J Cryptol (2019) 32:239–264
https://doi.org/10.1007/s00145-018-9292-1

Journal of
CRYPTOLOGY

On the Impossibility of Structure-Preserving Deterministic Primitives

Masayuki Abe
NTT Secure Platform Laboratories, NTT Corporation, Tokyo, Japan
abe.masayuki@lab.ntt.co.jp

Jan Camenisch
IBM Research - Zurich, Rüschlikon, Switzerland
jca@zurich.ibm.com

Rafael Dowsley
Aarhus University, Aarhus C, Denmark
rafael@cs.au.dk

Maria Dubovitskaya
IBM Research - Zurich, Rüschlikon, Switzerland
mdu@zurich.ibm.com

Communicated by Eike Kiltz.

Received 4 December 2015 / Revised 28 December 2017
Online publication 14 May 2018

Abstract. In structure-preserving cryptography over bilinear groups, cryptographic schemes are restricted to exchange group elements only, and their correctness must be verifiable only by evaluating pairing product equations. Several primitives, such as structure-preserving signatures, commitments, and encryption schemes, have been proposed. Although deterministic primitives, such as verifiable pseudorandom functions or verifiable unpredictable functions, play an important role in the construction of cryptographic protocols, no structure-preserving realizations of them are known. This is not coincident: In this paper, we show that it is impossible to construct *algebraic* structure-preserving deterministic primitives that provide provability, uniqueness, and unpredictability. This includes verifiable random functions, unique signatures, and verifiable unpredictable functions as special cases. The restriction of structure-preserving primitives to be algebraic is natural, otherwise it would not be known how to verify correctness only by evaluating pairing product equations. We further extend our negative result to pseudorandom functions and deterministic public key encryption as well as non-strictly structure-preserving primitives, where target group elements are also allowed in their ranges and public keys.

Keywords. Structure-preserving cryptography, Verifiable random functions, Unique signatures, Groth–Sahai proofs.

1. Introduction

Most practical cryptographic protocols are built from cryptographic primitives such as signature, encryption, and commitments schemes, pseudorandom functions, and zero-knowledge (ZK) proofs. Thereby, the ZK proofs often "glue" different building blocks together by proving relations among their inputs and outputs. The literature provides a fair number of different cryptographic primitives (e.g., CL-signatures [20,21], Pedersen commitments [50], ElGamal and Cramer-Shoup encryption [27,30], verifiable encryption of discrete logarithms [23], and verifiable pseudorandom functions [29]) that are based on the discrete logarithm problem and that, together with so-called generalized Schnorr protocols [18,51], provide a whole framework for the construction of practical protocols. Examples of such constructions include anonymous credential systems [6,19], oblivious transfer with access control [15], group signatures [8,44], and e-cash [17]. The non-interactive versions of generalized Schnorr protocols are secure only in the random oracle model, as they are obtained via the Fiat–Shamir heuristic [32], and it is well known that random oracles cannot be instantiated securely [25,36]. Consequently, many protocols constructed from this framework are secure in the random oracle model only.

A seminal step toward a framework allowing for security proofs in the *standard model* was the introduction of the so-called GS proofs by Groth and Sahai [38]. These are efficient non-interactive proofs of knowledge or language membership and are secure in the standard model. They make use of bilinear maps to verify statements and, because of this, are limited to languages of certain types of equations, such as pairing product and multi-exponentiation equations. In particular, GS proofs are proofs of *knowledge* only for witnesses that are group elements but not for exponents. Thus, it is unfortunately not possible to use GS proofs as a replacement for generalized Schnorr proofs in the "discrete logarithm-based framework of cryptographic primitives" described earlier. To alleviate this and to obtain a similar construction framework, the research community has engaged in a quest for alternative cryptographic primitives that are *structure-preserving* [3], i.e., primitives for which the public keys, inputs, and output consist of (source) group elements and the verification predicate is a conjunction of pairing product equations (PPEs), thus making them compatible with proof systems that support only pairing product equations. Such a framework is especially attractive because GS proofs are "on-line" extractable (i.e., the extractor works without having to rewind the prover), a property that is essential for the construction of UC-secure [24] protocols.

Structure-preserving realizations exist for primitives such as signature schemes [2–4, 13,40], commitment schemes [3,5], and encryption schemes [16]. However, no *structure-preserving* constructions are known for important primitives including pseudorandom functions (PRFs) [28,35], verifiable unpredictable functions (VUFs) [47], verifiable random functions (VRFs) [41,47], simulatable VRFs [26], unique signatures (USigs) [37,46,47], and deterministic encryption (DE) [11,12], despite the fact that these primitives are widely used in the literature. Examples include efficient search on encrypted data [12] from DE; micropayments [49] from unique signatures; resettable ZK proofs [48], updatable ZK databases [45], and verifiable transaction escrow schemes [43] from VRFs. PRFs together with a proof of correct evaluation have been used to construct compact e-cash [17], keyword search [33], set intersection protocols [39], and adap-

tive oblivious transfer protocols [14,22,42]. We further refer to Abdalla et al. [1] and Hohenberger and Waters [41] for a good overview of applications of VRFs.

Our Results In this paper, we show that it is no coincidence that no structure-preserving constructions of PRFs, VRFs, VUFs, USigs, and DE are known: It is in fact impossible to construct them with *algebraic* algorithms. To this end, we provide a generic definition of a secure structure-preserving deterministic primitive (SPDP) and show that such a primitive cannot be built using algebraic operations only. The latter is a very reasonable restriction: All known constructions of structure-preserving primitives are algebraic. We then show that PRFs, VRFs, VUFs, and USigs are special cases of an SPDP. We further extend our results to DE and to "non-strictly" structure-preserving primitives, which are allowed to have target group elements in their public keys and ranges.

Let us point out that of course our results do not rule out the possibility of constructing efficient protocols from GS proofs and non-structure-preserving primitives. Indeed, a couple of such protocols are known where, although some of the inputs include exponents (e.g., x), they turned out to be sufficient if only knowledge of a group element (e.g., g^x) is proved. Examples here include the construction of a compact e-cash scheme [7] from the Dodis–Yampolskiy VRF [29] and of the so-called F-unforgeable signature scheme [6] and its use in the construction of anonymous credentials.

Related Work Some impossibility results and lower bounds for structure-preserving primitives are known. Abe et al. [4] show that in the case of asymmetric pairings any signature made with a structure-preserving signature scheme must consist of at least three group elements when the signing algorithm is algebraic. They also give constructions meeting this bound. Lower bounds for structure-preserving commitment schemes are presented by Abe et al. [5]. They show that a commitment cannot be shorter than the message and that verifying the opening of a commitment in a symmetric bilinear group setting requires evaluating at least two independent PPEs. They also provide optimal constructions that match these lower bounds.

Paper Organization In Sect. 2, we specify our notation, define the syntax and security properties of an algebraic SPDP, and show that such primitives are impossible to construct. In Sect. 2.5, we present some generalizations to primitives that are not strictly structure-preserving. Then, in Sect 3, we show how our results can be applied to structure-preserving PRFs, VRFs, VUFs, and USigs. Section 3.4 is devoted to the impossibility results for structure-preserving DE. Finally, we conclude the paper and discuss open problems and possible future research directions in Sect. 4.

2. Definitions and Impossibility Results for Algebraic Structure-Preserving Deterministic Primitives

2.1. *Preliminaries*

Notation We say that a function is *negligible* in the security parameter λ if it is asymptotically smaller than the inverse of any fixed polynomial in λ. The function is said to

be *non-negligible* in λ, otherwise. We say that an event occurs with *overwhelming* probability if it occurs with probability $p(\lambda) \geq 1 - \mathsf{negl}(\lambda)$, where $\mathsf{negl}(\lambda)$ is a negligible function of λ.

We denote as $Y \xleftarrow{\$} \mathsf{F}(X)$ a *probabilistic* algorithm that on input X outputs Y. A similar notation $Y \leftarrow \mathsf{F}(X)$ is used for a *deterministic* algorithm with X and Y.

We use an upper-case, multiplicative notation for group elements and lower-case letters for exponents. Let $\mathcal{G}$ be a bilinear group generator that takes as input a security parameter 1^λ and outputs the description of a bilinear group $\Lambda = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G_1, G_2)$, where $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ are groups of prime order p, e is an efficient, non-degenerated bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, and G_1 and G_2 are generators of groups $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. We denote as $\Lambda^* = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ the description Λ without the group generators. By Λ_{sym}, we denote the symmetric setting where $\mathbb{G}_1 = \mathbb{G}_2$ and $G_1 = G_2$. In the symmetric setting, we simply write $\mathbb{G}$ for both $\mathbb{G}_1$ and $\mathbb{G}_2$, and G for G_1 and G_2.

We also denote the set of all possible vectors of group elements from both $\mathbb{G}_1$ and $\mathbb{G}_2$ as $\{\mathbb{G}_1, \mathbb{G}_2\}^*$, and from $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ as $\{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}^*$. Let $H_1 \in \mathbb{G}_1$, $H_2 \in \mathbb{G}_2$. Then, for example, $(H_2, H_1) \in \{\mathbb{G}_1, \mathbb{G}_2\}^*$ and $(H_1^a, H_2^b, H_2^c, H_1^d) \in \{\mathbb{G}_1, \mathbb{G}_2\}^*$ for $a, b, c, d \in \mathbb{Z}_p$. Following Groth and Sahai [38], equations of the form

$$\prod_i e(X_i, A_i) \prod_j e(B_j, Y_j) \prod_i \prod_j e(X_i, Y_j)^{c_{ij}} = 1$$

for variables $X_i \in \mathbb{G}_1$ and $Y_j \in \mathbb{G}_2$ and constants $B_j \in \mathbb{G}_1$, $A_i \in \mathbb{G}_2$, and $c_{ij} \in \mathbb{Z}_p$ are referred to as PPEs, and equations of the form

$$\prod_i A_i^{x_i} \prod_i Y_i^{b_i} \prod_i \prod_j Y_i^{c_{ij} x_j} = T$$

for variables $x_i \in \mathbb{Z}_p$ and $Y_i \in \mathbb{G}_2$, and constants $b_i, c_{ij} \in \mathbb{Z}_p$ and $A_i, T \in \mathbb{G}_2$ are referred to as multi-scalar multiplication equations (MSEs) in $\mathbb{G}_2$. The MSEs in $\mathbb{G}_1$ are defined similarly. The Groth–Sahai proof system allows one to prove relations represented by those types of equations in a ZK or witness-indistinguishable manner with reasonable efficiency.

Algebraic Algorithms For a bilinear group Λ generated by $\mathcal{G}$, an algorithm Alg that takes group elements $(X_1, \ldots, X_n)$ as input and outputs a group element Y is called algebraic if Alg always "knows" a representation of Y with respect to $(X_1, \ldots, X_n)$. We consider this property with respect to the source groups only. A formal definition for the minimal case, where Alg takes group elements from only one group $\mathbb{G}$ and outputs one element of this group, is provided below.

Definition 1. (*Algebraic Algorithm*) Let Alg be a probabilistic polynomial-time algorithm that takes as an input a bilinear group description Λ generated by $\mathcal{G}$, a tuple of group elements $(X_1, \ldots, X_n) \in \mathbb{G}^n$ for some $n \in \mathbb{N}$, and some auxiliary string $aux \in \{0, 1\}^*$ and outputs a group element $Y \in \mathbb{G}$ and a string ext. Algorithm Alg is algebraic with respect to $\mathcal{G}$ if there is a probabilistic polynomial-time extractor algorithm Ext that takes the same input as Alg (including the random coins) and generates output $(c_1, \ldots, c_n, ext)$

such that for all $\Lambda \xleftarrow{\$} \mathcal{G}(1^\lambda)$, all polynomial sizes n, all $(X_1, \ldots, X_n) \in \mathbb{G}^n$, and all auxiliary strings aux, the following inequality holds:

$$\Pr\left[\begin{array}{l}(Y, ext) \leftarrow \mathsf{Alg}(\Lambda^*, X_1, \ldots, X_n, aux; r)\,; \\ (c_1, \ldots, c_n, ext) \leftarrow \mathsf{Ext}(\Lambda^*, X_1, \ldots, X_n, aux; r)\end{array}\middle| Y \neq \prod X_i^{c_i}\right] \leq \mathsf{negl}(\lambda),$$

where the probability is taken over the choice of the coins r.

It is straightforward to extend this definition to algorithms that output multiple elements of $\mathbb{G}_1$ and $\mathbb{G}_2$ of Λ. We note that all known constructions of structure-preserving primitives are algebraic in the sense defined here. If the considered algorithms were non-algebraic, it is not known how to prove their correct execution with GS proofs.

One may see a similarity between the above definition and the knowledge of exponent assumption (KEA) [9] as both involve an extractor. We emphasize, however, that the algebraic algorithm definition characterizes *honest* algorithms, whereas KEA is an assumption on adversaries.

2.2. *Definitions of Structure-Preserving Deterministic Primitives*

We define the syntax of an SPDP. An SPDP consists of the following five algorithms: Setup, KeyGen, Comp, Prove, and Verify. Besides the parameter generation (Setup), key generation (KeyGen), and main computation algorithm (Comp), they include a proving (Prove) and a verification (Verify) algorithm, which together guarantee that the output value of Comp was computed correctly. We call this the *provability property*. It captures the verifiability notion of some deterministic primitives such as VRFs, USigs, and VUFs. Furthermore, for the deterministic primitives that do not have an inherent verification property such as PRFs and DE, it covers their widely used combinations with non-interactive proof systems. One of the main advantages of structure-preserving primitives and one of the reasons to construct them is their compatibility with existing non-interactive zero-knowledge (NIZK) proof systems.

Definition 2. (*Provable Structure-Preserving Deterministic Primitive*) A provable structure-preserving deterministic primitive Σ_{SPDP} with respect to a bilinear group generator $\mathcal{G}$ consists of five algorithms $\Sigma_{\mathsf{SPDP}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Comp}, \mathsf{Prove}, \mathsf{Verify})$ that operate as follows:

- $CP \xleftarrow{\$} \mathsf{Setup}(\Lambda)$ is a probabilistic algorithm that takes bilinear group description $\Lambda \leftarrow \mathcal{G}(1^\lambda)$ as input and outputs the common parameters CP that include Λ, which is composed of the default generators, but does not contain any other elements of $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$. CP defines the secret key space $\mathcal{SK}$, the public key space $\mathcal{PK}$, the proof space $\mathcal{P}$, and a deterministic polynomial-time computable function $F : \mathcal{SK} \times \mathcal{X} \to \mathcal{Y}$ for some domain $\mathcal{X}$ and range $\mathcal{Y}$. It is implicitly, if not, given to every further algorithm.
- $(PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(CP)$ is a probabilistic key generation algorithm that takes as input CP and outputs a public key $PK \in \mathcal{PK}$ and a secret key $SK \in \mathcal{SK}$.

- $Y \leftarrow \mathsf{Comp}(X, SK)$ is a deterministic algorithm that takes $X \in \mathcal{X}$ and SK as input and outputs $Y = F_{SK}(X) \in \mathcal{Y}$.
- $P \xleftarrow{\$} \mathsf{Prove}(X, SK)$ is a probabilistic algorithm that takes X and SK as input and outputs a proof $P \in \mathcal{P}$.
- $0/1 \leftarrow \mathsf{Verify}(X, Y, P, PK)$ is a deterministic verification algorithm that takes X, Y, P, and PK as input and outputs 1 or 0, representing acceptance or rejection of proof P, respectively.

The following properties must be satisfied:

1. **Structure-Preserving** $\mathcal{PK}$, $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{P}$ are subsets of $\{\mathbb{G}_1, \mathbb{G}_2\}^*$. Furthermore, the verification algorithm is restricted to perform group membership testing, group operations, and evaluation of PPEs over Λ.
2. **Uniqueness** For all λ, $CP \in \mathsf{Setup}(1^\lambda)$ and $(PK, SK) \in \mathsf{KeyGen}(CP)$, there exist no values (X, Y, Y', P, P') such that $Y \neq Y'$ and $\mathsf{Verify}(X, Y, P, PK) = \mathsf{Verify}(X, Y', P', PK) = 1$.
3. **Provability** For all λ, $CP \in \mathsf{Setup}(1^\lambda)$, $(PK, SK) \in \mathsf{KeyGen}(CP)$, $X \in \mathcal{X}$, $Y \leftarrow \mathsf{Comp}(X, SK)$, and $P \in \mathsf{Prove}(X, SK)$, it holds that $\mathsf{Verify}(X, Y, P, PK) = 1$.

Note that the uniqueness is defined with respect to correctly generated keys, which is more relaxed than that in the case of verifiable pseudorandom functions where uniqueness is defined over any keys. As we argue impossibility, however, the relaxed form strengthens our result.

Throughout the paper, we use the convention that the common parameters CP include Λ with the default generators but do not contain any other elements of $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ in Λ. Furthermore, the parameters are given as input to every relevant algorithm. It is also assumed that PK can be recovered from SK. This is justified as we will only consider algebraic KeyGen.

Now, we define two security properties. The *unpredictability* property states that no polynomial-time adversary can predict the output value Y for an input X after having called the Comp and Prove oracles with inputs that are different from X. The *pseudorandomness* property states that no polynomial-time adversary can distinguish Y from a random value.

Definition 3. (*Unpredictability*) A provable structure-preserving deterministic primitive Σ_{SPDP} is unpredictable if for all probabilistic polynomial-time algorithms $\mathcal{A}$ we have that

$$\mathsf{Pr}\left[\begin{array}{l} \Lambda \leftarrow \mathcal{G}(1^\lambda)\ ;\ CP \xleftarrow{\$} \mathsf{Setup}(\Lambda)\ ; \\ (PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(CP)\ ; \\ (X, Y) \leftarrow \mathcal{A}^{\mathsf{Comp}(\cdot, SK), \mathsf{Prove}(\cdot, SK)}(PK) \end{array} \middle| \begin{array}{l} Y = \mathsf{Comp}(X, SK) \wedge \\ X \notin S \end{array} \right] \leq \mathsf{negl}(\lambda)\ ,$$

where S is the set of inputs queried to the oracles Comp and Prove.

Definition 4. (*Pseudorandomness*) A provable structure-preserving deterministic primitive Σ_{SPDP} is pseudorandom if for all probabilistic polynomial-time distinguishers $\mathcal{D} = (\mathcal{D}_1, \mathcal{D}_2)$ we have that

$$\Pr\left[\begin{array}{l|l} \Lambda \leftarrow \mathcal{G}(1^\lambda)\,;\ CP \xleftarrow{\$} \mathsf{Setup}(\Lambda)\,; & \\ (PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(CP)\,; & b = b' \wedge \\ (X, st) \leftarrow \mathcal{D}_1^{\mathsf{Comp}(\cdot,SK),\mathsf{Prove}(\cdot,SK)}(PK)\,; & X \notin S \\ Y_0 \leftarrow F_{SK}(X)\,;\ Y_1 \xleftarrow{\$} \mathcal{Y}\,;\ b \xleftarrow{\$} \{0,1\}\,; & \\ b' \xleftarrow{\$} \mathcal{D}_2^{\mathsf{Comp}(\cdot,SK),\mathsf{Prove}(\cdot,SK)}(Y_b, st) & \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where S is the set of queries to the oracles Comp and Prove.

One can see that Σ_{SPDP} having the unpredictability property is a structure-preserving VUF and Σ_{SPDP} with the pseudorandomness property is a structure-preserving VRF.

2.3. *Inexistence of Structure-Preserving Verifiable Unpredictable Functions*

We now prove that a structure-preserving VUF, as defined in the previous section, cannot exist. Namely, we show that a provable SPDP cannot be unpredictable according to Definition 3 because of its uniqueness and provability properties.

Theorem 1. *Let* $\Sigma_{\mathsf{SPDP}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Comp}, \mathsf{Prove}, \mathsf{Verify})$ *be a provable structure-preserving deterministic primitive with respect to* $\mathcal{G}$*. If the discrete logarithm problem is hard in the groups of* Λ *generated by* $\mathcal{G}$ *and* KeyGen*,* Comp*, and* Prove *are algebraic algorithms with respect to* $\mathcal{G}$*, then* Σ_{SPDP} *is not unpredictable.*

Proof. For simplicity, we first consider a symmetric bilinear setting ($\Lambda = \Lambda_{\mathsf{sym}}$), where $\mathcal{PK}, \mathcal{X}, \mathcal{Y}, \mathcal{P} \subset \{\mathbb{G}\}^*$. Furthermore, we consider X to consist only of a single group element. We then show that the same result holds for the input being a tuple of group elements from $\mathbb{G}$ and also in the asymmetric setting, for both Type 2 pairings (where an efficiently computable homomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$ exists and there is no efficiently computable homomorphism from $\mathbb{G}_1$ to $\mathbb{G}_2$) and Type 3 pairings (where there are no efficiently computable homomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$) [34].

The outline of the proof is as follows. First, in Lemma 1 we show that because of the provability and uniqueness properties of Σ_{SPDP}, as specified in Definition 2, the output of Comp must have a particular format, namely $\mathsf{Comp}(X, SK) = (G^{a_1}X^{b_1}, \ldots, G^{a_\ell}X^{b_\ell})$ for some (secret) constants $a_1, \ldots, a_\ell, b_1, \ldots, b_\ell \in \mathbb{Z}_p$. Then, in Lemma 2, we prove that, if the output of Comp has this format, then the unpredictability property (cf. Definition 3) does not hold for Σ_{SPDP}. This means that no structure-preserving VUF can exist. □

Lemma 1. *Let* $\Sigma_{\mathsf{SPDP}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Comp}, \mathsf{Prove}, \mathsf{Verify})$ *be a structure-preserving deterministic primitive with respect to* $\mathcal{G}$ *according to Definition 2. If* KeyGen*,* Comp*, and* Prove *are algebraic algorithms with respect to* $\mathcal{G}$*, and the discrete logarithm problem is hard in the base group of* Λ *generated by* $\mathcal{G}$*, then, with overwhelming prob-*

ability taken over the random coins used in the algorithms in Σ_{SPDP}, $\mathsf{Comp}(X, SK) = (Y_1, \ldots, Y_\ell) = (G^{a_1} X^{b_1}, \ldots, G^{a_\ell} X^{b_\ell})$ *holds for some constants* $a_1, \ldots, a_\ell, b_1, \ldots, b_\ell \in \mathbb{Z}_p$.

Proof. Let $\Lambda \leftarrow \mathcal{G}(1^\lambda)$ and $CP \leftarrow \mathsf{Setup}(\Lambda)$. Fix $(PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(CP)$, where $PK \subset \{\mathbb{G}\}^*$. Let $x \xleftarrow{\$} \mathbb{Z}_p$ and $X = G^x$.

First, note that because Comp, Prove, and KeyGen are algebraic algorithms, their outputs can be expressed as

$$\mathsf{Comp}(X, SK) = Y = (Y_1, \ldots, Y_\ell) \text{ with } Y_i = G^{a_i} X^{b_i}, \tag{1}$$

$$\mathsf{Prove}(X, SK) = P = (P_1, \ldots, P_n) \text{ with } P_j = G^{u_j} X^{v_j}, \text{ and} \tag{2}$$

$$PK = (S_1, \ldots, S_m) \text{ with } S_f = G^{s_f}, \tag{3}$$

where $a_i = H_{1,i}(X, SK)$, $b_i = H_{2,i}(X, SK)$, $u_j = H_{3,j}(X, SK, r)$, and $v_j = H_{4,j}(X, SK, r)$ for some arbitrary functions $H_{\ell,m} : \{0,1\}^* \to \mathbb{Z}_p$ where r is the randomness used by the Prove algorithm. We note that a_i, b_i, u_j, and v_j can depend on X in an arbitrary manner, but, as Comp and Prove are algebraic, one can extract a_i, b_i, u_j, and v_j as values from $\mathbb{Z}_p$ using the extractors of the algorithms Comp and Prove.

Second, recall that, according to Definition 2, the verification algorithm consists of PPEs. Let the kth PPE used in the verification algorithm be

$$\prod_{f=1}^{m} e\Bigg(S_f, X^{c_{k,1,f}} \prod_{t=1}^{m} S_t^{c_{k,2,f,t}} \prod_{i=1}^{\ell} Y_i^{c_{k,3,f,i}} \prod_{j=1}^{q} P_j^{c_{k,4,f,j}}\Bigg) \cdot \prod_{q=1}^{n} e\Bigg(P_q, \prod_{j=1}^{n} P_j^{c_{k,5,q,j}}\Bigg) \cdot$$
$$\cdot\, e\Bigg(X, X^{c_{k,6}} \prod_{i=1}^{\ell} Y_i^{c_{k,7,i}} \prod_{j=1}^{q} P_j^{c_{k,8,j}}\Bigg) \cdot \prod_{w=1}^{\ell} e\Bigg(Y_w, \prod_{i=1}^{\ell} Y_i^{c_{k,9,w,i}} \prod_{j=1}^{n} P_j^{c_{k,10,w,j}}\Bigg) = 1.$$

The intuition behind the proof is as follows. We note that Comp should perform the computation without necessarily "knowing" the discrete logarithm of the input; otherwise, one can use Comp to solve the discrete logarithm for X. Now, one can see that the relation in the exponents of the kth PPE for the tuple (X, Y, P, PK) induces a polynomial $Q_k(x)$ in the discrete logarithm $x = \log_G X$. Basically, we can rewrite the kth PPE as $e(G, G)^{Q_k(x)} = 1$. First, we prove that $Q_k(x)$ is a trivial function; otherwise, it would be possible to solve the discrete logarithm problem for the given X by solving Q_k. Second, we show that if Q_k is trivial, then, by the uniqueness property, a_i and b_i are constants. Let a_i, b_i, u_j, and v_j be the values computed for one specific $X : Y_i = G^{a_i} X^{b_i}$, $P_j = G^{u_j} X^{v_j}$, and $\mathsf{Verify}(PK, X, Y, P) = 1$. Proposition 2 shows that these values can be reused to compute a correct $\tilde{Y}$ for any other $\tilde{X} \in \mathcal{X}$. Therefore, if $\tilde{Y}_i$ is computed as $G^{a_i} \tilde{X}^{b_i}$ and $\tilde{P}_j$ as $G^{u_j} \tilde{X}^{v_j}$, instead of using the normal computation procedures, then $(\tilde{X}, \tilde{Y}, \tilde{P}, PK)$ is also accepted by the verification algorithm due to the triviality of Q_k. Then, from the uniqueness and provability properties, it follows that a_i, b_i are the only valid values, i.e., constants.

Now, we provide the proof in detail. First, we prove that all polynomials Q_k induced by the verification PPEs, as described above, are constants with overwhelming probability. □

Proposition 1. *If the discrete logarithm problem in the base group of Λ is hard, then Q_k is a trivial function.*

Proof. The proof is done by constructing a reduction algorithm R that takes as an input a group description $\Lambda = (p, \mathbb{G}, \mathbb{G}_T, e, G)$ generated by a group generator $\mathcal{G}(1^\lambda)$ and a random element $X \in \mathbb{G}$ and outputs $x \in \mathbb{Z}_p$ that satisfies $X = G^x$ with high probability.

The R works as follows. It first takes Λ as an input and sets the common parameters $CP = \Lambda$. It then runs $\mathsf{KeyGen}(CP)$, $\mathsf{Comp}(X, SK)$, and $\mathsf{Prove}(X, SK)$ for the given X. It also runs the corresponding extractors for KeyGen, Comp, and Prove. The extractor for KeyGen outputs representations s_f that satisfy $S_f = G^{s_f}$ with overwhelming probability. Similarly, the extractor for Comp outputs representations a_i and b_i, such that $Y_i = G^{a_i} X^{b_i}$, and the extractor for Prove outputs u_j and v_j, such that $P_j = G^{u_j} X^{v_j}$, as concrete values in $\mathbb{Z}_p$.

This set of extracted exponents s_f, a_i, b_i, u_j, and v_j induces a quadratic formula Q_k in the exponents of the kth pairing product verification equation. Let us call the variable of this exponent equation $\tilde{x}$, then we can write the kth PPE as $e(G, G)^{Q_k(\tilde{x})} = 1$. Given the representations, R can compute $Q_k(\tilde{x}) = d_2\tilde{x}^2 + d_1\tilde{x} + d_0$ in $\mathbb{Z}_p$. The condition that $Q_k(\tilde{x})$ is non-trivial guarantees that $d_2 \neq 0$ or $d_1 \neq 0$. However, R can then solve $Q_k(\tilde{x}) = 0$ for $\tilde{x}$ with standard algebra. Due to the provability property, x is one of the possible solutions to $\tilde{x}$. Therefore, if the equation is non-trivial, we can solve this equation for $\tilde{x}$ and obtain the discrete logarithm of $X : \tilde{x} = x$. If the discrete logarithm problem is hard in the base group of Λ, then Q_k must be trivial. □

Now, we show that if Q_k is trivial, then, by the provability and uniqueness properties, a_i and b_i are constants.

Proposition 2. *Fix (PK, SK, X) and let $a_i \leftarrow H_{1,i}(X, SK)$, $b_i \leftarrow H_{2,i}(X, SK)$, $u_j \leftarrow H_{3,j}(X, SK, r)$, and $v_j \leftarrow H_{4,j}(X, SK, r)$. If all the relations in the exponents of the PPEs are trivial, then, for any $\tilde{X} \in \mathbb{G}$, $\tilde{Y} = (\tilde{Y}_1, \ldots, \tilde{Y}_\ell)$ with $\tilde{Y}_i = G^{a_i}\tilde{X}^{b_i}$ and $\tilde{P} = (\tilde{P}_1, \ldots, \tilde{P}_n)$ with $\tilde{P}_j = G^{u_j}\tilde{X}^{v_j}$, it holds that $(\tilde{X}, \tilde{Y}, \tilde{P}, PK)$ will be accepted by the verification algorithm.*

Proof. Consider fixed (PK, SK, X), any $\tilde{X} \in \mathbb{G}$, and $\tilde{Y}$ and $\tilde{P}$ computed from $\tilde{X}$ as specified in the proposition. Note that the verification algorithm only evaluates PPEs and performs group membership tests. First, all group membership tests are clearly successful for the tuple $(\tilde{X}, \tilde{Y}, \tilde{P}, PK)$. Since all polynomials Q_k are trivial and due to the way in which $\tilde{Y}$ and $\tilde{P}$ are defined, it holds that the result of evaluating the kth PPE will be the same for any tuple $(\tilde{X}, \tilde{Y}, \tilde{P}, PK)$. Therefore, $\mathsf{Verify}(\tilde{X}, \tilde{Y}, \tilde{P}, PK)$ should output the same value for every $\tilde{X} \in \mathbb{G}$. Now, considering the case in which $\tilde{X} = X$ we have $\tilde{Y} = (\tilde{Y}_1, \ldots, \tilde{Y}_\ell)$ with $\tilde{Y}_i = G^{a_i}X^{b_i}$ and $\tilde{P} = (\tilde{P}_1, \ldots, \tilde{P}_n)$ with $\tilde{P}_j = G^{u_j}X^{v_j}$. However, due to the correctness of the extractors of Comp and Prove, these $\tilde{Y}$ and $\tilde{P}$ are exactly the outputs of $\mathsf{Comp}(X, SK)$ and $\mathsf{Prove}(X, SK)$. Therefore, by the provability

property, it holds that $\mathsf{Verify}(X, \tilde{Y}, \tilde{P}, PK) = 1$ for $\tilde{X} = X$; thus, for any $\tilde{X} \in \mathbb{G}$, $\mathsf{Verify}(\tilde{X}, \tilde{Y}, \tilde{P}, PK) = 1$ also. □

Now, for an arbitrary $\tilde{X} \in \mathbb{G}$, consider the tuple $(PK, SK, \tilde{X}, \tilde{Y}, \tilde{P}, a_1, \ldots, a_\ell, b_1, \ldots, b_\ell)$ of values as defined above. The $\tilde{P}$ is a valid proof for $(\tilde{X}, \tilde{Y})$; thus, the uniqueness property guarantees that there is no other $\hat{Y} \neq \tilde{Y}$ for which there is a valid proof that $\hat{Y}$ is the output corresponding to $\tilde{X}$. However, the provability property guarantees that for $(\tilde{X}, \mathsf{Comp}(\tilde{X}, SK))$ there is a valid proof of correctness. Hence, for any $\tilde{X} \in \mathbb{G}$, it holds that

$$\mathsf{Comp}(\tilde{X}, SK) = \tilde{Y} = (\tilde{Y}_1, \ldots, \tilde{Y}_\ell) = (G^{a_1}\tilde{X}^{b_1}, \ldots, G^{a_\ell}\tilde{X}^{b_\ell}).$$

□

Lemma 2. *Suppose that* $\Sigma_{\mathsf{SPDP}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Comp}, \mathsf{Prove}, \mathsf{Verify})$ *is a provable Structure-Preserving Deterministic Primitive such that* $\mathsf{Comp}(X, SK) = (Y_1, \ldots, Y_\ell) = (G^{a_1}X^{b_1}, \ldots, G^{a_\ell}X^{b_\ell})$ *for some constants* $a_1, \ldots, a_\ell, b_1, \ldots, b_\ell \in \mathbb{Z}_p$. *Then,* Σ_{SPDP} *does not satisfy the unpredictability requirement from Definition 3.*

Proof. Pick $\hat{X}, \tilde{X}$ and define $\overline{X}$, such that $\overline{X} = \hat{X}^2/\tilde{X} \notin \{\hat{X}, \tilde{X}\}$. Then, an adversary that learns

$$\mathsf{Comp}(\hat{X}, SK) = (\hat{Y}_1, \ldots, \hat{Y}_\ell) = (G^{a_1}\hat{X}^{b_1}, \ldots, G^{a_\ell}\hat{X}^{b_\ell}), \text{ and}$$
$$\mathsf{Comp}(\tilde{X}, SK) = (\tilde{Y}_1, \ldots, \tilde{Y}_\ell) = (G^{a_1}\tilde{X}^{b_1}, \ldots, G^{a_\ell}\tilde{X}^{b_\ell})$$

can compute the value of $\mathsf{Comp}(\overline{X}, SK)$ as:

$$\left(\frac{\hat{Y}_1^2}{\tilde{Y}_1}, \ldots, \frac{\hat{Y}_\ell^2}{\tilde{Y}_\ell}\right) = \left(\frac{G^{2a_1}\hat{X}^{2b_1}}{G^{a_1}\tilde{X}^{b_1}}, \ldots, \frac{G^{2a_\ell}\hat{X}^{2b_\ell}}{G^{a_\ell}\tilde{X}^{b_\ell}}\right) = \left(G^{a_1}\left(\frac{\hat{X}^2}{\tilde{X}}\right)^{b_1}, \ldots, G^{a_\ell}\left(\frac{\hat{X}^2}{\tilde{X}}\right)^{b_\ell}\right)$$
$$= \left(G^{a_1}\overline{X}^{b_1}, \ldots, G^{a_\ell}\overline{X}^{b_\ell}\right) = \mathsf{Comp}(\overline{X}, SK);$$

therefore, Σ_{SPDP} is not unpredictable. □

We next show that the same result holds for inputs being a tuple of group elements from $\mathbb{G}$.

X is a Tuple of Group Elements Both Lemmas 1 and 2 can be easily modified to the case in which X consists of more than one (say t) group element as follows. The reduction algorithm, after receiving the discrete logarithm challenge X_1, will select $t-1$ random exponents $x_2, \ldots, x_t$ and fix X_i as G^{x_i} for $i = 2, \ldots, t$. Then, the lemmas use the first group element X_1 in the place of the original X. Note that in the computation of Y_j and P_j, the exponents corresponding to $X_2, \ldots, X_t$ can be incorporated into $H_{1,j}(X, SK)$ and $H_{3,j}(X, SK, r)$ since the prover "knows" $x_2, \ldots, x_t$. If the quadratic equations $Q_k(\tilde{x}_1)$ in the exponents of the PPEs are not trivial, then the first element of the input can be used to solve the discrete logarithm problem; otherwise, supposing that

the uniqueness and provability properties hold, the elements of the output will be of the form $\tilde{Y}_i = G^{a_i}\tilde{X}_1^{b_i}$ (for the fixed values $x_2, \ldots, x_t$) and this can be used to break the unpredictability property by asking two queries in which only the first elements of the inputs are different (i.e., $\hat{X}_1$ and $\tilde{X}_1$) Then, learning the output corresponding to a third input that has $\overline{X}_1 = \hat{X}_1^2/\tilde{X}_1$ as the first element and has the remaining elements equal to the ones in the oracle queries.

Asymmetric Bilinear Groups Setting Lemmas 1 and 2 can be generalized to the asymmetric setting as well. We consider both Type 2 and Type 3 pairings. If $\mathcal{X}$ consists of t group elements, we choose $t-1$ random exponents $x_2, \ldots, x_t$ and fix X_i as $G_1^{x_i}$ if the ith input element is in group $\mathbb{G}_1$, or $G_2^{x_i}$ if the ith input element is in group $\mathbb{G}_2$. Then, either some quadratic equation $Q_k(\tilde{x}_1)$ in the exponents of the PPEs is not trivial in x_1, and this can be used to solve the discrete logarithm problem in the base group in which X_1 is contained, or one of the three security properties (provability, uniqueness, and unpredictability) does not hold.

In the case of Type 3 pairings, where there are no efficiently computable homomorphisms between the groups, each Y_j (let $\mathbb{G}_c$ denote the group in which it is and G_c its generator) is of the form

$$Y_j = G_c^{H_{1,j}(X,SK)} X_1^{H_{2,j}(X,SK)} ,$$

where $H_{2,j}(X, SK) = 0$ if X_1 and Y_j are not in the same group and each P_j (that is in the group $\mathbb{G}_c$) is of the form

$$P_j = G_c^{H_{3,j}(X,SK,r)} X_1^{H_{4,j}(X,SK,r)} ,$$

where $H_{4,j}(X, SK, r) = 0$ if X_1 and P_j are not in the same group. In both cases, the exponents corresponding to $X_2, \ldots, X_t$ are incorporated into $H_{1,j}(X, SK)$ and $H_{3,j}(X, SK, r)$. Then, the argument continues, as in the previous case.

In the case of Type 2 pairings, there is an efficiently computable homomorphism $\phi : \mathbb{G}_2 \to \mathbb{G}_1$. Then, an element Y_j of the output (or an element P_j of the proof) that is in group $\mathbb{G}_1$ can depend on both group generators and on X_1 or its mapping $\phi(X_1)$ into $\mathbb{G}_1$. Namely, if $X_1 \in \mathbb{G}_1$, then $Y_j, P_j \in \mathbb{G}_1$ have the form:

$$\begin{aligned} Y_j &= G_1^{H_{1,j}(X,SK)} X_1^{H_{2,j}(X,SK)} \phi(G_2)^{H_{5,j}(X,SK)} , \\ P_j &= G_1^{H_{3,j}(X,SK,r)} X_1^{H_{4,j}(X,SK,r)} \phi(G_2)^{H_{6,j}(X,SK,r)} , \end{aligned}$$

and $Y_j, P_j \in \mathbb{G}_2$ have the form:

$$\begin{aligned} Y_j &= G_2^{H_{5,j}(X,SK)} , \\ P_j &= G_2^{H_{6,j}(X,SK,r)} . \end{aligned}$$

If $X_1 \in \mathbb{G}_2$, then $Y_j, P_j \in \mathbb{G}_1$ have the form:

$$Y_j = G_1^{H_{1,j}(X,SK)} \phi(X_1)^{H_{2,j}(X,SK)} \phi(G_2)^{H_{5,j}(X,SK)} ,$$
$$P_j = G_1^{H_{3,j}(X,SK,r)} \phi(X_1)^{H_{4,j}(X,SK,r)} \phi(G_2)^{H_{6,j}(X,SK,r)} ,$$

and $Y_j, P_j \in \mathbb{G}_2$ have the form:

$$Y_j = X_1^{H_{2,j}(X,SK)} G_2^{H_{5,j}(X,SK)} ,$$
$$P_j = X_1^{H_{4,j}(X,SK,r)} G_2^{H_{6,j}(X,SK,r)} .$$

Then, we should have $H_{1,j}(X, SK) = a_j$, $H_{2,j}(X, SK) = b_j$, and $H_{5,j}(X, SK) = z_j$ for constants a_j, b_j, and z_j if the provability and uniqueness hold. However, in this case the unpredictability does not hold for the same reasons as before.

Putting Lemmas 1 and 2 together completes the proof of Theorem 1. □

2.4. *Extension to Quasi-Deterministic Functions*

Consider a quasi-deterministic functions where the uniqueness condition is relaxed so that for each input value there are at most $poly(\lambda)$ output values for which a valid proof exists. A simple, but perhaps artificial, example would be a disjunction of polynomial number of deterministic functions whose output and proof consist of those from one of the underlying functions applied to the same input. The previous result can be extended to quasi-deterministic functions. It is possible to prove an analogue of Proposition 2. The idea is that for any fixed (PK, SK, X) and extracted values $a_i \leftarrow H_{1,i}(X, SK, r)$, $b_i \leftarrow H_{2,i}(X, SK, r)$, $u_j \leftarrow H_{3,j}(X, SK, r)$, and $v_j \leftarrow H_{4,j}(X, SK, r)$, if all the relations in the exponents of the PPEs are trivial, then, for any $\tilde{X} \in \mathbb{G}$, $\tilde{Y} = (\tilde{Y}_1, \ldots, \tilde{Y}_\ell)$ with $\tilde{Y}_i = G^{a_i} \tilde{X}^{b_i}$ and $\tilde{P} = (\tilde{P}_1, \ldots, \tilde{P}_n)$ with $\tilde{P}_j = G^{u_j} \tilde{X}^{v_j}$, it holds that $(\tilde{X}, \tilde{Y}, \tilde{P}, PK)$ will be accepted by the verification algorithm. However, since there are at most $poly(\lambda)$ outputs with valid proofs for each input, it follows that there are at most $poly(\lambda)$ distinct sets of extracted values (a_i, b_i). Therefore, an adversary, after making $poly(\lambda) + 1$ queries to the $\mathsf{Comp}(\cdot)$ oracle, has a non-negligible probability of randomly picking two outputs that use the same pair of exponents, in which case it can break the unpredictability property in the same way as done in Lemma 2.

2.5. *Extension to "Non-strictly" Structure-Preserving Primitives*

The definition in the previous sections captures so-called "strictly" SPDPs, i.e., $\mathcal{PK}$ and $\mathcal{Y}$ can contain only source group elements. Let us discuss the case of structure-preserving primitives that also have target group elements in their public key space and/or their range. A target group element can be represented by two source group elements using pairing randomization techniques [3] or even deterministically, by fixing the randomizing exponents. By this, the provability property can be preserved. Now, the question is: If the uniqueness property holds, can the output be unpredictable? In

this section, we argue that our impossibility result can be extended to some cases of "non-strictly" SPDPs, formally defined below:

Definition 5. (*"Non-strictly" Structure-Preserving Deterministic Primitive*) Let $\Sigma_{\mathsf{SPDP}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Comp}, \mathsf{Prove}, \mathsf{Verify})$ be a structure-preserving deterministic primitive according to Definition 2, except that the range of Comp and KeyGen can also contain target group elements ($\mathcal{Y}, \mathcal{PK} \subset \{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}^*$). Then, Σ_{SPDP} is called a "non-strictly" structure-preserving deterministic primitive.

We first extend the notion of the algebraic algorithms from Definition 1 so that it operates in all groups of Λ. Then, one can use the extractors of KeyGen and Comp to also extract representations of target group elements output by them. A formal definition for the minimal case of Type 3 asymmetric groups and outputs consisting of one element of each group is provided below.

Definition 6. (*Extended Algebraic Algorithms*) Let Alg be a probabilistic polynomial-time algorithm that takes as an input Λ of a Type 3 asymmetric bilinear group generated by $\mathcal{G}$, X consisting of two tuples of group elements $(X_{1,1}, \ldots, X_{1,n}) \in \{\mathbb{G}_1\}^n$ and $(X_{2,1}, \ldots, X_{2,m}) \in \{\mathbb{G}_2\}^m$ for some $n, m \in \mathbb{N}$, and some auxiliary string $aux \in \{0,1\}^*$ and outputs group elements $Y \in \mathbb{G}_1$, $W \in \mathbb{G}_2$, and $Z \in \mathbb{G}_T$ and a string ext. The algorithm Alg is algebraic with respect to $\mathcal{G}$ if there is a probabilistic polynomial-time extractor algorithm Ext that takes the same input as Alg (including the random coins) and generates output $(c = (c_1, \ldots, c_n), d = (d_1, \ldots, d_m), f = (f_1, \ldots, f_{nm}), ext)$ such that for all $\Lambda \xleftarrow{\$} \mathcal{G}(1^\lambda)$, all polynomial-sized n, m, all $X = (X_{1,1}, \ldots, X_{1,n}, X_{2,1}, \ldots, X_{2,m}) \in \{\mathbb{G}_1\}^n \times \{\mathbb{G}_2\}^m$, and all auxiliary strings aux, the following inequality holds

$$\mathsf{Pr}\left[\begin{array}{l|l} (Y, W, Z, ext) \leftarrow \mathsf{Alg}(\Lambda^*, X, aux; r)\,; & Y \neq \prod_{i=1}^{n} X_{1,i}^{c_i} \vee \\ (c, d, f, ext) \leftarrow \mathsf{Ext}(\Lambda^*, X, aux; r) & W \neq \prod_{j=1}^{m} X_{2,j}^{d_j} \vee \\ & Z \neq \prod_{i=1}^{n}\prod_{j=1}^{m} e(X_{1,i}, X_{2,j})^{f_{(j-1)n+i}} \end{array}\right] \leq \mathsf{negl}(\lambda)\ ,$$

where the probability is taken over the choice of r.

Similarly to Definition 1, this definition can be extended to algorithms that output multiple elements of the groups of Λ and for other types of groups. Adaptation to Type 1 is trivial by considering $X_{1,i} = X_{2,i}$, and to Type 2 is also straightforward by letting $X_{1,k+i} = \phi(X_{2,i})$ for some $k < n$ and all $i \in \{1, \ldots, m\}$.

Below, we show that if the provability and uniqueness properties (according to Definition 5) hold, then the unpredictability property does not hold.

Theorem 2. *Let* $\Sigma_{\mathsf{SPDP}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Comp}, \mathsf{Prove}, \mathsf{Verify})$ *be a "non-strictly" provable structure-preserving deterministic primitive with respect to bilinear group generator* $\mathcal{G}$*. If the discrete logarithm problem is hard in the source groups generated by* $\mathcal{G}$ *and* KeyGen*,* Comp*, and* Prove *are algebraic with respect to* $\mathcal{G}$*, as defined in Definition 6, then* Σ_{SPDP} *is not unpredictable.*

Proof. We first address the case of Type 3 bilinear groups. The outline of the proof is the same as that for Theorem 1. Without loss of generality, we consider the case in which X consists of only one element $X_1 \in \mathbb{G}_1$ (since the adversary can set all inputs except X_1 to 1). We show that for any Σ_{SPDP} that is provable and has the uniqueness property according to Definition 2, the output of Comp must have a particular format.

Fix $(PK, SK) \leftarrow \mathsf{KeyGen}(CP)$, where the public key consists of both source and target group elements, i.e., $PK \subset \{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}^*$. Let $X = X_1$ be the input. Since Comp is deterministic and KeyGen, Comp, and Prove are all algebraic algorithms over Λ, their respective outputs can be represented as

$$\mathsf{Comp}(X, SK) = Y = (Y_1, \ldots, Y_\ell, W_1, \ldots, W_{\ell'}, Z_1, \ldots, Z_{\ell''}), \text{ and}$$
$$\mathsf{Prove}(X, SK) = P = (P_1, \ldots, P_n, P'_1, \ldots, P'_{n'}),$$

where

$$Y_i = G_1^{a_{1,i}} X_1^{b_{1,i}}, W_i = G_2^{a_{2,i}}, Z_i = e(G_1, G_2)^{a'_i} e(X_1, G_2)^{b'_i},$$
$$P_i = G_1^{u_i} X_1^{v_i}, P'_i = G_2^{u'_i},$$

and one can extract the exponents $a_{1,i}$, $b_{1,i}$, $a_{2,i}$, a'_i, b'_i, u_i, v_i, and u'_i as values in $\mathbb{Z}_p$ using the extractors of the algorithms KeyGen, Comp, and Prove. Recall that the verification algorithm consists of evaluations of PPEs. The proof works very similarly to that of Lemma 1. For $X = X_1 \in \mathbb{G}_1$, with unknown exponent $x_1 = \log_{G_1} X_1$, the relation among the exponents of the kth PPE for the tuple (X, Y, P, PK) with respect to the base $e(G_1, G_2)$ induces a polynomial $Q_k(x_1)$ in x_1. Thus, the kth PPE can be written as $e(G_1, G_2)^{Q_k(x_1)} = 1$. As before, $Q_k(x_1)$ is a trivial function since otherwise it would be possible to solve the discrete logarithm problem for the given X_1 by solving Q_k. Let $a_{1,i}$, $b_{1,i}$, $a_{2,i}$, a'_i, b'_i, u_i, v_i, and u'_i be the exponents obtained from one specific computation of Y_i W_i, Z_i, and P_i with input X_1. Then, $\tilde{Y}_i$, $\tilde{W}_i$, $\tilde{Z}_i$, and $\tilde{P}_i$, computed for an arbitrary input $\tilde{X}_1 \in \mathbb{G}_1$ with these fixed exponents, pass the verification due to the triviality of Q_k. From the uniqueness property, it now follows that $a_{1,i}$, $b_{1,i}$, $a_{2,i}$, a'_i, and b'_i are the only valid values with respect to $\tilde{X}_1$. Thus, they are constants.

Next, we prove that if the output of Comp has this format, then the unpredictability property (Definition 3) does not hold for Σ_{SPDP}. We do so by constructing an adversary breaking the unpredictability property as follows. Select $\hat{X}_1, \tilde{X}_1 \in \mathbb{G}_1$, set $\hat{X} = \hat{X}_1$ and $\tilde{X} = \tilde{X}_1$, and define $\overline{X}$ such that $\overline{X}_1 = \hat{X}_1^2 / \tilde{X}_1 \notin \{\hat{X}_1, \tilde{X}_1\}$. As we already proved in Lemma 2 that the unpredictability does not hold for source group elements, we assume for simplicity that the output consists only of the target group elements. The adversary that learns $\mathsf{Comp}(\hat{X}, SK) = (\hat{Z}_1, \ldots, \hat{Z}_{\ell''})$ with $\hat{Z}_i = e(G_1, G_2)^{a'_i} e(\hat{X}_1, G_2)^{b'_i}$ and $\mathsf{Comp}(\tilde{X}, SK) = (\tilde{Z}_1, \ldots, \tilde{Z}_{\ell''})$ with $\tilde{Z}_i = e(G_1, G_2)^{a'_i} e(\tilde{X}_1, G_2)^{b'_i}$ can compute the value of $\mathsf{Comp}(\overline{X}, SK) = (\overline{Z}_1, \ldots, \overline{Z}_{\ell''})$ as $\left(\frac{\hat{Z}_1^2}{\tilde{Z}_1}, \ldots, \frac{\hat{Z}_{\ell''}^2}{\tilde{Z}_{\ell''}} \right)$ because we have that

$$\frac{\hat{Z}_i^2}{\tilde{Z}_i} = \frac{e(G_1, G_2)^{2a'_i}\, e(\hat{X}_1, G_2)^{2b'_i}}{e(G_1, G_2)^{a'_i}\, e(\tilde{X}_1, G_2)^{b'_i}} = e(G_1, G_2)^{a'_i}\, e\left(\frac{\hat{X}_1^2}{\tilde{X}_1}, G_2\right)^{b'_i} = \overline{Z}_i.$$

Therefore, Σ_{SPDP} is also not unpredictable for the target group elements.

We next proceed to the case of Type 1. Let X be a group element in $\mathbb{G}$. Then, as for the above case, if the discrete logarithm problem is hard in $\mathbb{G}$, we have $\mathsf{Comp}(X, SK) = (Y_1, \ldots, Y_\ell, Z_1, \ldots, Z_{\ell'})$ with $Y_i = G^{a_i} X^{b_i}$, and $Z_i = e(G, G)^{a'_i} e(X, G)^{b'_i} e(X, X)^{c'_i}$ for constants a_i, b_i, a'_i, b'_i, and c'_i. We construct an adversary that breaks the unpredictability properties as follows. It makes three queries $X = 1$, $X = G$, and $X = G^2$ and receives the outputs $e(G, G)^{a'_i}$, $e(G, G)^{a'_i + b'_i + c'_i}$, and $e(G, G)^{a'_i + 2b'_i + 4c'_i}$. The adversary can then compute $e(G, G)^{a'_i}$, $e(G, G)^{b'_i}$, and $e(G, G)^{c'_i}$, which are sufficient to compute

$$Z_i = e(G, G)^{a'_i} e(G^x, G)^{b'_i} e(G^x, G^x)^{c'_i}$$

for any x. Thus, the scheme is not unpredictable.

Finally, regarding Type 2 bilinear groups, we have two cases. First, if the scheme accepts inputs containing at least one element from $\mathbb{G}_1$, then the proof for Type 3 groups applies since we can argue in exactly the same manner by setting all $\mathbb{G}_2$ elements in inputs to 1. Second, if the scheme accepts inputs containing at least one element from $\mathbb{G}_2$, the proof for Type 1 applies since we can move the element to $\mathbb{G}_1$ using homomorphism ϕ. □

Note that the same holds even if we further allow proof P to include elements in the target group since their source group representations can be extracted and we can argue the triviality of $Q_k(x)$ in the same manner.

3. Application to Concrete Schemes

In this section, we show how the definition of an abstract provable SPDP given in Sect. 2 relates to the definitions of structure-preserving VRFs and USigs. We show that the security properties of an SPDP are necessary conditions for any VRF or USig to be secure. Note that the requirements are necessary conditions, but may not be sufficient, e.g., in the case of VRF, pseudorandomness is a stronger requirement than unpredictability. We also discuss how the SPDP definition relates to structure-preserving PRFs and deterministic encryption.

3.1. *Impossibility of Structure-Preserving Unique Signatures*

Let $\Sigma_{\mathsf{USig}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ be a structure-preserving signature scheme with respect to group generator $\mathcal{G}$ [3]. The setup algorithm Setup takes Λ generated by $\mathcal{G}$ and outputs a CP. The key generation algorithm KeyGen takes the CP and out-

puts a key pair PK and SK. The signing algorithm Sign takes SK and message X as input and outputs a signature Y. The verification algorithm Verify takes PK, X, and Y as input and outputs 1 or 0. Structure preservation requires that PK, X, and Y consist only of source group elements of Λ. The signature scheme must satisfy the standard notions of correctness and existential unforgeability against adaptively chosen message attacks. According to Lyskanskaya [46], Σ is a *unique* signature scheme if the signing function Sign is deterministic and the following uniqueness property holds: There are no two distinct signatures Y and Y' accepted by the verification algorithm with respect to the same message X. Namely, there is no tuple (PK, X, Y, Y') such that $Y \neq Y'$ and $\mathsf{Verify}(X, Y, PK) = \mathsf{Verify}(X, Y', PK) = 1$ holds. We then state the following impossibility.

Theorem 3. *If the discrete logarithm problem is hard in the groups of Λ generated by $\mathcal{G}$, there is no structure-preserving unique signature scheme whose* KeyGen *and* Sign *are algebraic algorithms with respect to $\mathcal{G}$ and that is existentially unforgeable against adaptive chosen message attacks.*

To prove Theorem 3, it is sufficient to show that unforgeable structure-preserving unique signature schemes are unpredictable SPDPs and apply Theorem 1.

Lemma 3. *A structure-preserving unique signature scheme that is existentially unforgeable against adaptive chosen message attacks is an unpredictable structure-preserving deterministic primitive.*

Proof. We first verify the syntactical consistency of a structure-preserving signature scheme Σ_{USig} as an SPDP Σ_{SPDP}. Observe that the algorithms Setup and KeyGen of Σ_{USig} are exactly the same as those of Σ_{SPDP}. The Sign algorithm of Σ_{USig} corresponds to the Comp algorithm of a SPDP. There is no algorithm in Σ_{USig} that corresponds to Prove of Σ_{SPDP}. Therefore, we can think of a constant function that outputs a constant P to be Prove. Then, Verify of Σ_{USig} can be seen as Verify of Σ_{SPDP} that ignores input P. With the above syntactical correspondence, one can inspect that the correctness of Σ_{USig} implies the provability, as in Definition 2. The uniqueness of Σ_{USig} is also the same as that in Definition 2.

It remains to show that existential unforgeability against adaptive chosen message attacks implies the unpredictability property. Suppose that the adversary attacking the unpredictability, as in Definition 3, succeeds in computing Y such that $Y = \mathsf{Comp}(X, SK)$ holds for a fresh X. Then, such (X, Y) satisfies $\mathsf{Verify}(X, Y, P, PK) = 1$ for the above constant P. It then satisfies $\mathsf{Verify}(X, Y, PK) = 1$ for Verify of Σ_{USig} due to the correspondence of Verify in Σ_{USig} and as Σ_{SPDP}. Accordingly, (X, Y) is a valid forgery breaking the existential unforgeability of Σ_{USig}. □

3.2. *Impossibility of Structure-Preserving Verifiable Random Functions*

A structure-preserving VRF consists of algorithms $\Sigma_{\mathsf{VRF}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Comp}, \mathsf{Prove}, \mathsf{Verify})$ whose syntax is exactly the same as that of an SPDP. Namely, its public key space, domain, range, and proof space consist only of source group elements

and it satisfies the structure-preserving, provability, and uniqueness properties. A VRF is required to satisfy the pseudorandomness property. We argue that constructing a structure-preserving VRF is impossible in the following sense.

Theorem 4. *If the discrete logarithm problem is hard in the source groups of Λ generated by $\mathcal{G}$, there is no structure-preserving verifiable random function whose* KeyGen, Comp, *and* Prove *algorithms are algebraic with respect to $\mathcal{G}$ and that is pseudorandom, as defined in Definition 4.*

To prove the theorem, it is sufficient to show that a structure-preserving VRF satisfying the pseudorandomness property is an unpredictable SPDP and apply Theorem 1.

Lemma 4. *A structure-preserving verifiable random function that is pseudorandom according to Definition 4 is an SPDP satisfying the unpredictability property, as defined in Definition 3.*

Proof. The syntactical equivalence is by definition. We focus on the part that pseudorandomness implies unpredictability. From any adversary $\mathcal{A}$ that wins the unpredictability game from Definition 3 with non-negligible probability, we construct a distinguisher $\mathcal{D} = (\mathcal{D}_1, \mathcal{D}_2)$ that wins the pseudorandomness game from Definition 4 with a probability non-negligibly larger than $1/2$. The $\mathcal{D}$ uses $\mathcal{A}$ that breaks the unpredictability. The $\mathcal{D}_1$ executes a copy of $\mathcal{A}$ internally and forwards the oracle queries and answers appropriately. If $\mathcal{A}$ produces an output pair (X, Y), where Y is an output value for a fresh input X that was not queried to the oracle before, then $\mathcal{D}_1$ uses X as its output and forwards Y to $\mathcal{D}_2$ that uses Y to distinguish if the returned challenge Y_b is a random value or the output of the real function. If no such pair (X, Y) is produced by $\mathcal{A}$, then $\mathcal{D}$ makes a random guess. From the construction, the advantage of $\mathcal{D}$ is the same as the success probability of $\mathcal{A}$, as claimed. □

We note that this result also rules out the construction of a structure-preserving simulatable VRF (sVRF) [26], which is a special case of a VRF (see Definition 1 from [26]) and is a key building block, for instance, of some e-cash schemes [7].

3.3. *Impossibility of Structure-Preserving Pseudorandom Functions*

We define structure-preserving PRFs and their security as follows. Note that keys are allowed to be scalar values and the pseudorandomness is defined in a weaker form. The standard pseudorandomness property requires that, given an unlimited number of oracle accesses to either a PRF or a truly random function, no polynomial-time adversary can distinguish which oracle it has accessed. The following weaker notion, in which only one call to the oracle that is either a PRF or a random function is provided, is implied by the standard one. The implication can be proved using a standard hybrid argument.

Definition 7. (*Structure-Preserving Pseudorandom Function*) A function family $F : \mathcal{SK} \times \mathcal{X} \rightarrow \mathcal{Y}$ is called a pseudorandom function if there are probabilistic polynomial-

time algorithms Setup and KeyGen and a deterministic polynomial-time algorithm Comp such that:

- $CP \xleftarrow{\$} \mathsf{Setup}_{\mathrm{prf}}(\Lambda)$ is an algorithm that takes as input a Λ and outputs the common parameters CP.
- $SK \xleftarrow{\$} \mathsf{KeyGen}_{\mathrm{prf}}(CP)$ is an algorithm that takes as input CP and outputs a (secret) key $SK \in \mathcal{SK}$.
- $Y \leftarrow \mathsf{Comp}_{\mathrm{prf}}(X, SK)$ is a deterministic algorithm that takes as input $X \in \mathcal{X}$ and $SK \in \mathcal{SK}$ and outputs the function value $Y = F_{SK}(X) \in \mathcal{Y}$.

It is structure-preserving if $\mathcal{X}$ and $\mathcal{Y}$ are subsets of $\{\mathbb{G}_1, \mathbb{G}_2\}^*$ and if the relation defined by $Y = \mathsf{Comp}_{\mathrm{prf}}(X, SK)$, where X and Y form the statement and SK is a witness, can be represented by PPEs (when SK is in $\{\mathbb{G}_1, \mathbb{G}_2\}^*$) or MSEs (when SK is in $\{\mathbb{Z}_p, \mathbb{G}_1, \mathbb{G}_2\}^*$).

Definition 8. (*Pseudorandomness of Pseudorandom Function*) For all probabilistic polynomial-time distinguishers $\mathcal{D} = (\mathcal{D}_1, \mathcal{D}_2)$, we have

$$\mathsf{Pr}\left[\begin{array}{l} CP \xleftarrow{\$} \mathsf{Setup}(\Lambda)\ ;\ SK \xleftarrow{\$} \mathsf{KeyGen}(CP)\ ; \\ (X, st) \leftarrow \mathcal{D}_1{}^{\mathsf{Comp}(\cdot, SK)}(CP)\ ; \\ Y_0 \leftarrow F_{SK}(X)\ ;\ Y_1 \xleftarrow{\$} \mathcal{Y}\ ;\ b \xleftarrow{\$} \{0,1\}\ ; \\ b' \xleftarrow{\$} \mathcal{D}_2{}^{\mathsf{Comp}(\cdot, SK)}(Y_b, st) \end{array} \middle| \; b = b' \wedge X \notin S \right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where S is the set of queries to the oracle Comp.

The purpose of using a structure-preserving PRF, for instance, is to show that one knows some X and SK that results in Y by coupling it with the Groth–Sahai proof system. We show that a structure-preserving PRF implies a structure-preserving VRF which, however, does not exist, as discussed in the previous section. More precisely, we argue that a combination of structure-preserving PRFs with the Groth–Sahai proof system results in a VRF in the CRS model. To see exactly what property of the GS proof system is used in our construction, we reformulate it as a structure-preserving commit-then-prove ZK proof system, as done by Escala and Groth [31], with some adjustments to our context.

Let $\mathcal{R}_{\mathcal{G}}$ be a relation represented by conjunction and disjunction of PPEs and MSEs with respect to the groups generated by $\mathcal{G}$. Constants in PPEs or MSEs are statements, and satisfying values assigned to the variables are witnesses. The language $\mathcal{L}_{\mathcal{G}}$ characterized by $\mathcal{R}_{\mathcal{G}}$ is a set of statements for which a witness exists.

Definition 9. (*Structure-Preserving Non-Interactive Commit-then-Prove Zero-Knowledge Proof System*)

A structure-preserving non-interactive commit-then-prove zero-knowledge proof system for relation $\mathcal{R}_{\mathcal{G}}$ consists of the following algorithms.

- $CP \xleftarrow{\$} \mathsf{Setup}(\Lambda)$: On input of a Λ, it outputs a common parameter CP for the proof system.

- $CRS \xleftarrow{\$} \mathsf{CrsGen}(CP)$: On input of a CP, it outputs a common reference string CRS.
- $(C, R) \xleftarrow{\$} \mathsf{Commit}(CRS, W)$: On input of a CRS and a witness W, it outputs a commitment C and the random coin R used to compute C.
- $P \xleftarrow{\$} \mathsf{Prove}(CRS, S, W, R)$: On input of a CRS, a statement S, a witness W, and a randomness R, it generates a zero-knowledge proof P in which W satisfies S (provided that this is the case).
- $0/1 \leftarrow \mathsf{Verify}(CRS, S, P, C)$: On input of a CRS, S, P, and a C, it outputs 1 or 0 representing acceptance or rejection of P, respectively.

It is required that CRS, P, and C are in $\{\mathbb{G}_1, \mathbb{G}_2\}^*$, and Verify is done only by group membership testing, group operations, and evaluating PPEs. It is perfectly sound if $1 \leftarrow \mathsf{Verify}(CRS, S, P, C)$, then there exists unique (W, R) that satisfies $(C, R) \leftarrow \mathsf{Commit}(CRS, W)$ and statement S. For any Λ produced by $\mathcal{G}$, $(\Lambda, S, W) \in R_{\mathcal{G}}$, $(C, R) \xleftarrow{\$} \mathsf{Commit}(CRS, W)$ and $P \xleftarrow{\$} \mathsf{Prove}(CRS, S, W, R)$, algorithm $\mathsf{Verify}(CRS, S, P, C)$ outputs 1. It is composable zero-knowledge if there exists polynomial-time algorithms $\mathsf{SimCrsGen}$, $\mathsf{SimCommit}$, and $\mathsf{SimProve}$ such that

- $(CRS, \tau) \xleftarrow{\$} \mathsf{SimCrsGen}(CP)$: On input of a CP, it outputs a CRS and a simulation key τ,
- $(C, R) \xleftarrow{\$} \mathsf{SimCommit}(CRS, \tau)$: On input of a CRS and τ, it outputs C and R used to compute C, and
- $P \xleftarrow{\$} \mathsf{SimProve}(CRS, S, \tau, R)$: On input of a CRS, S, τ, and R, it simulates proof P

holds, the distributions of a CRS as produced by CrsGen and $\mathsf{SimCrsGen}$ are indistinguishable, and the distributions of (P, C), with P output by Prove or $\mathsf{SimProve}$ and C by Commit or $\mathsf{SimCommit}$, are identical for a simulated CRS and $(\Lambda, S, W) \in \mathcal{R}_{\mathcal{G}}$ chosen by an adversary with access to τ.

Let $PRF = (\mathsf{Setup}_{\text{prf}}, \mathsf{KeyGen}_{\text{prf}}, \mathsf{Comp}_{\text{prf}})$ be a structure-preserving PRF with respect to $\mathcal{G}$ and $NIZK = (\mathsf{Setup}_{\text{nizk}}, \mathsf{CrsGen}_{\text{nizk}}, \mathsf{Commit}_{\text{nizk}}, \mathsf{Prove}_{\text{nizk}}, \mathsf{Verify}_{\text{nizk}})$ be a structure-preserving non-interactive commit-and-prove ZK proof system with respect to $\mathcal{G}$.

Theorem 5. *If the discrete logarithm problem is hard in the source groups of Λ generated by $\mathcal{G}$, and there exists a structure-preserving non-interactive commit-then-prove zero-knowledge proof system $NIZK = (\mathsf{Setup}_{\text{nizk}}, \mathsf{CrsGen}_{\text{nizk}}, \mathsf{Commit}_{\text{nizk}}, \mathsf{Prove}_{\text{nizk}}, \mathsf{Verify}_{\text{nizk}})$ with respect to $\mathcal{G}$ whose $\mathsf{CrsGen}_{\text{nizk}}$, $\mathsf{Commit}_{\text{nizk}}$, and $\mathsf{Prove}_{\text{nizk}}$ are algebraic algorithms, then there is no structure-preserving pseudorandom function $PRF = (\mathsf{Setup}_{\text{prf}}, \mathsf{KeyGen}_{\text{prf}}, \mathsf{Comp}_{\text{prf}})$ whose $\mathsf{KeyGen}_{\text{prf}}$ and $\mathsf{Comp}_{\text{prf}}$ are algebraic algorithms with respect to $\mathcal{G}$ and is pseudorandom as defined in Definition 8.*

Proof. Suppose that a PRF is a structure-preserving PRF whose $\mathsf{KeyGen}_{\text{prf}}$ and $\mathsf{Comp}_{\text{prf}}$ are algebraic algorithms with respect to $\mathcal{G}$ and is pseudorandom, as defined in Definition 8. Also suppose that the $NIZK$ is a structure-preserving non-interactive commit-then-

prove ZK proof system with respect to $\mathcal{G}$ whose $\mathsf{CrsGen}_{\mathrm{nizk}}$, $\mathsf{Commit}_{\mathrm{nizk}}$, and $\mathsf{Prove}_{\mathrm{nizk}}$ are algebraic algorithms. From such *PRF* and *NIZK*, we construct a structure-preserving VRF whose KeyGen and Comp are algebraic algorithms, and that is pseudorandom.

- $CP \xleftarrow{\$} \mathsf{Setup}(\Lambda)$: $CP_{\mathrm{prf}} \xleftarrow{\$} \mathsf{Setup}_{\mathrm{prf}}(\Lambda)$, $CP_{\mathrm{nizk}} \xleftarrow{\$} \mathsf{Setup}_{\mathrm{nizk}}(\Lambda)$, $CP := (CP_{\mathrm{prf}}, CP_{\mathrm{nizk}})$.
- $(PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(CP)$:$SK_{\mathrm{prf}} \xleftarrow{\$} \mathsf{KeyGen}_{\mathrm{prf}}(CP_{\mathrm{prf}})$, $CRS \xleftarrow{\$} \mathsf{CrsGen}_{\mathrm{nizk}}(CP_{\mathrm{nizk}})$, $(C, R) \xleftarrow{\$} \mathsf{Commit}_{\mathrm{nizk}}(CRS, SK_{\mathrm{prf}})$, $SK := (SK_{\mathrm{prf}}, R, CRS)$, $PK := (C, CRS)$. Return (PK, SK).
- $Y \leftarrow \mathsf{Comp}(X, SK)$: $(SK_{\mathrm{prf}}, R, CRS) \leftarrow SK$. $Y \leftarrow \mathsf{Comp}_{\mathrm{prf}}(X, SK_{\mathrm{prf}})$. Return Y.
- $P \xleftarrow{\$} \mathsf{Prove}(X, SK)$: Parse $(SK_{\mathrm{prf}}, R, CRS) \leftarrow SK$ and compute $Y \leftarrow \mathsf{Comp}_{\mathrm{prf}}(X, SK_{\mathrm{prf}})$. Then, output $P \xleftarrow{\$} \mathsf{Prove}_{\mathrm{nizk}}(CRS, (X, Y), SK_{\mathrm{prf}}, R)$.
- $0/1 \leftarrow \mathsf{Verify}(X, Y, PK, P)$: Parse $(C, CRS) \leftarrow PK$. Run $b \leftarrow \mathsf{Verify}_{\mathrm{nizk}}(CRS, (X, Y), P, C)$ and return b.

Syntactical consistency as a VRF can be verified by inspection. Its KeyGen, Comp, and Prove algorithms are algebraic with respect to $\mathcal{G}$ because $\mathsf{KeyGen}_{\mathrm{prf}}$, $\mathsf{Comp}_{\mathrm{prf}}$, $\mathsf{CrsGen}_{\mathrm{nizk}}$, $\mathsf{Commit}_{\mathrm{nizk}}$, and $\mathsf{Prove}_{\mathrm{nizk}}$ are all algebraic. It is structure-preserving since PK consists of C and CRS in $\{\mathbb{G}_1, \mathbb{G}_2\}^*$, X, Y, and P are also in $\{\mathbb{G}_1, \mathbb{G}_2\}^*$, and Verify only calls $\mathsf{Verify}_{\mathrm{nizk}}$ that meets the requirements.

Provability holds from the correctness of the *NIZK*, as Verify is identical to $\mathsf{Verify}_{\mathrm{nizk}}$. Uniqueness holds due to the soundness of the *NIZK* and the fact that $\mathsf{Comp}_{\mathrm{prf}}$ is deterministic. Namely, if (SK, X, Y) satisfies relation $Y = \mathsf{Comp}_{\mathrm{prf}}(X, SK)$, then $Y \neq Y'$ does not satisfy $Y' = \mathsf{Comp}_{\mathrm{prf}}(X, SK)$ since $\mathsf{Comp}_{\mathrm{prf}}$ is deterministic. Thus, by the soundness of the NIZK, there is no P' that is accepted by the verification algorithm for (SK, X, Y'). The pseudorandomness holds due to the pseudorandomness of the *PRF*, as in Definition 8, and the composable ZK property of *NIZK*.

Such a structure-preserving VRF contradicts Theorem 4. Since the GS proof system instantiates the above *NIZK* under standard assumptions,[1] we can conclude that a structure-preserving PRF cannot exist. □

3.4. *Impossibility Results for Structure-Preserving Deterministic Encryption*

Deterministic encryption was introduced by Bellare, Boldyreva, and O'Neill [10]. Here, we provide their definition of DE, adopted to the structure-preserving setting.

Definition 10. (*Structure-Preserving Deterministic Encryption*) A structure-preserving deterministic encryption scheme with respect to $\mathcal{G}$ consists of the following algorithms.

- $CP \xleftarrow{\$} \mathsf{Setup}(\Lambda)$ is an algorithm that takes as input a Λ and outputs common parameters CP which define $\mathcal{PK}$, $\mathcal{X}$, and $\mathcal{Y}$.

[1] If the correctness relation for $\mathsf{Comp}_{\mathrm{prf}}$ is given by MSEs, an accompanying GS-proof may contain scalar values [38]. However, they then can be turned into source group elements simply by lifting them up on the CRS as bases so that the output of Prove consists only of source group elements.

- $(PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(CP)$ is a probabilistic key generation algorithm that takes as input CP and outputs a public key $PK \in \mathcal{PK}$ and a secret key SK. PK is included in or uniquely computable from SK.
- $Y \leftarrow \mathsf{Enc}(X, PK)$ is a deterministic algorithm that takes as input a plaintext $X \in \mathcal{X}$ and a PK and outputs a ciphertext $Y \in \mathcal{Y}$.
- $X \leftarrow \mathsf{Dec}(Y, SK)$ is a deterministic algorithm that takes as input a $Y \in \mathcal{Y}$ and a SK and outputs a plaintext $X \in \mathcal{X}$.

It is required that $X = \mathsf{Dec}(\mathsf{Enc}(X, PK), SK)$ holds for all $X \in \mathcal{X}$ and all pairs (PK, SK) computed legitimately by $\Lambda \leftarrow \mathcal{G}(1^\lambda)$, $CP \leftarrow \mathsf{Setup}(\Lambda)$, $(PK, SK) \leftarrow \mathsf{KeyGen}(CP)$.

It is structure-preserving with respect to encryption if additionally $\mathcal{PK}$, $\mathcal{X}$, and $\mathcal{Y} \subset \{\mathbb{G}_1, \mathbb{G}_2\}^*$, and the relation $Y \leftarrow \mathsf{Enc}(X, PK)$, where Y and PK form the statement and X is a witness, can be represented by PPEs. Similarly, it is structure-preserving with respect to decryption if additionally the relation $X \leftarrow \mathsf{Dec}(Y, SK)$, where X and Y form the statement and SK is a witness, can be represented by PPEs or MSEs.

Intuitively, the security notion for DE states that the adversary should not be able to distinguish ciphertexts that correspond to messages that come from two message distributions with high min-entropy conditioned on previous messages. Following [12], we define IND security of structure-preserving deterministic encryption schemes as follows. We call a list $\mathcal{X} = (X_1, \ldots, X_n)$ of random variables over $\{\mathbb{G}_1, \mathbb{G}_2\}$ a block source (parameterized by some n and t) if $H_\infty(X_i | X_1, \ldots, X_{i-1}) \geq t$ for all $i \in \{1, \ldots, n\}$.

Definition 11. (*IND Security for Deterministic Encryption*) A deterministic encryption scheme $\Sigma_{\mathsf{DE}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is PRIV-secure for block sources if, for any block source $\mathcal{X}$ of polynomial length n, any function $f : \{\mathbb{G}_1, \mathbb{G}_2\}^n \to \{0,1\}^*$ and all poly-time adversaries A, the advantage $Real(A, f, \mathcal{X}) - Ideal(A, f, \mathcal{X})$ is negligible, where

$$Real(A, f, \mathcal{X}) = \mathsf{Pr}\left[\begin{array}{l} CP \xleftarrow{\$} \mathsf{Setup}(\Lambda)\,; \\ (PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(CP)\,; \\ X \xleftarrow{\$} \mathcal{X} \end{array} \middle| \; A(PK, \mathsf{Enc}(PK, X)) = f(X) \right]$$

and

$$Ideal(A, f, \mathcal{X}) = \mathsf{Pr}\left[\begin{array}{l} CP \xleftarrow{\$} \mathsf{Setup}(\Lambda)\,; \\ (PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(CP)\,; \\ X, X' \xleftarrow{\$} \mathcal{X} \end{array} \middle| \; A(PK, \mathsf{Enc}(PK, X')) = f(X) \right]$$

Theorem 6. *Assuming the hardness of the discrete logarithm problem in the base groups of $\Lambda \in \mathcal{G}$, there is no IND-secure deterministic encryption scheme that is structure-preserving with respect to encryption.*

Note that being structure-preserving with respect to Enc implies that Enc is algebraic. (It is generic, in fact.) To prove Theorem 6, we actually prove Lemma 5 that states a slightly stronger result that there is no structure-preserving DE with respect to encryption protecting any kind of secrecy of plaintexts.

Lemma 5. *Assuming the hardness of the discrete logarithm problem in the base groups of $\Lambda \in \mathcal{G}$, any structure-preserving deterministic encryption scheme with respect to encryption, whose* KeyGen *and* Enc *are algebraic with respect to $\mathcal{G}$, allows decryption only with the public key.*

Proof. Let $\Sigma_{\mathsf{DE}} = (\mathsf{Setup}_{\mathsf{de}}, \mathsf{KeyGen}_{\mathsf{de}}, \mathsf{Enc}_{\mathsf{de}}, \mathsf{Dec}_{\mathsf{de}})$ be a structure-preserving DE scheme whose KeyGen and Enc are algebraic with respect to $\mathcal{G}$. We construct a provable structure-preserving deterministic primitive Σ_{SPDP} from Σ_{DE} as follows.

- $CP \xleftarrow{\$} \mathsf{Setup}(\Lambda)$: $CP \xleftarrow{\$} \mathsf{Setup}_{\mathsf{de}}(\Lambda)$.
- $(PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(CP)$: $(SK, PK) \xleftarrow{\$} \mathsf{KeyGen}_{\mathsf{de}}(CP)$.
- $Y \leftarrow \mathsf{Comp}(X, SK)$: Derive PK from SK, and run $Y \leftarrow \mathsf{Enc}_{\mathsf{de}}(X, PK)$. Return Y.
- $P \xleftarrow{\$} \mathsf{Prove}(X, SK)$: Return a constant in $\{\mathbb{G}_1, \mathbb{G}_2\}^*$ as P.
- $0/1 \leftarrow \mathsf{Verify}(X, Y, P, PK)$: Return 1 if $Y = \mathsf{Enc}_{\mathsf{de}}(X, PK)$. Return 0, otherwise.

We verify that the above constitute an Σ_{SPDP} according to Definition 2. Syntactical consistency can be verified by inspection. We focus on the security properties. First, it is structure-preserving since PK, X, Y, P are in $\{\mathbb{G}_1, \mathbb{G}_2\}^*$ and Verify only evaluates Enc of Σ_{DE} that meets the requirement. Provability and uniqueness hold trivially from the fact that Enc is deterministic.

We now note that the above SPDP is not unpredictable because Comp essentially uses PK instead of SK. Nevertheless, according to Lemma 1 (the conditions required in the lemma are satisfied here as we assume the hardness of the discrete logarithm problem in Λ and $\mathsf{KeyGen}_{\mathsf{de}}$ and $\mathsf{Enc}_{\mathsf{de}}$ are algebraic), the ciphertext that encrypts group elements $X = (X_1, \ldots, X_n)$ for some n looks as follows: $\mathsf{Comp}(X, SK) = \mathsf{Enc}_{\mathsf{de}}(X, PK) = Y = (G^{a_1} \prod_{j=1}^{n} {X_j}^{b_{1,j}}, \ldots, G^{a_\ell} \prod_{j=1}^{n} {X_j}^{b_{\ell,j}})$, where $a_1, \ldots, a_\ell, b_{1,1}, \ldots, b_{\ell,n}$ are constants in $\mathbb{Z}_p$, and G is a group generator. Since Enc is structure-preserving encryption, G^{a_i} and $b_{i,j}, i = 1, \ldots, \ell, j = 1, \ldots, n$ should depend only on PK and be efficiently computable from PK. Hence, the plaintext X can be recovered solely by using PK. □

We note that SPDE with respect to decryption exists in the random oracle model. The following is a modification of the encrypt-then-hash deterministic encryption scheme of Bellare et al. [10]. Consider a deterministic variant of ElGamal encryption scheme whose secret key is $x \in \mathbb{Z}_p$ and public key is $Y = G_1^x$. It encrypts message $M \in \mathbb{G}_1$ into $(C_1, C_2) = (M \cdot Y^{H(Y\|M)}, G_1^{H(Y\|M)})$ and decrypts by C_1/C_2^x without checking the well formness of the ciphertext. The scheme is IND-secure in the random oracle model under the SXDH assumption (but obviously not secure against chosen ciphertext attacks) and structure-preserving with respect to decryption since the relation determined by the decryption can be verified by two MSEs $C_1/C_2^x = M$ and $Y = G^x$ where (x, M) is the

witness. By coupling it with the GS proof system, one can prove one's knowledge of the correct decryption of the ciphertext. It would, however, be of limited use since the result of decryption must not be revealed. Nevertheless, it is a rare example that uses a hash function but remains structure-preserving.

4. Conclusion

We proved that it is impossible to construct algebraic structure-preserving VRFs, VUFs, USigs, PRFs, and DE schemes. We further extend our results to "non-strictly" structure-preserving primitives, which are allowed to have target group elements in their public keys and ranges. Although our results are restricted to the class of algebraic algorithms, all known constructions of structure-preserving primitives consist of algebraic algorithms. Finding constructions of secure structure-preserving algorithms that allow non-algebraic operations but whose correctness of computation still can be verified using a system of PPEs is an interesting open problem. Finally, we note that deterministic primitives might exist in a restricted form, where only one query to the oracle is allowed.

Acknowledgements

The authors would like to thank Kristiyan Haralambiev for the useful discussions and the anonymous reviewers for their helpful comments and suggestions. The research leading to these results was supported in part by the European Community's Seventh Framework Programme for the projects ABC4Trust (Grant Agreement No. 257782) and PERCY (Grant Agreement No. 321310).

References

[1] M. Abdalla, D. Catalano, D. Fiore, Verifiable random functions from identity-based key encapsulation, in A. Joux, editor, *Advances in Cryptology—EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26–30, 2009. Proceedings*. LNCS, vol. 5479 (Springer, Berlin, 2009), pp. 554–571

[2] M. Abe, M. Chase, B. David, M. Kohlweiss, R. Nishimaki, M. Ohkubo, Constant-size structure-preserving signatures: generic constructions and simple assumptions. *J. Cryptol.* **29**(4):833–878 (2016)

[3] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, M. Ohkubo, Structure-preserving signatures and commitments to group elements. *J. Cryptol.* **29**(2):363–421 (2016)

[4] M. Abe, J. Groth, K. Haralambiev, M. Ohkubo, Optimal structure-preserving signatures in asymmetric bilinear groups, in *Advances in Cryptology—CRYPTO'11*. LNCS. (Springer, Berlin, 2011)

[5] M. Abe, K. Haralambiev, M. Ohkubo, Group to group commitments do not shrink, in D. Pointcheval, T. Johansson, editors, *Advances in Cryptology—EUROCRYPT 2012—31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15–19, 2012. Proceedings*. LNCS, vol. 7237 (Springer, Berlin 2012), pp. 301–317

[6] M. Belenkiy, M. Chase, M. Kohlweiss, A. Lysyanskaya, Non-interactive anonymous credentials, in R. Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008*. LNCS, vol. 4948 (Springer, Berlin, 2008). Also available on IACR ePrint Archive, 2007/384

[7] M. Belenkiy, M. Chase, M. Kohlweiss, A. Lysyanskaya, Compact e-cash and simulatable VRFs revisited, in H. Shacham, B. Waters, editors, *Pairing-Based Cryptography—Pairing 2009, Third International Conference, Palo Alto, CA, USA, August 12–14, 2009, Proceedings*. LNCS, vol. 5671 (Springer, Berlin 2009), pp. 114–131.

[8] M. Bellare, D. Micciancio, B. Warinschi, Foundations of group signatures: formal definitions, simplified requirements and a construction based on general assumptions, in E. Biham, editor, *Advances in Cryptology—EUROCRPYT'03*. LNCS, vol. 2656 (2003), pp. 614–629

[9] M. Bellare, A. Palacio, The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols, in M. Franklin, editor, *Advances in Cryptology—CRYPTO 2004*. LNCS, vol. 3152 (Springer, Berlin 2004), pp. 273–289

[10] M. Bellare, A. Boldyreva, A. O'Neill, Deterministic and efficiently searchable encryption, in A. Menezes, editor, *Advances in Cryptology—CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2007, Proceedings*. LNCS, vol. 4622 (Springer, Berlin, 2007), pp. 535–552

[11] M. Bellare, M. Fischlin, A. O'Neill, T. Ristenpart, Deterministic encryption: definitional equivalences and constructions without random oracles, in D. Wagner, editor, *Advances in Cryptology—CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17–21, 2008. Proceedings*. LNCS, vol. 5157 (Springer, Berlin, 2008), pp. 360–378

[12] A. Boldyreva, S. Fehr, A. O'Neill, On notions of security for deterministic encryption, and efficient constructions without random oracles, in D. Wagner, editor, *Advances in Cryptology—CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17–21, 2008. Proceedings*. LNCS, vol. 5157 (Springer, Berlin, 2008), pp. 335–359

[13] J. Camenisch, M. Dubovitskaya, K. Haralambiev, Efficient structure-preserving signature scheme from standard assumptions, in *SCN*. LNCS, vol. 7485 (Springer, Berlin, 2012), pp. 76–94

[14] J. Camenisch, M. Dubovitskaya, G. Neven, Oblivious transfer with access control, in E. Al-Shaer, S. Jha, A.D. Keromytis, editors, *Proceedings of the 2009 ACM Conference on Computer and Communications Security, CCS 2009, Chicago, Illinois, USA, November 9–13, 2009* (ACM, 2009), pp. 131–140

[15] J. Camenisch, M. Dubovitskaya, G. Neven, G.M. Zaverucha, Oblivious transfer with hidden access control policies, in D. Catalano, N. Fazio, R. Gennaro, A. Nicolosi, editors, *Public Key Cryptography—PKC 2011—14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6–9, 2011. Proceedings*. LNCS, vol. 6571 (Springer, Berlin, 2011), pp. 192–209

[16] J. Camenisch, K. Haralambiev, M. Kohlweiss, J. Lapon, V. Naessens, Structure preserving CCA secure encryption and applications, in *Advances in Cryptology—Asiacrypt 2011*. LNCS (Springer, Berlin, 2011)

[17] J. Camenisch, S. Hohenberger, A. Lysyanskaya, Compact e-cash, in R. Cramer, editor, *Advances in Cryptology—EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22–26, 2005, Proceedings*. LNCS, vol. 3494 (Springer, Berlin 2005), pp. 302–321

[18] J. Camenisch, A. Kiayias, M. Yung, On the portability of generalized Schnorr proofs, in A. Joux, editor, *Advances in Cryptology—EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26–30, 2009. Proceedings*. LNCS, vol. 5479 (Springer, Berlin, 2009), pp. 425–442

[19] J. Camenisch, A. Lysyanskaya, An efficient system for non-transferable anonymous credentials with optional anonymity revocation, in B. Pfitzmann, editor, *Advances in Cryptology—EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6–10, 2001, Proceeding*. LNCS, vol. 2045 (Springer, Berlin, 2001), pp. 93–118

[20] J. Camenisch, A. Lysyanskaya, A signature scheme with efficient protocols, in S. Cimato, C. Galdi, G. Persiano, editors, *Security in Communication Networks, Third International Conference, SCN 2002, Amalfi, Italy, September 11–13, 2002. Revised Papers*. LNCS, vol. 2576 (Springer, Berlin, 2002), pp. 268–289

[21] J. Camenisch, A. Lysyanskaya, Signature schemes and anonymous credentials from bilinear maps, in M.K. Franklin, editor, *Advances in Cryptology—CRYPTO 2004, 24th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15–19, 2004, Proceedings*. LNCS, vol. 3152 (Springer, Berlin, 2004), pp. 56–72

[22] J. Camenisch, G. Neven, A. Shelat, Simulatable adaptive oblivious transfer, in M. Naor, editor, *Advances in Cryptology—EUROCRYPT 2007, 26th Annual International Conference on the Theory and Applica-*

tions of Cryptographic Techniques, Barcelona, Spain, May 20–24, 2007, Proceedings. LNCS, vol. 4515 (Springer, Berlin, 2007), pp. 573–590

[23] J. Camenisch, V. Shoup, Practical verifiable encryption and decryption of discrete logarithms, in D. Boneh, editor, *Advances in Cryptology—CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17–21, 2003, Proceedings*. LNCS, vol. 2729 (Springer, Berlin, 2003), pp. 126–144

[24] R. Canetti, Universally composable security: a new paradigm for cryptographic protocols, in *Proceedings of the 42nd IEEE Annual Symposium on Foundations of Computer Science* (2001), pp. 136–145

[25] R. Canetti, O. Goldreich, S. Halevi, The random oracle methodology, revisited, in *Proceedings of the 30th Annual ACM Symposium on Theory of Computing* (1998), pp. 209–218

[26] M. Chase, A. Lysyanskaya, Simulatable VRFs with applications to multi-theorem NIZK, in A. Menezes, editor, *Advances in Cryptology—CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2007, Proceedings*. LNCS, vol. 4622 (Springer, Berlin, 2007), pp. 303–322

[27] R. Cramer, V. Shoup, Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Comput.* **33**(1):167–226 (2003)

[28] Y. Dodis, Efficient construction of (distributed) verifiable random functions, in Y. Desmedt, editor, *Public Key Cryptography—PKC 2003, 6th International Workshop on Theory and Practice in Public Key Cryptography, Miami, FL, USA, January 6–8, 2003, Proceedings*. LNCS, vol. 2567 (Springer, Berlin, 2003), pp. 1–17

[29] Y. Dodis, A. Yampolskiym, A verifiable random function with short proofs and keys, in S. Vaudenay, editor, *Public Key Cryptography—PKC 2005, 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets, Switzerland, January 23–26, 2005, Proceedings*. LNCS, vol. 3386 (Springer, Berlin, 2005), pp. 416–431

[30] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, in G.R. Blakley, D. Chaum, editors, *Advances in Cryptology—CRYPTO'84*. LNCS, vol. 196 (Springer, Berlin, 1985), pp. 10–18

[31] A. Escala, J. Groth, Fine-tuning Groth–Sahai proofs, in H. Krawczyk, editor, *Public-Key Cryptography—PKC 2014—17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26–28, 2014. Proceedings*. LNCS, vol. 8383 (Springer, Berlin, 2014), pp. 630–649

[32] A. Fiat, A. Shamir, How to prove yourself: practical solutions to identification and signature problems, in A.M. Odlyzko, editor, *Advances in Cryptology—CRYPTO'86*. LNCS, vol. 263 (Springer, Berlinm 1987), pp. 186–199

[33] M.J. Freedman, Y. Ishai, B. Pinkas, O. Reingold, Keyword search and oblivious pseudorandom functions, in J. Kilian, editor, *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10–12, 2005, Proceedings*. LNCS, vol. 3378 (Springer, Berlin, 2005), pp. 303–324

[34] S.D. Galbraith, K.G. Peterson, N.P. Smart, Pairings for cryptographers. *Discrete Appl. Math.* **156**(16):3113–3121 (2008)

[35] O. Goldreich, S. Goldwasser, S. Micali, How to construct random functions. *J. ACM* **33**(4):792–807 (1986)

[36] S. Goldwasser, Y.T. Kalai, On the (in)security of the Fiat–Shamir paradigm, in *44th Symposium on Foundations of Computer Science (FOCS 2003), 11–14 October 2003, Cambridge, MA, USA, Proceedings* (IEEE Computer Society, 2003), pp. 102–113

[37] S. Goldwasser, R. Ostrovsky, Invariant signatures and non-interactive zero-knowledge proofs are equivalent (extended abstract), in E.F. Brickell, editor, *Advances in Cryptology—CRYPTO'92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16–20, 1992, Proceedings*. LNCS, vol. 740 (Springer, Berlin, 1992), pp. 228–245

[38] J. Groth, A. Sahai, Efficient noninteractive proof systems for bilinear groups. *SIAM J. Comput.* **41**(5):1193–1232 (2012)

[39] C. Hazay, Y. Lindell, Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries, in R. Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19–21, 2008*. LNCS, vol. 4948 (Springer, Berlin, 2008), pp. 155–175

[40] D. Hofheinz, T. Jager, Tightly secure signatures and public-key encryption, in *CRYPTO*. LNCS, vol. 7417 (Springer, Berlin, 2012), pp. 590–607

[41] S. Hohenberger, B. Waters, Constructing verifiable random functions with large input spaces, in H. Gilbert, editor, *Advances in Cryptology—EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30–June 3, 2010. Proceedings*. LNCS, vol. 6110 (Springer, Berlin, 2010), pp. 656–672

[42] S. Jarecki, X. Liu, Efficient oblivious pseudorandom function with applications to adaptive OT and secure computation of set intersection, in O. Reingold, editor, *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15–17, 2009. Proceedings*. LNCS, vol. 5444 (Springer, Berlin, 2009), pp. 577–594

[43] S. Jarecki, V. Shmatikov, Handcuffing big brother: an abuse-resilient transaction escrow scheme, in C. Cachin, J. Camenisch, editors, *Advances in Cryptology—EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2–6, 2004, Proceedings*. LNCS, vol. 3027 (Springer, Berlin, 2004), pp. 590–608

[44] A. Kiayias, M. Yung, Group signatures with efficient concurrent join, in R. Cramer, editor, *Advances in Cryptology—EUROCRYPT 2005*. LNCS, vol. 3494 (Springer, Berlin, 2005), pp. 198–214

[45] M. Liskov, Updatable zero-knowledge databases, in B.K. Roy, editor, *Advances in Cryptology—ASIACRYPT 2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, December 4–8, 2005, Proceedings*. LNCS, vol. 3788 (Springer, Berlin, 2005), pp. 174–198

[46] A. Lysyanskaya, Unique signatures and verifiable random functions from the DH-DDH separation, in M. Yung, editor, *Advances in Cryptology—CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18–22, 2002, Proceedings*. LNCS, vol. 2442 (Springer, Berlin, 2002), pp. 597–612

[47] S. Micali, M.O. Rabin, S.P. Vadhan, Verifiable random functions, in *40th Annual Symposium on Foundations of Computer Science, FOCS'99, 17–18 October, 1999, New York, NY, USA* (IEEE Computer Society, 1999), pp. 120–130

[48] S. Micali, L. Reyzin, Soundness in the public-key model, in J. Kilian, editor, *Advances in Cryptology—CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19–23, 2001, Proceedings*. LNCS, vol. 2139 (Springer, Berlin 2001), pp. 542–565

[49] S. Micali, R.L. Rivest, Micropayments revisited, in B. Preneel, editor, *Topics in Cryptology—CT-RSA 2002, The Cryptographer's Track at the RSA Conference, 2002, San Jose, CA, USA, February 18–22, 2002, Proceedings*. LNCS, vol. 2271 (Springer, Berlin, 2002), pp. 149–163

[50] T.P. Pedersen, Non-interactive and information-theoretic secure verifiable secret sharing, in J. Feigenbaum, editor, *Advances in Cryptology—CRYPTO'91*. LNCS, vol. 576 (Springer, Berlin 1992), pp. 129–140

[51] C.P. Schnorr, Efficient signature generation for smart cards. *J. Cryptol.* **4**(3):239–252 (1991).