\end{matrix}\right]$$

is negligible in λ.

***X*-indistinguishability.** *For any non-uniform PPT distinguisher $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, the advantage*

$$X(\lambda) \cdot Adv^{sel\text{-}ind}_{S,\mathcal{A}}(\lambda) := X(\lambda) \cdot \left(\Pr\left[Exp^{sel\text{-}ind}_{S,\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}\right)$$

is negligible in λ, where $Exp^{sel\text{-}ind}_{S,\mathcal{A}}(\lambda)$ is defined as in Fig. 3.

For our construction we use an obfuscator for the class $\mathcal{S}^{X\text{-ind}}$.

According to Theorem 2 in the proceedings of [17], a pIO which is secure with respect to $\mathcal{S}^{X\text{-ind}}$ for a circuit family $\mathcal{C}$ that only contains circuits of size at most λ can be obtained from sub-exponentially secure indistinguishability obfuscation (IO) for deterministic circuits in conjunction with sub-exponentially secure puncturable PRF. The construction given in [17] satisfies this security requirement even if the circuit family $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ contains circuits with polynomial size in λ as long as the input length of those circuits is at most λ.

2.8 Fully Homomorphic Encryption Scheme

Let $\mathcal{C} = (\mathcal{C}_\lambda)_{\lambda\in\mathbb{N}}$ be a family of sets of polynomial sized circuits of arity $a(\lambda)$, i.e. the set $\mathcal{C}_\lambda$ contains circuits of polynomial size in λ. We assume that for any $\lambda \in \mathbb{N}$ the circuits in $\mathcal{C}_\lambda$ share the common input domain $(\{0,1\}^{\mathsf{poly}(\lambda)})^{a(\lambda)}$ for a fixed polynomial $\mathsf{poly}(\lambda)$. A homomorphic encryption scheme enables evaluation of circuits on encrypted data. The first fully homomorphic encryption scheme was proposed in [28]. In this paper, we abide by the notation used in [4].

Definition 8 (Homomorphic public-key encryption (HPKE) scheme (syntax and security)). *A* homomorphic public-key encryption scheme *with message space $\mathcal{M} \subseteq \{0,1\}^*$ for a deterministic circuit family $\mathcal{C} = (\mathcal{C}_\lambda)_{\lambda\in\mathbb{N}}$ of arity $a(\lambda)$ and input domain $(\{0,1\}^{\mathsf{poly}(\lambda)})^{a(\lambda)}$ is a tuple of PPT algorithms* $\textsc{Hpke} = (\textsc{Gen}, \textsc{Enc}, \textsc{Dec}, \textsc{Eval})$.

$\textsc{Gen}(1^\lambda) \to (pk, sk)$ *On input the unary encoded security parameter 1^λ,* Gen *outputs a public key pk and a secret key sk.*

$\textsc{Enc}(pk, m) \to c$ *On input the public key pk and a message $m \in \mathcal{M}$,* Enc *outputs a ciphertext $c \in \{0,1\}^{\mathsf{poly}(\lambda)}$ for message m.*

$\textsc{Dec}(sk, c) \to m$ *On input the secret key sk and a ciphertext $c \in \{0,1\}^{\mathsf{poly}(\lambda)}$,* Dec *outputs the corresponding message $m \in \mathcal{M}$ (or $\perp$, if the ciphertext is not valid).*

$\textsc{Experiment}\ Exp^{\text{ind-cpa}}_{\text{Hpke},\mathcal{A}}(\lambda)$

$(pk, sk) \leftarrow \textsc{Gen}(1^\lambda)$, $(m_0, m_1, st) \leftarrow \mathcal{A}(1^\lambda, pk, \mathsf{find})$
$b \leftarrow \{0,1\}$, $c \leftarrow \textsc{Enc}(pk, m_b)$
$b' \leftarrow \mathcal{A}(1^\lambda, c, st, \mathsf{attack})$
if $b' = b$ **then return** 1
return 0

Fig. 4. The description of the IND-CPA game $Exp^{\text{ind-cpa}}_{\text{Hpke},\mathcal{A}}(\lambda)$.

$\textsc{Eval}(pk, C, c_1, \ldots, c_{a(\lambda)}) \to c$ *On input the public key* pk*, a deterministic circuit* $C \in \mathcal{C}_\lambda$*, and ciphertexts* $(c_1, \ldots, c_{a(\lambda)}) \in (\{0,1\}^{\mathsf{poly}(\lambda)})^{a(\lambda)}$*,* Eval *outputs a ciphertext* $c \in \{0,1\}^{\mathsf{poly}(\lambda)}$*.*

We require Hpke *to meet the following requirements:*

Perfect correctness. *The triple* (Gen, Enc, Dec) *is perfectly correct as a PKE scheme, i.e. for any* $\lambda \in \mathbb{N}$*, any* $(pk, sk) \leftarrow \textsc{Gen}(1^\lambda)$*, any* $m \in \mathcal{M}$*, and any* $c \leftarrow \textsc{Enc}(pk, m)$*,* $\textsc{Dec}(sk, c) = m$*. Furthermore, the evaluation algorithm* Eval *is perfectly correct in the sense that for any* $\lambda \in \mathbb{N}$*, any* $(pk, sk) \leftarrow \textsc{Gen}(1^\lambda)$*, any* $m_1, \ldots, m_{a(\lambda)} \in \mathcal{M}$*, any* $c_i \leftarrow \textsc{Enc}(pk, m_i)$*, any* $C \in \mathcal{C}_\lambda$*, and any* $c \leftarrow \textsc{Eval}(pk, C, c_1, \ldots, c_{a(\lambda)})$*,* $\textsc{Dec}(sk, c) = C(m_1, \ldots, m_{a(\lambda)})$*.*

Compactness. *The size of the output of* Eval *is polynomial in* λ *and independent of the size of the circuit* C*.*

Security. *For any legitimate PPT adversary* $\mathcal{A}$*, the advantage*

$$Adv^{ind\text{-}cpa}_{\text{Hpke},\mathcal{A}}(\lambda) := Exp^{ind\text{-}cpa}_{\text{Hpke},\mathcal{A}}(\lambda) - \frac{1}{2}$$

is negligible in λ*, where* $Exp^{ind\text{-}cpa}_{\text{Hpke},\mathcal{A}}$ *is defined as in Fig. 4. An adversary* $\mathcal{A}$ *is* legitimate *if it outputs two messages* m_0*,* m_1 *of identical length.*

Without loss of generality, we assume that the secret key is the randomness that was used during the key generation. This enables to test whether key pairs are valid.

3 Construction

3.1 Group Scheme

A group scheme is an abstraction from the properties of groups formalized via a tuple of PPT algorithms. For our purposes, we further abstract this notion to suit groups where group elements do not necessarily have unique encodings. We adapt the notion described in [4] which in turn generalizes the notion introduced in [11]. As demonstrated in [4], such group schemes benefit from the fact that group elements can be represented with many different encodings. This allows

to add auxiliary information inside encodings of group elements in order to add more structure to the group. In our case, however, we exploit that group schemes with non-unique encodings can be used to conceal the structure of the group.

Definition 9 (Group scheme with non-unique encodings). *A* group scheme with non-unique encodings Γ *is a tuple of PPT algorithms* $\Gamma = (\textsc{Setup}, \textsc{Val}, \textsc{Sam}, \textsc{Add}, \textsc{Equal})$.

$\textsc{Setup}(1^\lambda) \to pp$ *On input the unary encoded security parameter* 1^λ, Setup *outputs public parameters* pp. *In particular,* pp *contains the group order* q. *We assume that* pp *is given implicitly to the following algorithms.*

We assume that any encoding is represented as a bit string. In order to decide, whether a given bit string is a valid encoding of a group element, Γ *provides a validation algorithm* Val. *We refer to bit strings causing* Val *to output* 1 *as (valid) encodings of group elements.*

$\textsc{Val}(h) \to \{0,1\}$ *On input a bit string* $h \in \{0,1\}^*$, Val *outputs* 1 *if* h *is a valid encoding with respect to* pp, *otherwise* Val *outputs* 0.

In general, it is not sufficient to compare encodings as bit strings in order to decide whether they represent the same group element. Hence, a group scheme needs to define an algorithm that provides this functionality. This algorithm is called Equal. *We require* Equal *to realize an equivalence relation on the set of valid encodings. For any valid encoding* $h \in \{0,1\}^*$, *let* $\mathcal{G}(h)$ *denote the equivalence class of this encoding. In other words,* $\mathcal{G}(h)$ *contains all encodings that correspond to the same group element as the encoding* h. *For any valid encoding* h, *we require that* $|\{a \in \{0,1\}^* \mid \textsc{Val}(a) = 1\}/\mathcal{G}(h)| = q$ *is the order of the group. We refer to the equivalence classes in* $\{a \in \{0,1\}^* \mid \textsc{Val}(a) = 1\}/\mathcal{G}(h)$ *as group elements.*

$\textsc{Equal}(a,b) \to \{0,1,\bot\}$ *On input two valid encodings* a *and* b, Equal *outputs* 1 *if* a *and* b *represent the same group element, otherwise* Equal *outputs* 0. *If either* a *or* b *is invalid,* Equal *outputs* $\bot$.

In order to perform the group operation on two given encodings, we define an addition algorithm Add.

$\textsc{Add}(a,b)$ *On input two valid encodings* a *and* b, Add *outputs an encoding corresponding to the group element that results from the addition of the group elements represented by* a *and* b. *If either* a *or* b *is invalid,* Add *outputs* $\bot$.

The sampling algorithm Sam *enables to produce an encoding of a group element and only uses information that is part of the public parameters* pp. *Let* h *be a bit string produced via* $\textsc{Sam}(1)$.

For any $z \in \mathbb{N}$, *let* $[z]$ *denote the group element corresponding to the equivalence class* $\mathcal{G}(h^z)$, *where the group operation is performed using* Add. *We require the distribution of* $\textsc{Sam}(z)$ *to be computationally indistinguishable from uniform distribution over* $[z]$.

$\textsc{Sam}(z) \to a$ *On input an exponent* $z \in \mathbb{N}$, Sam *outputs an encoding* a *from the equivalence class* $\mathcal{G}(h^z)$.

Given the order q of the group, it is sufficient to provide an addition algorithm to enable inversion of group elements. To invert a given group element, we use the square-and-multiply approach to add the given encoding $q-1$ times to itself. Further, it suffices to define an algorithm Zero that tests whether a given encoding corresponds to the identity element of the group instead of an algorithm Equal as above. To implement the algorithm Equal on input two encodings a and b, we invert b, add the result to a and test whether the result corresponds to the identity element using Zero.

According to [4], a group scheme with non-unique encodings, in addition to the algorithms defined above, provides an extraction algorithm. The extraction algorithm, given a valid encoding, produces a bit string such that all encodings that represent the same group element lead to the same bit string. However, we omit this algorithm, as our construction does not provide one. It remains an open problem to extend our construction with an extraction algorithm such that the validity of the (m,n)-Interactive Uber assumption (see Definition 10) can still be proven.

3.2 Interactive Uber Assumption

The Uber assumption is a very strong cryptographic assumption in bilinear groups first proposed in [10] and refined in [12]. It provides a natural framework that enables to assess the plausibility of cryptographic assumptions in bilinear groups.

In contrast to the original definition, we consider adaptive attacks (in which an adversary may ask adaptively for more information about the game secrets and choose his challenge).

Definition 10 ((m,n)-Interactive Uber assumption for group schemes). *Let* $m = m(\lambda)$ *and* $n = n(\lambda)$ *such that* $d := \binom{n+m}{m}$ *is a polynomial* [5] *in* λ, *and let* Γ *be a group scheme. The* (m,n)-Interactive Uber assumption *holds for* Γ *if for any legitimate PPT adversary* $\mathcal{A}$, *the advantage* $Adv^{uber}_{\Gamma,\mathcal{A}}(\lambda)$ *is negligible in* λ, *where*

$$Adv^{uber}_{\Gamma,\mathcal{A}}(\lambda) := \Pr\left[Exp^{uber}_{\Gamma,\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2}.$$

The game $Exp^{uber}_{\Gamma,\mathcal{A}}(\lambda)$ *is described in Fig. 5. An adversary* $\mathcal{A}$ *is* legitimate, *if and only if it always guarantees* $P^*(\boldsymbol{X}) \notin \langle 1, P_1(\boldsymbol{X}), \ldots, P_l(\boldsymbol{X})\rangle$ *and for any* $P(\boldsymbol{X}) \in \{P^*(\boldsymbol{X}), P_1(\boldsymbol{X}), \ldots, P_l(\boldsymbol{X})\}$, $\deg(P(\boldsymbol{X})) \leq n$ *in* $Exp^{uber}_{\Gamma,\mathcal{A}}(\lambda)$, *where* $\{P_1(\boldsymbol{X}), \ldots, P_l(\boldsymbol{X})\}$ *are the polynomials that* $\mathcal{A}$ *requests from its oracle* $\mathcal{O}$.

For technical reasons, we need the maximum total degree n of the polynomials appearing in $Exp^{\text{uber}}_{\Gamma,\mathcal{A}}(\lambda)$ and the number of unknowns m to be bounded a priori.

[5] If the parameters m and n both grow at most logarithmically in λ or one of them grows polynomially in λ while the other one is a constant, the binomial coefficient $d = \binom{n+m}{m}$ grows polynomially in λ.

$\underline{\text{Experiment } Exp^{\text{uber}}_{\Gamma,\mathcal{A}}(\lambda)}$

$pp \leftarrow \text{Setup}(1^\lambda),\ \boldsymbol{s} \leftarrow (\mathbb{Z}_q)^m$

$(P^*(\boldsymbol{X}), st) \leftarrow \mathcal{A}^{\mathcal{O}(\cdot)}(1^\lambda, pp, \mathsf{find})$

$b \leftarrow \{0,1\},\ r \leftarrow \mathbb{Z}_q$

$z_0 \leftarrow \text{Sam}(P^*(\boldsymbol{s})),\ z_1 \leftarrow \text{Sam}(r)$

$b' \leftarrow \mathcal{A}^{\mathcal{O}(\cdot)}(1^\lambda, z_b, st, \mathsf{attack})$

if $b = b'$ **then return** 1

return 0

$\underline{\text{Oracle } \mathcal{O}(P(\boldsymbol{X}))}$

return $\text{Sam}(P(\boldsymbol{s}))$

Fig. 5. The description of the (m, n)-Interactive Uber game $Exp^{\text{uber}}_{\Gamma,\mathcal{A}}(\lambda)$. The oracle $\mathcal{O}$ on input a polynomial $P(\boldsymbol{X})$, returns an encoding of the group element $[P(\boldsymbol{s})]$. We refer to $P^*(\boldsymbol{X})$ as "challenge polynomial" and to z_b as "challenge encoding". Further, we call the polynomials that $\mathcal{A}$ requests from the oracle $\mathcal{O}$ "query polynomials".

3.3 Our Construction

Inspired by the construction in [4], an encoding of a group element includes two ciphertexts each encrypting a vector determining an m-variate polynomial over $\mathbb{Z}_q$ of maximum total degree n with respect to some randomly sampled basis $\{a_1, \ldots, a_d\}$. That basis is hidden inside the public parameters of the group scheme via a perfectly binding commitment. An encoding corresponds to the group element whose discrete logarithm equals the evaluation of the thus determined polynomial at a random point $\boldsymbol{\omega} \in \mathbb{Z}_q^m$. That random point $\boldsymbol{\omega}$ is fixed in the public parameters via a point obfuscation po.

For our construction we employ the following building blocks: (i) a dual mode NIWI proof system Π, (ii) a homomorphic encryption scheme Hpke with message space $\mathcal{M} = \mathbb{Z}_q^d$ for a family of circuits of arity $a(\lambda) = 2$ adding two tuples in $\mathbb{Z}_q^d$ component-by-component modulo q, (iii) a point obfuscation PObf for message space $\mathcal{M}_k = \mathbb{Z}_q$, (iv) a family $\mathcal{TD} = (\mathcal{TD}_\lambda)_{\lambda\in\mathbb{N}}$ of families $\mathcal{TD}_\lambda$ of languages TD in a universe $\mathcal{X} = \mathcal{X}_\lambda$ with unique witnesses for $y \in \text{TD}$ such that the subset membership problem $\text{TD} \subseteq \mathcal{X}$ is hard, (v) a perfectly binding non-interactive commitment scheme Com for message space $\mathbb{Z}_q^{d\times d}$, and (vi) a general purpose X-Ind pIO $pi\mathcal{O}$ (i.e. a pIO that is secure with respect to $\mathcal{S}^{X\text{-ind}}$ for a circuit family that only contains circuits with input length at most l, where l is the security parameter used for $pi\mathcal{O}$). Let $n = n(\lambda)$ and let $m = m(\lambda)$ such that $\binom{n+m}{m}$ is a polynomial in λ. The group scheme we construct depends on n and m. We emphasize this fact by calling it $\Gamma_{m,n} := (\text{Setup}, \text{Val}, \text{Sam}, \text{Add}, \text{Equal})$. As mentioned above, we provide an algorithm that tests if a given encoding is an encoding of the identity group element, instead of implementing Equal.

In Fig. 6 we describe the algorithm Setup of our construction. The number q is a prime number that is greater than $2^{p(\lambda)}$ and will serve as the order of our group scheme. We require p to be a polynomial such that $p(\lambda) \geq poly(\lambda)$, where $poly$ is used to scale the security parameter of $pi\mathcal{O}$. We emphasize that our construction allows to arbitrarily choose the group order q as long as q is greater than $2^{p(\lambda)}$ and prime. Therefore, q can be understood as an input of

ALGORITHM SETUP(1^λ)

$(gpk, gsk) \leftarrow \mathsf{Setup}_{\Pi}(1^\lambda)$
$(pk, sk) \leftarrow \text{GEN}(1^\lambda)$, $(pk', sk') \leftarrow \text{GEN}(1^\lambda)$
$\boldsymbol{\omega} \leftarrow (\mathbb{Z}_q)^m$, $\mathsf{po}_i \leftarrow \text{POBF}(1^\lambda, \omega_i)$ for $1 \le i \le m$, $\mathsf{po} := (\mathsf{po}_1, \ldots, \mathsf{po}_m)$
$\text{TD} \leftarrow \mathcal{TD}_\lambda$, $y \leftarrow \mathcal{X} \setminus \text{TD}$
$A \leftarrow \{B \in \text{GL}_d(\mathbb{Z}_q) \mid B \cdot e_1 = e_1\}$
$ck \leftarrow \text{COMSETUP}(1^\lambda)$, $(com, op) \leftarrow \text{COMMIT}_{ck}(A)$
$(crs, td_{\text{ext}}) \leftarrow \mathsf{K}(gpk, gsk)$
$\Lambda_{\text{add}} \leftarrow pi\mathcal{O}(1^{poly(\lambda)}, C_{\mathsf{Add}})$, $\Lambda_{\text{zero}} \leftarrow pi\mathcal{O}(1^{poly(\lambda)}, C^{(0)}_{\mathsf{Zero}})$
return $pp := (q, gpk, crs, y, \text{TD}, pk, pk', \Lambda_{\text{add}}, \Lambda_{\text{zero}}, \mathsf{po}, ck, com)$

Fig. 6. The implementation of the SETUP algorithm producing public parameters pp.

the algorithm SETUP. For the sake of simplicity, we do not write q as input and assume that SETUP generates a suitable group order.

We remark that the circuits C_{Add} and $C^{(0)}_{\mathsf{Zero}}$ that appear in the algorithm SETUP implement the addition of two group elements and a test for the identity element respectively. For a description of these circuits we refer the reader to Fig. 7. The polynomial $poly(\lambda) \ge \lambda$ that is used to scale the security parameter for the obfuscator $pi\mathcal{O}$ upper bounds the input length of these circuits C_{Add} and $C^{(0)}_{\mathsf{Zero}}$. All versions of addition circuits and all versions zero testing circuits that appear during the proofs are padded to the same length respectively. We emphasize that it is necessary to scale the used security parameter as the pIO $pi\mathcal{O}$ we rely on is secure with respect to $\mathcal{S}^{X\text{-ind}}$ for a circuit family that only contains circuits with input length at most λ', where λ' denotes the security parameter that is used to invoke $pi\mathcal{O}$.

Encodings of Group Elements. Encodings of group elements are of the form $h = (C, C', \pi)$. The first two entries C and C' are ciphertexts encrypting vectors $\vec{f} \in \mathbb{Z}_q^d$ and $\vec{f'} \in \mathbb{Z}_q^d$ respectively under the public keys pk and pk' respectively, where d is the dimension of the $\mathbb{Z}_q$ vector space of m-variate polynomials over $\mathbb{Z}_q$ with total degree at most n, i.e. $d = \binom{n+m}{m}$. We require the dimension d of the vector space to grow at most polynomially in λ. The last entry π is the so-called consistency proof. We refer to the vectors $\vec{f}$ and $\vec{f'}$ as *representation vectors* of the group element and to the tuple $(\vec{f}, \vec{f'})$ as *representation* of the group element. Let $\alpha = (\alpha_1, \ldots, \alpha_m) \in \mathbb{N}^m$ denote tuples with $\sum_{i=1}^m \alpha_i \le n$ and let

$$\varphi_{\text{pol}} \colon \mathbb{Z}_q^d \to \mathbb{Z}_q[\boldsymbol{X}], (\ldots, v_\alpha, \ldots)^T \mapsto \sum_\alpha v_\alpha \cdot X_1^{\alpha_1} \cdots X_m^{\alpha_m}$$

be the vector space homomorphism mapping the standard basis of $\mathbb{Z}_q^d$ to a natural basis of the vector space of m-variate polynomials of degree at most n. For well-definedness we use the lexicographical order on the tuples $(\alpha_1, \ldots, \alpha_m) \in \mathbb{N}^m$,

particularly, the first vector of the standard basis of $\mathbb{Z}_q^d$ is mapped to the constant polynomial 1. The image of φ_{pol} is $\mathrm{Im}(\varphi_{\mathrm{pol}}) = \{p \in \mathbb{Z}_q[\boldsymbol{X}] \,|\, \deg(p) \leq n\}$ and the kernel is $\ker(\varphi_{\mathrm{pol}}) = \{0\}$. Hence, φ_{pol} is an isomorphism between the vector spaces $\mathbb{Z}_q^d$ and $\mathrm{Im}(\varphi_{\mathrm{pol}})$.

We recall that $\textsc{Setup}(1^\lambda)$ samples the matrix A uniformly at random from $\mathrm{GL}_d(\mathbb{Z}_q)$ such that the first column equals e_1. Hence, the matrix A^{-1} exists and has the form $A^{-1} = (a_1 \mid a_2 \mid \ldots \mid a_d)$ such that $a_1 = e_1$. The columns $a_1, \ldots, a_d \in \mathbb{Z}_q^d$ form a basis of the vector space $\mathbb{Z}_q^d$.

The coefficients of the representation vectors $\vec{f} = (f_1, \ldots, f_d)^T$ and $\vec{f'} = (f'_1, \ldots, f'_d)^T$ of a group element define the polynomials $f(\boldsymbol{X}), f'(\boldsymbol{X}) \in \mathrm{Im}(\varphi_{\mathrm{pol}})$ via

$$\begin{aligned} f(\boldsymbol{X}) &:= \sum_{i=1}^{d} f_i \cdot \varphi_{\mathrm{pol}}(a_i) & f'(\boldsymbol{X}) &:= \sum_{i=1}^{d} f'_i \cdot \varphi_{\mathrm{pol}}(a_i) \\ &= \varphi_{\mathrm{pol}}\left(A^{-1} \cdot \vec{f}\right) & &= \varphi_{\mathrm{pol}}\left(A^{-1} \cdot \vec{f'}\right) \end{aligned}$$

In other words, the representation vectors $\vec{f}$ and $\vec{f'}$ are the representations of the abstract polynomials $f(\boldsymbol{X})$ and $f'(\boldsymbol{X})$ respective to the basis $\{\varphi_{\mathrm{pol}}(a_1) = \varphi_{\mathrm{pol}}(e_1), \varphi_{\mathrm{pol}}(a_2), \ldots, \varphi_{\mathrm{pol}}(a_d)\}$. Intuitively, a valid encoding that contains the representation vector $\vec{f} \in \mathbb{Z}_q^d$ corresponds to the group element $[f(\boldsymbol{\omega})]$, where $\boldsymbol{\omega}$ is the value that is fixed in the public parameters of the group scheme via po. The same holds for the representation vector $\vec{f'}$ resulting in a redundant encoding. This approach is similar to the Naor-Yung paradigm [35].

We call the representation $(\vec{f}, \vec{f'})$ *consistent* if both representation vectors correspond to the same group element, i.e. the evaluation of the corresponding polynomials $f(\boldsymbol{X})$ and $f'(\boldsymbol{X})$ at $\boldsymbol{\omega}$ are equal. Otherwise, we call such a representation *inconsistent*. If the representation $(\vec{f}, \vec{f'})$ is consistent, we call this representation *constant* if the corresponding polynomials $f(\boldsymbol{X})$ and $f'(\boldsymbol{X})$ are constant (i.e. are of total degree at most 0). If a consistent representation is not constant we call this representation *non-constant*. The purpose of the so-called consistency proof is to ensure consistency of encodings, i.e. to ensure that the corresponding representation is consistent. Further, we use the terms constant, non-constant, consistent, and inconsistent to characterize encodings if the associated representation has the respective properties.

Consistency Proof and Validation Algorithm. The above mentioned *consistency proof* ensures that the representations, that are encrypted inside of encodings, are consistent. In other words, the consistency proof ensures that both representation vectors $\vec{f}$ and $\vec{f'}$ used for an encoding lead to the same group element. We realize this by using the dual mode NIWI proof system Π to produce the consistency proof π for a relation $\mathcal{R}$. The relation $\mathcal{R}$ is a disjunction of three main statements $\mathcal{R} = \mathcal{R}_1 \vee \mathcal{R}_2 \vee \mathcal{R}_3$:

The relation $\mathcal{R}_1$ is satisfied for representations that are constant and consistent. We formalize this via relation $\mathcal{R}_{1.a}$:

$$\mathcal{R}_{1.a} := \left[\vec{f} = \vec{f'} \wedge \deg\left(\varphi_{\text{pol}}(\vec{f})\right) \leq 0\right]$$

We recall the convention that the degree of the zero polynomial is defined to be $-\infty$. For technical reasons, we need to make sure that the knowledge of the secret decryption keys (sk, sk') and the knowledge of the used encryption randomness are both sufficient as witnesses. Thus, additionally to $\mathcal{R}_{1.a}$ we define the two relations $\mathcal{R}_b$ and $\mathcal{R}_c$. The relations $\mathcal{R}_b$ and $\mathcal{R}_c$ connect the ciphertexts C, C' of the encoding with the corresponding representation vectors $\vec{f}$, $\vec{f'}$ appearing in relation $\mathcal{R}_{1.a}$.

$$\mathcal{R}_b := \left[\quad C = \textsc{Enc}(pk, \vec{f}; R) \wedge C' = \textsc{Enc}(pk', \vec{f'}; R') \quad\right]$$
$$\mathcal{R}_c := \left[\begin{array}{c} (pk, sk) = \textsc{Gen}(sk) \ \wedge\ \vec{f} = \textsc{Dec}(sk, C) \quad \wedge \\ (pk', sk') = \textsc{Gen}(sk') \wedge \vec{f'} = \textsc{Dec}(sk', C') \end{array}\right]$$

At this point we make use of the assumption that a secret decryption key equals the randomness that was used to produce the corresponding public encryption key. The relation $\mathcal{R}_1$ is defined as follows:

$$\mathcal{R}_1 := \mathcal{R}_{1.a} \wedge (\mathcal{R}_b \vee \mathcal{R}_c)\,. \tag{7}$$

Given a consistent and constant representation $(\vec{f}, \vec{f'})$ and resulting ciphertexts C and C', there are two possible witnesses to produce the consistency proof for the relation $\mathcal{R}_1$: using the secret decryption keys (sk, sk', $\vec{f}$, $\vec{f'}$) and using the encryption randomness $((\vec{f}, R), (\vec{f'}, R'))$.

The relation $\mathcal{R}_2$ is satisfied for representations that are consistent. Again, we formalize this via a relation $\mathcal{R}_{2.a}$:

$$\mathcal{R}_{2.a} := \left[\begin{array}{lll} \varphi_{\text{pol}}\left(A^{-1} \cdot \vec{f}\right)(\boldsymbol{\omega}) & = \varphi_{\text{pol}}\left(A^{-1} \cdot \vec{f'}\right)(\boldsymbol{\omega}) & \wedge \\ \forall i \in \{1, \ldots, m\} \colon \mathsf{po}_i(\omega_i) & = 1 & \wedge \\ \textsc{Open}_{ck}(com, op) & = A \ \wedge\ A \neq \perp & \end{array}\right]$$

The relation $\mathcal{R}_2$ is defined as follows:

$$\mathcal{R}_2 := \mathcal{R}_{2.a} \wedge (\mathcal{R}_b \vee \mathcal{R}_c)\,. \tag{8}$$

Given a consistent representation $(\vec{f}, \vec{f'})$ and resulting ciphertexts C and C', there are two possible witnesses to produce the consistency proof for the relation $\mathcal{R}_2$: using the secret decryption keys (sk, sk', $\vec{f}$, $\vec{f'}$, $\boldsymbol{\omega}$, op) and using the encryption randomness $((\vec{f}, R), (\vec{f'}, R'), \boldsymbol{\omega}, op)$. To be precise, the matrix A

also is part of these witnesses. However, as we can assume that A is a part of op, we omit this fact in our notation.

The relation $\mathcal{R}_3$ introduces a trapdoor enabling production of consistency proofs for inconsistent encodings.

$$\mathcal{R}_3 := \big[y \in \mathrm{TD}\big]. \tag{9}$$

This relation only depends on the instance (TD, y) of the subset membership problem $\mathrm{TD} \subseteq \mathcal{X}$ defined in the public parameters. We recall that if $y \in \mathrm{TD}$, there exists a unique witness w_y satisfying the witness relation for the SMP. Hence, the witness for the relation $\mathcal{R}_3$ is (w_y). Given public parameters pp that are generated via $\textsc{Setup}(1^\lambda)$, y is not in TD. Therefore, there exists no trapdoor if pp is generated honestly.

Let rp denote the parts of the public parameters that are necessary to produce consistency proofs, i.e. $rp := (q,\ pk,\ pk',\ \mathsf{po},\ ck,\ com,\ \mathrm{TD},\ y)$. To be precise, the corresponding language L has the following form:

$$\begin{aligned} L :=& \{x = (\underbrace{q, pk, pk', \mathsf{po}, ck, com, \mathrm{TD}, y}_{=rp}, C, C') \,|\, \exists w : (x, w) \in \mathcal{R}\} \\ =& \ L_1 \cup L_2 \cup L_3, \end{aligned}$$

where $L_i := \{x = (rp, C, C') \,|\, \exists w\colon (x, w) \in \mathcal{R}_i\}$. For the sake of clarity, we henceforth omit the parameters rp and treat the tuple $(C,\ C')$ as the statement.

The validation algorithm $\textsc{Val}$, on input a bit string $h \in \{0,1\}^*$, parses h into $(C,\ C',\ \pi)$ and executes $\mathsf{Verify}(gpk,\ crs,\ x,\ \pi)$ of the underlying NIWI proof system Π for the relation $\mathcal{R}$.

Addition and Zero Algorithm. The implementations of the algorithms $\textsc{Add}$ and $\textsc{Zero}$ need to know secret information that is associated with the public parameters, for instance the secret decryption keys. Therefore, we implement these algorithms as probabilistic circuits and "hard-code" the necessary secret parameters inside. The security requirement of the employed obfuscator $pi\mathcal{O}$ enables to conceal the implementation of these circuits and, hence, conceals the secret parameters that are hard-coded. The PPT algorithms $\textsc{Add}$ and $\textsc{Zero}$ simply execute the respective obfuscated circuit Λ_{add} and Λ_{zero}.

In Fig. 7 we present the implementation of the circuit C_{Add} and the implementation of the circuit $C^{(0)}_{\mathsf{Zero}}$. We remark that C_{Zero} only uses the representation vector $\vec{f}$ and ignores the representation vector $\vec{f'}$. This enables to exploit the Naor-Yung like double encryption.

The addition circuit C_{Add} is similar to the one constructed in [4]. The difference is limited to the fact that in our case C_{Add} differentiates between three instead of two different possibilities to produce the new consistency proof. The encodings of group elements in the construction of [4] are of the form $(h,\ C,\ C',\ \pi)$, where C and C' are some ciphertexts and π is a corresponding consistency proof. The value h is the group element in an underlying group that

CIRCUIT $C_{\mathsf{Add}}[gpk, rp, sk, sk', \boldsymbol{\omega}, op, td_{\text{ext}}](a, b)$

if $\neg\text{VAL}(a) \vee \neg\text{VAL}(b)$ **then return** $\perp$
parse $a =: (C^{(a)}, C'^{(a)}, \pi^{(a)})$
parse $b =: (C^{(b)}, C'^{(b)}, \pi^{(b)})$
$C^{(c)} := \text{EVAL}(pk, \oplus, C^{(a)}, C^{(b)})$
$C'^{(c)} := \text{EVAL}(pk', \oplus, C'^{(a)}, C'^{(b)})$
$\vec{f}^{(a)} := \text{DEC}(sk, C^{(a)}), \vec{f}'^{(a)} := \text{DEC}(sk', C'^{(a)})$
$\vec{f}^{(b)} := \text{DEC}(sk, C^{(b)}), \vec{f}'^{(b)} := \text{DEC}(sk', C'^{(b)})$
$\vec{f}^{(c)} := \oplus(\vec{f}^{(a)}, \vec{f}^{(b)}),\ \vec{f}'^{(c)} := \oplus(\vec{f}'^{(a)}, \vec{f}'^{(b)})$
if $(C^{(a)}, C'^{(a)}), (C^{(b)}, C'^{(b)}) \in L_1$ **then**
  $\pi^{(c)} \leftarrow \mathsf{Prove}(gpk, crs, (C^{(c)}, C'^{(c)}), (sk, sk', \vec{f}^{(c)}, \vec{f}'^{(c)}))$
elseif $(C^{(a)}, C'^{(a)}), (C^{(b)}, C'^{(b)}) \in L_2$ **then**
  $\pi^{(c)} \leftarrow \mathsf{Prove}(gpk, crs, (C^{(c)}, C'^{(c)}), (sk, sk', \vec{f}^{(c)}, \vec{f}'^{(c)}, \boldsymbol{\omega}, op))$
else
  let $\alpha \in \{a, b\} :\ (C^{(\alpha)}, C'^{(\alpha)}) \notin L_1 \cup L_2$
  $w_y \leftarrow \mathsf{Extract}(td_{\text{ext}}, (C^{(\alpha)}, C'^{(\alpha)}), \pi^{(\alpha)})$
  $\pi^{(c)} \leftarrow \mathsf{Prove}(gpk, crs, (C^{(c)}, C'^{(c)}), (w_y))$
return $c := (C^{(c)}, C'^{(c)}, \pi^{(c)})$

CIRCUIT $C^{(0)}_{\mathsf{Zero}}[q, sk, \boldsymbol{\omega}, A](a)$

if $\neg\text{VAL}(a)$ **then**
  return $\perp$
parse $a =: (C, C', \pi)$
$\vec{f} \leftarrow \text{DEC}(sk, C)$
$f(\boldsymbol{X}) := \varphi_{\text{pol}}(A^{-1} \cdot \vec{f})$
if $f(\boldsymbol{\omega}) = 0$ **then**
  return 1
return 0

Fig. 7. Circuit C_{Add} (left) for addition of two group elements, and circuit $C^{(0)}_{\mathsf{Zero}}$ (right) for testing whether a given encoding is an encoding of the identity element. Additionally to the publicly available parameters gpk and rp, C_{Add} has the secret decryption keys sk, sk', the values $\boldsymbol{\omega}$, the opening op, and the extraction trapdoor td_{ext} hard-coded. The circuit $C^{(0)}_{\mathsf{Zero}}$ knows the publicly available parameter q and additionally has the secret parameters sk, $\boldsymbol{\omega}$, and A hard-coded. The circuit $\oplus$ realizes addition in $\mathbb{Z}_q^d$.

is represented by the encoding. As h uniquely identifies the represented group element, the equality test simply compares these values of the given encodings. In our case, however, the encodings do not contain a similar entry. Therefore, the implementation of the equality test, or rather the zero test, needs to decrypt the ciphertext C in order to be able to make a statement about the represented group element.

Sampling Algorithm. The sampling algorithm SAM, on input an exponent $z \in \mathbb{N}$, uses the representation $(\vec{f},\ \vec{f'}) := ((z,\ 0,\ \ldots,\ 0)^T,\ (z,\ 0,\ \ldots,\ 0)^T)$ to produce an encoding of the requested group element. The consistency proof is produced for relation $\mathcal{R}_1$ using the witness $((\vec{f},\ R),\ (\vec{f'},\ R'))$, where R and R' are the randomnesses that are used to encrypt $\vec{f}$ and $\vec{f'}$ respectively. If the sampling algorithm does not receive any input, it samples the exponent z from $\{0,\ \ldots,\ q-1\}$ uniformly at random and proceeds as above. Due to the IND-CPA security of HPKE, the distribution of the output of $\text{SAM}(z)$ is

computationally indistinguishable from uniform distribution over the equivalence class $\mathcal{G}(\text{SAM}(z))$.

We remark that our group scheme allows for re-randomization of encodings. To re-randomize a given encoding, we sample an encoding of the identity element and use the addition algorithm to add it to the encoding to be randomized. We require the employed homomorphic encryption scheme to satisfy an additional natural property. Namely, we require that ciphertexts can be re-randomized by homomorphically adding a fresh ciphertext of 0. This property is also known as circuit privacy.

3.4 Main Theorem

Theorem 1. *Let $\Gamma_{m,n}$ be the group scheme constructed in Sect. 3.3. Further, let piO be a probabilistic indistinguishability obfuscator with respect to $\mathcal{S}^{X\text{-}ind}$ for a circuit family containing circuits with input length at most poly(λ), let $\mathcal{TD} = (\mathcal{TD}_\lambda)_{\lambda\in\mathbb{N}}$ be a family of families $\mathcal{TD}_\lambda = \{TD\}$ of languages $TD \subseteq \mathcal{X}_\lambda$ such that the subset membership problem is hard, let Π be a dual mode NIWI proof system, let* HPKE *be an IND-CPA secure HPKE scheme, let* COM *be a perfectly binding non-interactive commitment scheme, and let* POBF *be a point obfuscation. Then, the (m, n)-Interactive Uber assumption (cf. Definition 10) holds for $\Gamma_{m,n}$.*

In Table 1 we give an overview on the proof of Theorem 1. Informally, the "Switching lemma" states that encodings containing different representations of the same group element are hard to distinguish. The distribution $\widetilde{pp}$ denotes the distribution of public parameters that are sampled according to SETUP with the difference that y is sampled from within the trapdoor language TD. The distribution $\widehat{pp}$ denotes the same distribution as $\widetilde{pp}$ with the difference that the CRS is sampled in hiding mode and Λ_{add} is computed for an addition circuit that simulates consistency proofs and, hence, does not need to know the matrix A or the value $\boldsymbol{\omega}$. On a high level, the "Swap lemma" states that these two distributions of public parameters are computationally indistinguishable.

The distribution $\overline{pp}^{(i)}$ (for $i \in \{0, \ldots, m\}$) denotes the same distribution as $\widehat{pp}$ with the difference that Λ_{zero} is an obfuscation of a zero testing circuit that tests whether the polynomial $f(X_1, \ldots, X_i, \omega_{i+1}, \ldots, \omega_m)$ equals the zero polynomial. Furthermore, the point obfuscations in $\widehat{pp}$ obfuscate $\perp$ whereas the point obfuscations in $\overline{pp}^{(i)}$ obfuscate the values $\omega_{i+1}, \ldots, \omega_m$. The distribution $\underline{pp}$ is the same as $\overline{pp}^{(m)}$ with the difference that Λ_{zero} is produced for a zero testing circuit that simply tests whether the representation vector $\vec{f}$ equals zero in $\mathbb{Z}_q^d$ and, hence, does not need to know the matrix A and $\boldsymbol{\omega}$ anymore.

The "Randomization lemma" basically states that the images of a certain subspace under a randomly sampled vector space isomorphism do not leak any information on the behavior of that isomorphism on pre-images that do not lie in that span.

For the formal definitions and the full proofs we refer the reader to the full version [3].

Table 1. An overview on the steps of the proof of 1. The boxes emphasize changes compared to the previous game. Let W_i denote the witness that is used to prove relation $\mathcal{R}_i$ for $i \in \{1, 2, 3\}$. The witnesses W_1 and W_2 contain the used encryption randomness. Further, for a polynomial $P(\boldsymbol{X})$, let $R_P := A \cdot \varphi_{\text{pol}}^{-1}(P(\boldsymbol{c} \circ \boldsymbol{X}))$, and for a vector $v^* \in \mathbb{Z}_q^d$, let $\overline{R}_{v^*} := \varphi_{\text{pol}}\left(A^{-1} \cdot v^*\right)(\boldsymbol{\omega}) \cdot e_1$.

	Publ. param.	Secret s	Representations for queries P / challenge P^*		Witness for π	Remark
Game_0	pp	$\boldsymbol{s} \leftarrow \mathbb{Z}_q^m$	$P(\boldsymbol{s}) \cdot e_1$	$P^*(\boldsymbol{s}) \cdot e_1$	W_1	the real Uber game
Game_1	pp	$\boxed{\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m}$	$P(\boldsymbol{s}) \cdot e_1$	$P^*(\boldsymbol{s}) \cdot e_1$	W_1	negl. statistical distance
Game_2	pp	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	$\boxed{R_P}$	$\boxed{R_{P^*}}$	W_1, W_2	Switching lemma (see [3])
Game_3	$\boxed{\widetilde{pp}}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	R_{P^*}	W_1, W_2	SMP TD $\subseteq \mathcal{X}$
Game_4	$\boxed{\widehat{pp}}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	R_{P^*}	W_1, W_2	Swap lemma (see [3])
Game_5	$\widehat{pp}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	R_{P^*}	$\boxed{W_3}$	perfect WI of Π
Game_6	$\boxed{\overline{pp}^{(0)}}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	R_{P^*}	W_3	hiding property of Com
Game_7	$\boxed{\overline{pp}^{(m)}}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	R_{P^*}	W_3	see [3]
Game_8	$\boxed{\underline{pp}}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	R_{P^*}	W_3	security of $pi\mathcal{O}$
Game_9	$\underline{pp}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	$\boxed{v^* \leftarrow \mathbb{Z}_q^d}$	W_3	Rand. lemma (see [3])
Game_{10}	$\boxed{\overline{pp}^{(m)}}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	$v^* \leftarrow \mathbb{Z}_q^d$	W_3	security of $pi\mathcal{O}$
Game_{11}	$\boxed{\overline{pp}^{(0)}}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	$v^* \leftarrow \mathbb{Z}_q^d$	W_3	see [3]
Game_{12}	$\boxed{\widehat{pp}}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	$v^* \leftarrow \mathbb{Z}_q^d$	W_3	hiding property of Com
Game_{13}	$\widehat{pp}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	$v^* \leftarrow \mathbb{Z}_q^d$	$\boxed{W_1, W_2}$	perfect WI of Π
Game_{14}	$\boxed{\widetilde{pp}}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	$v^* \leftarrow \mathbb{Z}_q^d$	W_1, W_2	Swap lemma (see [3])
Game_{15}	$\boxed{pp}$	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	R_P	$v^* \leftarrow \mathbb{Z}_q^d$	W_1, W_2	SMP TD $\subseteq \mathcal{X}$
Game_{16}	pp	$\boldsymbol{s} := \boldsymbol{c} \circ \boldsymbol{\omega},\ \boldsymbol{c} \leftarrow \left(\mathbb{Z}_q^\times\right)^m$	$\boxed{P(\boldsymbol{s}) \cdot e_1}$	$\boxed{\overline{R}_{v^*},\ v^* \leftarrow \mathbb{Z}_q^d}$	W_1	Switching lemma (see [3])
Game_{17}	pp	$\boxed{\boldsymbol{s} \leftarrow \mathbb{Z}_q^m}$	$P(\boldsymbol{s}) \cdot e_1$	$\overline{R}_{v^*},\ v^* \leftarrow \mathbb{Z}_q^d$	W_1	negl. statistical distance
Game_{18}	pp	$\boldsymbol{s} \leftarrow \mathbb{Z}_q^m$	$P(\boldsymbol{s}) \cdot e_1$	$\boxed{r \cdot e_1,\ r \leftarrow \mathbb{Z}_q}$	W_1	identically distributed

Acknowledgements. We would like to thank Antonio Faonio, Pooya Farshim, and Jesper Buus Nielsen for many interesting discussions. We would also like to thank the reviewers for many helpful comments.

References

1. Abe, M., Groth, J., Haralambiev, K., Ohkubo, M.: Optimal structure-preserving signatures in asymmetric bilinear groups. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 649–666. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_37
2. Abe, M., Groth, J., Ohkubo, M., Tibouchi, M.: Unified, minimal and selectively randomizable structure-preserving signatures. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 688–712. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54242-8_29
3. Agrikola, T., Hofheinz, D.: Interactively secure groups from obfuscation. Cryptology ePrint Archive, report 2018/010. https://eprint.iacr.org/2018/010 (2018)
4. Albrecht, M.R., Farshim, P., Hofheinz, D., Larraia, E., Paterson, K.G.: Multilinear maps from obfuscation. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016. LNCS, vol. 9562, pp. 446–473. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49096-9_19
5. Barak, B., Garg, S., Kalai, Y.T., Paneth, O., Sahai, A.: Protecting obfuscation against algebraic attacks. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 221–238. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_13
6. Bellare, M., Palacio, A.: The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 273–289. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28628-8_17
7. Bellare, M., Stepanovs, I.: Point-function obfuscation: a framework and generic constructions. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016. LNCS, vol. 9563, pp. 565–594. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49099-0_21
8. Bitansky, N., Canetti, R., Kalai, Y.T., Paneth, O.: On virtual grey box obfuscation for general circuits. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8617, pp. 108–125. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44381-1_7
9. Boneh, D., Boyen, X.: Efficient selective-id secure identity-based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_14
10. Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical identity based encryption with constant size ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_26
11. Boneh, D., Silverberg, A.: Applications of multilinear forms to cryptography. Contemp. Math. **324**(1), 71–90 (2003)
12. Boyen, X.: The uber-assumption family. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 39–56. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85538-5_3

13. Brakerski, Z., Rothblum, G.N.: Virtual black-box obfuscation for all circuits via generic graded encoding. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 1–25. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54242-8_1
14. Brown, D.R.L.: Generic Groups, Collision Resistance, and ECDSA. Des. Codes Cryptograph. **35**(1), 119–152 (2005)
15. Brown, D.R.L.: Toy factoring by Newton's method. IACR ePrint Archive, report 2008/149 (2008). http://eprint.iacr.org/2008/149
16. Canetti, R.: Towards realizing random oracles: Hash functions that hide all partial information. In: Kaliski, B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 455–469. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0052255
17. Canetti, R., Lin, H., Tessaro, S., Vaikuntanathan, V.: Obfuscation of probabilistic circuits and applications. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9015, pp. 468–497. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_19
18. Cantor, D.G., Zassenhaus, H.: A new algorithm for factoring polynomials over finite fields. Math. Comput. **36**, 587–592 (1981)
19. Chase, M., Maller, M., Meiklejohn, S.: Déjà Q all over again: tighter and broader reductions of q-type assumptions. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10032, pp. 655–681. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53890-6_22
20. Chase, M., Meiklejohn, S.: Déjà Q: using dual systems to revisit q-type assumptions. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 622–639. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_34
21. Cheon, J.H.: Security analysis of the strong Diffie-Hellman problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 1–11. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_1
22. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-46035-7_4
23. Damgård, I.: Towards practical public key systems secure against chosen ciphertext attacks. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 445–456. Springer, Heidelberg (1992). https://doi.org/10.1007/3-540-46766-1_36
24. Danezis, G., Fournet, C., Groth, J., Kohlweiss, M.: Square span programs with applications to succinct NIZK arguments. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8873, pp. 532–550. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45611-8_28
25. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976)
26. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 1–17. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38348-9_1
27. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: Proceedings of FOCS 2013. IEEE Computer Society, pp. 40–49 (2013)
28. Gentry, C.: A fully homomorphic encryption scheme. Ph.D thesis. Stanford University (2009)
29. Goldreich, O.: Foundations of Cryptography: Basic Tools. Cambridge University Press, Cambridge (2001)

30. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_24
31. Hada, S., Tanaka, T.: On the existence of 3-round zero-knowledge protocols. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 408–423. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055744
32. Koblitz, N., Menezes, A.: The brave new world of bodacious assumptions in cryptography. Not. AMS **57**, 357–365 (2010)
33. Lipmaa, H.: On the CCA1-security of Elgamal and Damgård's Elgamal. In: Lai, X., Yung, M., Lin, D. (eds.) Inscrypt 2010. LNCS, vol. 6584, pp. 18–35. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21518-6_2
34. Meiklejohn, S.: An extension of the Groth-Sahai proof system. Ph.D thesis. Brown University (2009)
35. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing, pp. 427–437. ACM (1990)
36. Pass, R., Seth, K., Telang, S.: Indistinguishability obfuscation from semantically-secure multilinear encodings. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 500–517. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_28
37. Wee, H.: On obfuscating point functions. In: Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing, pp. 523–532. ACM (2005)
38. Zimmerman, J.: How to obfuscate programs directly. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 439–467. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_15