An Algebraic Framework for Pseudorandom Functions and Applications to Related-Key Security

Michel Abdalla(✉), Fabrice Benhamouda(✉), and Alain Passelègue(✉)

ENS, CNRS, INRIA, and PSL, École normale supérieure, 45 Rue d'Ulm, 75230 Paris Cedex 05, France
{michel.abdalla,fabrice.ben.hamouda,alain.passelegue}@ens.fr

Abstract. In this work, we provide a new algebraic framework for pseudorandom functions which encompasses many of the existing algebraic constructions, including the ones by Naor and Reingold (FOCS'97), by Lewko and Waters (CCS'09), and by Boneh, Montgomery, and Raghunathan (CCS'10), as well as the related-key-secure pseudorandom functions by Bellare and Cash (Crypto'10) and by Abdalla *et al.* (Crypto'14). To achieve this goal, we introduce two versions of our framework. The first, termed linearly independent polynomial security, states that the values $(g^{P_1(\vec{a})}, \ldots, g^{P_q(\vec{a})})$ are indistinguishable from a random tuple of the same size, when $P_1, \ldots, P_q$ are linearly independent multivariate polynomials of the secret key vector $\vec{a}$. The second, which is a natural generalization of the first framework, additionally deals with constructions based on the decision linear and matrix Diffie-Hellman assumptions. In addition to unifying and simplifying proofs for existing schemes, our framework also yields new results, such as related-key security with respect to arbitrary permutations of polynomials. Our constructions are in the standard model and do not require the existence of multilinear maps.

1 Introduction

Pseudorandom functions (PRFs), originally defined by Goldreich, Goldwasser, and Micali [19], are one of the most fundamental primitives in cryptography. Informally speaking, a function is said to be pseudorandom if its outputs are indistinguishable from that of a random function with respect to a computationally bounded adversary which only has black-box access to it. Hence, even if the adversary can control the inputs on which the function is computed and see the corresponding outputs, he or she should still not be able to distinguish this function from a perfectly random one.

Due to their simplicity and security properties, pseudorandom functions have been used in numerous applications, including symmetric encryption, authentication, and key exchange. In particular, since pseudorandom functions can be used to model real-world block-ciphers, such as AES [3], they are also extremely useful for the security analysis of protocols that rely on these primitives.

R. Gennaro and M. Robshaw (Eds.): CRYPTO 2015, Part I, LNCS 9215, pp. 388–409, 2015.
DOI: 10.1007/978-3-662-47989-6_19

Number-Theoretic Constructions. Despite its elegance, the original construction of pseudorandom functions by Goldreich, Goldwasser, and Micali based on pseudorandom generators was not very efficient. In order to improve its efficiency while still being able to prove its security under reasonable complexity assumptions, Naor and Reingold [27] proposed a new construction based on the Decisional Diffie-Hellman assumption (DDH) [27]. Let $\vec{a} = (a_0, \ldots, a_n) \in \mathbb{Z}_p^{n+1}$ be the key and $x = x_1 \| \ldots \| x_n \in \{0,1\}^n$ be the input of the PRF. Let g be a fixed public generator of a group $\mathbb{G}$ of prime order p. The Naor-Reingold PRF is then defined as

$$\mathsf{NR}(\vec{a}, x) = \left[a_0 \prod_{i=1}^{n} a_i^{x_i} \right]$$

where for any $a \in \mathbb{Z}_p$, $[a]$ stands for g^a, as defined in [18].

As mentioned in [17], the algebraic nature of the Naor-Reingold PRF has led to many applications, such as verifiable random functions [2,22], distributed PRFs [27], and related-key-secure PRFs [8], which are hard to obtain from generic PRFs. Hence, due to its importance, several other extensions of the Naor-Reingold PRF have been proposed [17,26] based on different assumptions, such as the Decision Linear assumption (DLin) [15] and the d-DDHI assumption [17,20].

In this work, our main contribution is to further extend the above line of work by providing a *generic algebraic framework* for building pseudorandom functions. In particular, all of the algebraic constructions mentioned above can be seen as a particular instantiations of our framework. In addition, our framework is general enough that it captures and extends other constructions such as the related-key-secure PRF constructions by Bellare and Cash [8] (BC) and by Abdalla *et al.* [1] (ABPP).

Linearly Independent Polynomial Security. To obtain our results, our first contribution is to introduce a new notion of linearly independent polynomial (LIP) security. Informally, it states that the values $([P_1(\vec{a})], \ldots, [P_q(\vec{a})])$ are indistinguishable from a random tuple of the same size, when $P_1, \ldots, P_q$ are linearly independent multivariate polynomials of degree at most d in any indeterminate and $\vec{a}$ is the PRF secret key vector. The new notion is based on a new MDDH assumption [18] over the underlying group $\mathbb{G}$, denoted $\mathcal{E}_{1,d}$-MDDH, which can be (tightly) reduced to either DDH or DDHI depending on value of d.

In order to illustrate the usefulness of the new notion, we show in Sect. 4 how to use it to provide alternative security proofs for the Naor-Reingold PRF [27] and the PRF by Boneh, Montgomery, and Raghunathan (BMR) in [17] as well as generalizations of both these PRFs, that we call *weighted* NR and *weighted* BMR. Intuitively, all these PRFs are defined over a prime order group $\mathbb{G} = \langle g \rangle$ as a function F that takes a key $\vec{a}$ and an input x and outputs an element in $\mathbb{G}$ of the shape $F(\vec{a}, x) = [P_x(\vec{a})]$ where the polynomial P_x depends on x. Hence, to prove the security of such constructions, we just need to prove that all polynomials P_x, for any entries x, are linearly independent.

We would like to remark that the actual formulation of the LIP security in Sect. 3 includes a value $a' \in \mathbb{Z}_p$ multiplying each $P_i(\vec{a})$ term, which allows for the use of different generators in the PRF constructions. While we could dispense with a' in the case where a' and the a_i values in $\vec{a}$ are scalars, we opted to use it to be consistent with the case in which these values are matrices, as in Sect. 6.

Applications to Related-Key Security. Related-key attacks (RKAs) were first introduced by Biham and Knudsen [11,24] and consider the setting in which an adversary could force a given cryptographic primitive to execute under a different but related key. Over the years, such attacks became more predominant and several related-key attacks have been proposed against existing block-ciphers (e.g., [12,13,23]). Since these attacks are quite powerful and hard to defend against, Bellare and Kohno [9] introduced a formal treatment of these attacks in the context of PRFs and pseudorandom permutations (PRPs) to better understand if and how one could achieve security in the presence of related-key attacks. One of their main observations is that certain classes of related-key attacks are impossible to protect against and, hence, their goal was to identify the set of classes Φ for which one could design secure RKA-PRFs and RKA-PRPs.

Let $F\colon \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ be a family of functions for a security parameter κ, and let $\Phi = \{\phi\colon \mathcal{K} \to \mathcal{K}\}$ be a set of related-key deriving (RKD) functions on the key space $\mathcal{K}$. Let $G\colon \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ be a random function and let $K \in \mathcal{K}$ be a random target key. Informally, in the RKA security model of [9], F is said to be a Φ-RKA-PRF if no polynomial-time adversary can distinguish the output of $F(\phi(K), x)$ from the output of $G(\phi(K), x)$, for pairs (ϕ, x) of its choice, with non-negligible probability.

Our second contribution is to show that the new LIP security notion can be used to prove directly the related-key security of certain constructions. In particular, we show that a particular case of our weighted BMR PRF construction is secure against permutations of the secret key. In these attacks, the attacker can obtain the output of the PRF with respect to any key that is a permutation of the original one.

To understand why RKA security can follow from the LIP security notion, let F be a PRF defined over a prime-order group $\mathbb{G} = \langle g \rangle$ that takes a key $\vec{a}$ and an input x and outputs $F(\vec{a}, x) = [P_x(\vec{a})]$. Let Φ be a class of RKD functions, where functions $\vec{\phi} = (\phi_1, \ldots, \phi_n) \in \Phi$ are such that ϕ_i are multivariate polynomials in $\mathbb{Z}_p[T_1, \ldots, T_n]$. Then, for a RKD function $\vec{\phi}$ and an input x, the PRF outputs $F(\vec{\phi}(\vec{a}), x) = \left[P_{\vec{\phi},x}(\vec{a})\right]$, where the polynomial $P_{\vec{\phi},x}(\vec{T}) = P_x(\vec{\phi}(\vec{T})) = P_x(\phi_1(\vec{T}), \ldots, \phi_n(\vec{T}))$ depends on $\vec{\phi}$ and x, with $\vec{T} = (T_1, \ldots, T_n)$. Hence, when all polynomials $P_{\vec{\phi},x}$ are linearly independent, the LIP security notion directly shows that F is Φ-RKA-secure.

Related-Key Security with Respect to Unique-Input Adversaries. Unfortunately, the case in which the polynomials $P_{\vec{\phi},x}$ are all linearly independent is not so easy to instantiate as we would like, and we have only been able to directly obtain RKA security for very restricted classes. Hence, to overcome these restrictions, our third contribution is to further extend our results in Sect. 5.2

to deal with the case where polynomials are only linearly independent when all the inputs x are distinct. This scenario is similar to the one considered in [1]. In particular, our new algebraic framework extends the one from [1] and provides constructions for new and larger classes of RKD functions. More precisely, we build in Sect. 5.2 RKA-PRFs against classes of permutations of univariate polynomials. Furthermore, in the full version, we also consider classes of univariate polynomials and multivariate affine RKD functions.

For simplicity, the results in Sect. 5.2 only hold with respect to PRFs of the form $[P_x(\vec{a})]$ where P_x is a polynomial that depends on x. However, a more general framework which does not make this assumption is described in the full version.

An Algebraic Framework for Non-commutative Structures. Finally, our last contribution is to extend the LIP security notion to work under weaker assumptions than DDH, such as DLin. As we point out in Sect. 6, the main difficulty in this case is that the key values a_i's may be matrices, which do not necessarily commute. To address this issue, we introduce natural conditions on the order of indeterminates which makes non-commutative and commutative polynomials behave in a similar manner. Through the new generalization, we not only deal with cases already covered by the LIP security notion, but we also capture PRFs based on the DLin and MDDH assumptions [18].

Further Discussions. In addition to the foundational work of Goldreich, Goldwasser, and Micali [19], several other frameworks for constructing PRFs have appeared in the literature, including [7,17,28] to name a few.

In [28], Naor and Reingold proposed the notion of pseudorandom synthesizers and provided several instantiations for it based on different complexity assumptions. Informally speaking, a pseudorandom synthesizer is a two-variable function, $S(\cdot,\cdot)$, so that, for polynomially many random and independent input assignments $(x_1,\ldots,x_m)$ and $(y_1,\ldots,y_m)$, the set of values $\{S(x_i,y_j)\}$ are computationally indistinguishable from uniform for i and j in $\{1,\ldots,m\}$.

In [7], Bellare, Canetti, and Krawczyk provide a framework for building variable-length input PRFs from fixed-length input ones, known as the cascade construction. In their framework, one obtains a larger-domain PRF F' simply by partitioning the input x into a number n of small blocks $x_1,\ldots,x_n$ matching the domain of the underlying PRF F and using the output of F on key k_i and input x_i as the secret key k_{i+1} for the next stage. Since their framework requires the output of the underlying PRF to be at least as long as the secret key, it cannot be applied to PRFs with very small domains.

To circumvent the restrictions of the cascade construction, Boneh, Montgomery, and Raghunathan proposed an extension in [17], known as the augmented cascade construction, in which supplemental secret information is provided in every iteration. Unlike the cascade construction, its security does not follow from the standard security of the underlying PRF, requiring it to meet a new notion called parallel security.

While these frameworks are more general than ours and capable of handling different complexity assumptions (e.g., [6]), they are more combinatorial

in nature and do not fully exploit the algebraic nature of the underlying PRFs. In particular, it is not clear how to extend them to the RKA setting, which is one of the main applications of our new algebraic framework. Moreover, even in the standard PRF setting, our framework seems to possess complementary features compared to the existing ones. Notably, it only requires the verification of an algebraic condition (such as testing the linear independence of the polynomials) for each instantiation, which is generally easier to prove.

Other Related Work. It is worth mentioning that in the context of related-key security, Lewi, Montgomery and Raghunathan [25] designed RKA-PRFs for similar classes of polynomial RKD functions. However, unlike their constructions, ours do not require multilinear maps. Also, our constructions are proven fully RKA-secure while theirs are only proven unique-input RKA-secure.

2 Definitions

Notations and Conventions. We denote by κ the security parameter. Let $F\colon \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ be a function that takes a key $K \in \mathcal{K}$ and an input $x \in \mathcal{D}$ and returns an output $F(K, x) \in \mathcal{R}$. The set of all functions $F\colon \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ is then denoted by $\mathsf{Fun}(\mathcal{K}, \mathcal{D}, \mathcal{R})$. Likewise, $\mathsf{Fun}(\mathcal{D}, \mathcal{R})$ denotes the set of all functions mapping $\mathcal{D}$ to $\mathcal{R}$. If S is a set, then $|S|$ denotes its size. We denote by $s \xleftarrow{\$} S$ the operation of picking at random s in S. If $\vec{x}$ is a vector then we denote by $|\vec{x}|$ its length, so $\vec{x} = (x_1, \ldots, x_{|\vec{x}|})$. For a binary string x, we denote its length by $|x|$ so $x \in \{0,1\}^{|x|}$, x_i its i-th bit, so $x = x_1 \,\|\, \ldots \,\|\, x_n$. We extend these notations to any d-ary string x, for $d \geq 2$. For a matrix $\boldsymbol{A}$ of size $k \times m$, we denote by $a_{i,j}$ the coefficient of $\boldsymbol{A}$ in the i-th row and the j-th column. For a vector $\vec{\phi} = (\phi_1, \ldots, \phi_n)$ of n functions from S_1 to S_2 with $|\vec{\phi}| = n$ and $\vec{a} \in S_1$, we denote by $\vec{\phi}(\vec{a})$ the vector $(\phi_1(\vec{a}), \ldots, \phi_n(\vec{a})) \in S_2^n$. We denote by $\mathbb{Z}_p[T_1, \ldots, T_n]$ the ring of multivariate polynomials in indeterminates $T_1, \ldots, T_n$. For a polynomial $P \in \mathbb{Z}_p[T_1, \ldots, T_n]$, we denote $P(T_1, \ldots, T_n)$ by $P(\vec{T})$ and by $P(\vec{a})$ the evaluation of P by setting $\vec{T}$ to $\vec{a}$, meaning that we set $T_1 = a_1, \ldots, T_n = a_n$. For $F\colon \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ and for a vector $\vec{x}$ over $\mathcal{D}$, we denote by $F(K, \vec{x})$ the vector $(F(K, x_1), \ldots, F(K, x_{|\vec{x}|}))$. We denote by $\mathfrak{S}_n$ the set of all permutations of $\{1, \ldots, n\}$.

Finally, we often implicitly consider a multiplicative group $\mathbb{G} = \langle g \rangle$ with public generator g of order p and we denote by $[[]\, g]a$, or simply $[a]$ if there is no ambiguity about the generator, the element g^a, for any $a \in \mathbb{Z}_p$. Similarly, if $\boldsymbol{A}$ is a matrix in $\mathbb{Z}_p^{k \times m}$, $[\boldsymbol{A}]$ is a matrix $\boldsymbol{U} \in \mathbb{G}^{k \times m}$, such that $u_{i,j} = [a_{i,j}]$ for $i = 1, \ldots, k$ and $j = 1, \ldots, m$.

Games [10]. Most of our definitions and proofs use the code-based game-playing framework, in which a game has an **Initialize** procedure, procedures to respond to adversary oracle queries, and a **Finalize** procedure. To execute a game G with an adversary $\mathscr{A}$, we proceed as follows. First, **Initialize** is executed and its outputs become the input of $\mathscr{A}$. When $\mathscr{A}$ executes, its oracle queries are

answered by the corresponding procedures of G. When $\mathscr{A}$ terminates, its outputs become the input of **Finalize**. The output of the latter, denoted $\mathrm{G}^{\mathscr{A}}$ is called the output of the game, and we let "$\mathrm{G}^{\mathscr{A}} \Rightarrow 1$" denote the event that this game output takes the value 1. The running time of an adversary by convention is the worst case time for the execution of the adversary with any of the games defining its security, so that the time of the called game procedures is included.

PRFs [8,19]. The advantage of an adversary $\mathscr{A}$ in attacking the standard PRF security of a function $F\colon \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ is defined via

$$\mathbf{Adv}^{\mathsf{prf}}_{F}(\mathscr{A}) = \Pr\left[\mathrm{PRFReal}^{\mathscr{A}}_{F} \Rightarrow 1\right] - \Pr\left[\mathrm{PRFRand}^{\mathscr{A}}_{F} \Rightarrow 1\right].$$

Game $\mathrm{PRFReal}_F$ first picks $K \xleftarrow{\$} \mathcal{K}$ and responds to oracle query $\mathbf{Fn}(x)$ via $F(K,x)$. Game $\mathrm{PRFRand}_F$ first picks $f \xleftarrow{\$} \mathsf{Fun}(\mathcal{D},\mathcal{R})$ and responds to oracle query $\mathbf{Fn}(x)$ via $f(x)$.

RKA-PRFs [8,9]. Let $F\colon \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ be a function and $\Phi \subseteq \mathsf{Fun}(\mathcal{K},\mathcal{K})$. The members of Φ are called RKD (Related-Key Deriving) functions. An adversary is said to be Φ-restricted if its oracle queries (ϕ, x) satisfy $\phi \in \Phi$. The advantage of a Φ-restricted adversary $\mathscr{A}$ in attacking the RKA-PRF security of F is defined via

$$\mathbf{Adv}^{\mathsf{prf\text{-}rka}}_{\Phi,F}(\mathscr{A}) = \Pr\left[\mathrm{RKPRFReal}^{\mathscr{A}}_{F} \Rightarrow 1\right] - \Pr\left[\mathrm{RKPRFRand}^{\mathscr{A}}_{F} \Rightarrow 1\right].$$

Game $\mathrm{RKPRFReal}_F$ first picks $K \xleftarrow{\$} \mathcal{K}$ and responds to oracle query $\mathbf{RKFn}(\phi, x)$ via $F(\phi(K), x)$. Game $\mathrm{RKPRFRand}_F$ first picks $K \xleftarrow{\$} \mathcal{K}$ and $G \xleftarrow{\$} \mathsf{Fun}(\mathcal{K},\mathcal{D},\mathcal{R})$ and responds to oracle query $\mathbf{RKFn}(\phi, x)$ via $G(\phi(K), x)$. We say that F is a Φ-RKA-secure PRF if for any Φ-restricted adversary, its advantage in attacking the RKA-PRF security is negligible.

Group Generators. All our PRFs and RKA-PRFs use a cyclic group of prime order p. The generator(s) used in their construction is supposed to be public. In particular, RKD functions *cannot* modify the generator(s). Our security proofs will then start by giving the generators to the adversary.

Hardness Assumptions. To get a simpler and unified framework, we introduce a particular MDDH assumption [18]: the $\mathcal{E}_{k,d}$-MDDH assumption, defined by the matrix distribution $\mathcal{E}_{k,d}$ which samples matrices $\boldsymbol{\Gamma}$ as follows

$$\boldsymbol{\Gamma} = \begin{pmatrix} \boldsymbol{A}_1^0 \cdot \boldsymbol{A}_0 \\ \boldsymbol{A}_1^1 \cdot \boldsymbol{A}_0 \\ \vdots \\ \boldsymbol{A}_1^d \cdot \boldsymbol{A}_0 \end{pmatrix} \in \mathbb{Z}_p^{k(d+1)\times k} \quad \text{with } \boldsymbol{A}_0, \boldsymbol{A}_1 \xleftarrow{\$} \mathbb{Z}_p^{k\times k}. \tag{1}$$

The advantage of an adversary $\mathscr{D}$ against the $\mathcal{E}_{k,d}$-MDDH assumption is

$$\mathbf{Adv}^{\mathcal{E}_{k,d}\text{-}\mathsf{mddh}}_{\mathbb{G}}(\mathscr{D}) = \Pr\left[\,\mathscr{D}(g, [\boldsymbol{\Gamma}], [\boldsymbol{\Gamma}\cdot\boldsymbol{W}])\,\right] - \Pr\left[\,\mathscr{D}(g, [\boldsymbol{\Gamma}], [\boldsymbol{U}])\,\right],$$

Table 1. Security of $\mathcal{E}_{k,d}$-MDDH

	$k=1$	$k=2$	$k\geq 3$
$d=1$	$= \mathbf{Adv}^{\mathsf{ddh}}_{\mathbb{G}}$	$\lesssim 2\cdot \mathbf{Adv}^{\mathcal{U}_2\text{-mddh}}_{\mathbb{G}}$	$\lesssim k\cdot \mathbf{Adv}^{\mathcal{U}_k\text{-mddh}}_{\mathbb{G}}$
$d\geq 2$	$\lesssim d\cdot \mathbf{Adv}^{d\text{-ddhi}}_{\mathbb{G}}$	generic bilinear group[a]	?[b]

$\mathbf{Adv}^{\mathsf{ddh}}_{\mathbb{G}}$, $\mathbf{Adv}^{d\text{-ddhi}}_{\mathbb{G}}$ and $\mathbf{Adv}^{\mathcal{U}_k\text{-mddh}}_{\mathbb{G}}$ are advantages for DDH, DDHI, and $\mathcal{U}_k$-MDDH. This latter assumption is weaker than k-Lin;
[a] Proven in the generic (symmetric) bilinear group model [14] in the full version;
[b] (Trivially) secure in the generic cyclic group model [30], but nothing known about security in generic (symmetric) k-linear group model [21,29]

where $\boldsymbol{\Gamma} \xleftarrow{\$} \mathcal{E}_{k,d}$, $\boldsymbol{W} \xleftarrow{\$} \mathbb{Z}_p^{k\times 1}$, $\boldsymbol{U} \xleftarrow{\$} \mathbb{Z}_p^{k(d+1)\times 1}$. As any MDDH assumption and as recalled in the full version, this assumption is random self-reducible, which enables us to make relatively tight proofs.

In Table 1, we summarize security results for $\mathcal{E}_{k,d}$-MDDH. For $k=1$ or $d=1$, the $\mathcal{E}_{k,d}$-MDDH assumption is implied by standard assumptions (DDH, DDHI, or k-Lin, recalled in the full version). $\mathcal{E}_{1,1}$-MDDH is actually exactly DDH.

For our RKA framework, we also make use of the d-Strong Discrete Logarithm (SDL) problem given in [20] and recalled in the full version.

3 Linearly Independent Polynomial Security

In this section, we define a new security notion, termed linearly independent polynomial (LIP) security, which captures that, given a cyclic group $\mathbb{G}=\langle g\rangle$ of prime order p, the hardness of distinguishing a tuple $(y_1,\ldots,y_q) = ([P_1(\vec{a})\cdot a'],\ldots,[P_q(\vec{a})\cdot a']) \in \mathbb{G}^q$ from a random tuple in $(y_1,\ldots,y_q) \xleftarrow{\$} \mathbb{G}^q$, where $\vec{a}$ is a secret random vector in $\mathbb{Z}_p^n$, a' is a secret random scalar in $\mathbb{Z}_p$, and P_j are linearly independent multivariate polynomials. Our LIP theorem (Theorem 1) shows that distinguishing these two tuples is harder than the $\mathcal{E}_{1,d}$-MDDH problem in $\mathbb{G}$, where d is the maximum degree in one indeterminate in polynomials $P_1,\ldots,P_q$. We point out that, on the one hand, if there were a linear relation between the polynomials, i.e., if there exists $(\lambda_1,\ldots,\lambda_q)\in\mathbb{Z}_p^q\backslash\{(0,\ldots,0)\}$, such that $\sum_{j=1}^q \lambda_j P_j = 0$, then it would be straightforward to break the LIP security by checking whether $\prod_{j=1}^q {y_j}^{\lambda_j} = 1$ (real case) or not (random case). So the linear independence of the P_j's is required.

On the other hand, if the polynomials P_j are linearly independent, then distinguishing the two tuples is hard in the generic group model, since in this model, the adversary can only compute linear combinations of the group elements it is given (and check for equality). The LIP security is therefore not surprising. What is surprising, is that it is possible to prove it under classical assumptions such as $\mathcal{E}_{1,d}$-MDDH, without an exponential blow-up.

In the following, we first consider a particular case of the LIP theorem in which the polynomials are given in their expanded form. This section not only serves as a warm-up for the sequel, but it also helps better grasp the challenges of the proof of the full theorem and gives a nice overview. Next, we formally state the LIP theorem.

3.1 Warm-Up: Expanded Multilinear Polynomials

As a warm-up, let us first suppose the polynomials P_j are multilinear and given in their expanded form: $P_j \in \mathbb{Z}_p[T_1, \ldots, T_n]$ and

$$P_j(\vec{T}) = \sum_{i \in \{0,1\}^n} \alpha_{j,i} T_1^{i_1} \cdots T_n^{i_n}.$$

There are 2^n monomials $T_1^{i_1} \cdots T_n^{i_n}$, even in that restricted case. So we need to suppose that either n is logarithmic in the security parameter, or, more generally, only a polynomial (in the security parameter) number of $\alpha_{j,i}$ are non-zero.

Let us now prove the LIP security of these polynomials. In the real case, we have:

$$y_j = [P_j(\vec{a})a'] = \left[\sum_{i \in \{0,1\}^n} \alpha_{j,i} a_1^{i_1} \cdots a_n^{i_n} a' \right] = \prod_{i \in \{0,1\}^n} \mathsf{NR}((a', \vec{a}), i)^{\alpha_{j,i}}, \qquad (2)$$

where $\mathsf{NR}((a', \vec{a}), i) = \left[a' \prod_{k=1}^n a_k^{i_k}\right]$ (for $i \in \{0,1\}^n$). NR is a secure PRF under the DDH assumption, meaning that all the values $\mathsf{NR}((a', \vec{a}), i)$ for all $i \in \{0,1\}^n$ look independent and uniformly random. Let us write $\vec{U}$ the column vector, with rows indexed by $i \in \{0,1\}^n$, containing all the discrete logarithm of these values, i.e., $u_i = a' \prod_{k=1}^n a_k^{i_k}$. Let us also write $\boldsymbol{M}$ the $q \times 2^n$ matrix, with columns indexed by $i \in \{0,1\}^n$, defined by $m_{j,i} = \alpha_{j,i}$. Then we can rewrite (2) as:

$$\left(y_1 \ldots y_q\right)^\intercal = \left[\boldsymbol{M} \cdot \vec{U}\right].$$

Since the polynomials P_j are linearly independent, the rows of $\boldsymbol{M}$ are linearly independent. Therefore, as $\left[\vec{U}\right]$ looks uniformly random in $\mathbb{G}^{2^n}$, $(y_1, \ldots, y_q)$ looks like a uniformly random tuple in $\mathbb{G}^q$. This proves the result of the LIP theorem in this multilinear case with expanded polynomial. Extending this result to non multilinear polynomial would just require slightly changing the assumption, as long as polynomials are given in their expanded form.

This result is already very useful. We will see in Sect. 4 that it enables to prove the security of the Naor-Reingold PRF and variants thereof.

Challenges for its Extension. Unfortunately, for certain settings such as those considered in the context of related-key security, or even for the Boneh-Montgomery-Raghunathan PRF [17], we cannot have polynomials in an expanded form, but only as a polynomial-size (in the number n of indeterminates and the maximum degree d in each indeterminate) formula (given by an abstract tree).[1] The problem is that the expanded version of these polynomials may be exponentially large. For example, $(T_1 + 1) \cdots (T_n + 1)$ has 2^n monomials.

Therefore, the main challenge is to prove the theorem without expanding the polynomials. This requires a much more subtle proof that we sketch here. This

[1] Details on the representation of polynomials are given in the full version.

first idea is the following: instead of replacing all monomials by independent random values at once, we first fix all values $T_2, \ldots, T_n$ to randomly chosen $a_2, \ldots, a_n$, and get polynomials in T_1 only. These polynomials can be expanded without an exponential blow-up, and each monomial $T_1, T_1^2, \ldots$ can be replaced by an independent random value (instead of $a_1, a_1^2, \ldots$ for some value a_1). Then, we can fix only $T_3, \ldots, T_n$ to randomly chosen $a_3, \ldots, a_n$, get a polynomial in T_1 and T_2, and replace all distinct monomial $(T_1, T_1^2, T_1T_2, T_2^2, \ldots)$ by independent random values. And we can continue like that until all monomials are replaced.

Obviously, if we do that so naively, we get back to the original problem: we have an exponential number of monomials. The second idea is to remark that we actually do not need to expand polynomials to replace all distinct monomials by random values and get the result, at each step of the previous idea. We could just assign random values to all polynomials (after fixing $T_{i+1}, \ldots, T_n$ to $a_{i+1}, \ldots, a_n$), if they are all linearly independent: this is exactly what we showed in the previous proof for expanded polynomials. And if they are not all linearly independent, we just need to take care of linear combinations, and compute the resulting value accordingly.

More precisely, for any polynomial P, let us write $Q_P \in \mathbb{Z}_p[T_1, \ldots, T_i]$ the polynomial obtained after fixing $T_{i+1}, \ldots, T_n$ to $a_{i+1}, \ldots, a_n$. To answer the j-th query P_j, we check whether Q_{P_j} is linearly independent from $(Q_{P_l})_{l=1,\ldots,j-1}$. If that is the case, we answer with an independent random value y_j. Otherwise, we find some linear combination between Q_{P_j} and $(Q_{P_l})_{l=1,\ldots,j-1}$, and we write $Q_{P_j} = \sum_{l=1}^{j-1} \lambda_l Q_{P_l}$ and outputs $\prod_{l=1}^{j-1} y_l^{\lambda_l}$, with y_l the output given for P_l.

The last difficulty is that this proof requires a test of linear dependence of multivariate polynomials. One way to do that would be to expand them, which is exactly what we are trying to avoid. So, instead, we use a statistical test based on the Schwartz-Zippel lemma, which basically consists in evaluating the polynomials in enough random points and looking for linear combination among the vectors of these evaluations.

proc Initialize	**proc Pl**(P)	**proc Finalize**(b')
$\vec{a} \xleftarrow{\$} \mathbb{Z}_p^n$; $a' \xleftarrow{\$} \mathbb{Z}_p$ $b \xleftarrow{\$} \{0,1\}$	If $b = 0$ then $y \leftarrow [P(\vec{a}) \cdot a']$ else $y \xleftarrow{\$} \mathbb{G}$ Return y	Return $b' = b$

Fig. 1. Game defining the (n, d)-LIP security for a group $\mathbb{G}$

3.2 Main Theorem: LIP Security

LIP **Security.** Let $\mathbb{G} = \langle g \rangle$ be a group of prime order p. We define the advantage of an adversary $\mathscr{A}$ against the (n, d)-LIP security of $\mathbb{G}$, denoted $\mathbf{Adv}_{\mathbb{G}}^{(n,d)\text{-lip}}(\mathscr{A})$ as the probability of success in the game defined in Fig. 1, with $\mathscr{A}$ being restricted to make queries $P \in \mathbb{Z}_p[T_1, \ldots, T_n]$ such that for any query P, the maximum degree in one indeterminate in P is at most d, and for any sequence $(P_1, \ldots, P_q)$ of queries, the polynomials $(P_1, \ldots, P_q)$ are always *linearly independent* over $\mathbb{Z}_p$. Another way to look at the security definition is to consider that when $b = 0$,

$\mathbf{PI}(P)$ outputs $[P(\vec{a})]_h = [P(\vec{a}) \cdot a']_g$, where the generator is $h = [a']_g$, which is not public (but can be obtained by querying the polynomial 1), and g is a public generator.

Theorem 1 (LIP). *Let $\mathbb{G} = \langle g \rangle$ be a group of prime order p. Let $\mathscr{A}$ be an adversary against the (n,d)-LIP security of $\mathbb{G}$ that makes q oracle queries $P_1, \ldots, P_q$. Then we can design an adversary $\mathscr{B}$ against the $\mathcal{E}_{1,d}$-MDDH problem in $\mathbb{G}$, such that $\mathbf{Adv}^{(n,d)\text{-lip}}_{\mathbb{G}}(\mathscr{A}) \leq n \cdot d \cdot \mathbf{Adv}^{\mathcal{E}_{1,d}\text{-mddh}}_{\mathbb{G}}(\mathscr{B}) + O(ndq/p)$. The running time of $\mathscr{B}$ is that of $\mathscr{A}$ plus the time to perform a polynomial number (in q, n, and d) of operations in $\mathbb{Z}_p$ and $\mathbb{G}$.*

The proof is detailed in the full version.

4 Recovering and Extending Existing Number-Theoretic PRFs

In Table 2, we recall known number-theoretic PRFs, namely the Naor-Reingold (NR) PRF [27], its variant NR^* defined in [8], and the algebraic PRF by Boneh, Montgomery, and Raghunathan (BMR) in [17]. We also introduce weighted (extended) versions of these PRFs, namely weighted NR (WNR) and weighted BMR (WBMR), in order to construct RKA-secure PRFs for new classes of RKD functions (Sect. 5). These weighted PRFs are obtained by applying particular permutations to the key space. Then, as PRFs, it is straightforward that the security of NR and BMR implies the security of their weighted versions. However, as detailed in Sect. 5, in the RKA setting, we can prove that some of these weighted PRFs are secure against certain classes of RKD functions while both NR and BMR are not, even if we apply the BC/ABPP frameworks.

Using the LIP theorem and changing the generators used (to get PRFs of the form $F(\vec{a}, x) = [P_x(\vec{a}) \cdot a'])$, the security proof of WNR and WBMR is straightforward, and so is the security proof of NR, NR^*, and BMR, as particular cases of WNR and WBMR. Concretely, for $\mathsf{WBMR}^{\vec{w}}$, we start by revealing the generator h to the adversary where

$$h = \left[\left(\prod_{i=1}^{n} \prod_{k \in \{0,\ldots,d\}} (a_i + w_i + k) \right) \cdot a' \right]_g = [P(\vec{a}) \cdot a']_g$$

which is a generator with overwhelming probability. Then, when the adversary makes a query x, it is clear that

$$\left[\prod_{i=1}^{n} \frac{1}{a_i + w_i + x_i} \right]_h = \left[\left(\prod_{i=1}^{n} \prod_{k \in \{0,\ldots,d\} \setminus \{x_i\}} (a_i + w_i + k) \right) \cdot a' \right]_g = [P_x(\vec{a}) \cdot a']_g$$

As each polynomial P_x is null on every input $-x'$ for $x' \in \{0, \ldots, d\}^n$, seen as a vector of $\mathbb{Z}_p^n$, except when $x' = x$, and as P is null on all $-x'$, P and $(P_x)_x$ are linearly independent. Then, we conclude the security proof of $\mathsf{WBMR}^{\vec{w}}$ by applying the LIP theorem. Formal proofs are provided in the full version.

Table 2. Existing number-theoretic PRFs and their weighted extensions

PRF F	Key $\vec{a}$ Key domain $\mathcal{K}$	Domain $\mathcal{D}$	Output	$\mathbf{Adv}_F^{\mathsf{prf}} \lesssim$
NR	$(a_0, \ldots, a_n)$ $\mathcal{K} = \mathbb{Z}_p^{n+1}$	$\{0,1\}^n$	$\left[a_0 \prod_{i=1}^{n} a_i^{x_i}\right]$	$n \cdot \mathbf{Adv}_{\mathbb{G}}^{\mathsf{ddh}}$
NR*	$(a_1, \ldots, a_n)$ $\mathcal{K} = \mathbb{Z}_p^{n}$	$\{0,1\}^n \setminus \{0^n\}$	$\left[\prod_{i=1}^{n} a_i^{x_i}\right]$	$n \cdot \mathbf{Adv}_{\mathbb{G}}^{\mathsf{ddh}}$
BMR	$(a_1, \ldots, a_n)$ $\mathcal{K} = \mathbb{Z}_p^{n}$	$\{0, \ldots, d\}^n$	$\left[\prod_{i=1}^{n} \frac{1}{a_i + x_i}\right]$	$nd \cdot \mathbf{Adv}_{\mathbb{G}}^{d\text{-}\mathsf{ddhi}}$
$\mathsf{WNR}^{\vec{w}}$ $(\vec{w} \in \mathbb{Z}_p^{n+1})^*$	$(a_0, \ldots, a_n)$ $\mathcal{K} = \mathbb{Z}_p^{n+1}$	if $w_0 \neq 0$: $\{0,1\}^n$, else: $\{0,1\}^n \setminus \{0^n\}$	$\left[a_0^{w_0} \prod_{i=1}^{n} a_i^{w_i x_i}\right]$	$n \cdot \mathbf{Adv}_{\mathbb{G}}^{\mathsf{ddh}\,\dagger}$
$\mathsf{WBMR}^{\vec{w}}$ $(\vec{w} \in \mathbb{Z}_p^{n})^{\ddagger}$	$(a_1, \ldots, a_n)$ $\mathcal{K} = \mathbb{Z}_p^{n}$	$\{0, \ldots, d\}^n$	$\left[\prod_{i=1}^{n} \frac{1}{a_i + w_i + x_i}\right]$	$nd \cdot \mathbf{Adv}_{\mathbb{G}}^{d\text{-}\mathsf{ddhi}}$

$\mathbb{G} = \langle g \rangle$ is a prime order group, and g is the generator used for the PRF construction;
The last column show approximate simplified bounds on the advantage $\mathbf{Adv}_F^{\mathsf{prf}}$ of a polynomial-time adversary against the security of the PRF F; exact bounds can be found in the full version;
Remarks: $\mathsf{NR} = \mathsf{WNR}^{(1,\ldots,1)}$, $\mathsf{NR}^* = \mathsf{WNR}^{(0,1,\ldots,1)}$, and $\mathsf{BMR} = \mathsf{WBMR}^{(0,\ldots,0)}$;
[a] For WNR, weights are $\vec{w} = (w_0, \ldots, w_n) \in \mathbb{Z}_p^{n+1}$;
[b] When $w_1, \ldots, w_n$ are coprime to $p-1$, and w_0 is 0 or coprime to $p-1$;
[c] For WBMR, weights are $\vec{w} = (w_1, \ldots, w_n) \in \mathbb{Z}_p^{n}$.

5 Application to Related-Key Security

In this section, we show how our theorem can be used to build RKA-secure PRFs from a PRF F defined over a prime order group $\mathbb{G} = \langle g \rangle$ that takes a key $\vec{a}$ and an input x and outputs a group element $F(\vec{a}, x) = [P_x(\vec{a})]$. Let Φ be a class of RKD functions, where functions $\vec{\phi} = (\phi_1, \ldots, \phi_n) \in \Phi$ are such that ϕ_i are multivariate polynomials in $\mathbb{Z}_p[T_1, \ldots, T_n]$. Then, for an RKD function $\vec{\phi}$ and an input x, the PRF outputs $F(\vec{\phi}(\vec{a}), x) = \left[P_{\vec{\phi},x}(\vec{a})\right]$, where the polynomial $P_{\vec{\phi},x}(\vec{T}) = P_x(\vec{\phi}(\vec{T})) = P_x(\phi_1(\vec{T}), \ldots, \phi_n(\vec{T}))$ depends on $\vec{\phi}$ and x. In particular, $P_{\mathsf{id},x} = P_x$ for all x, where id is the identity function.

When all polynomials $P_{\vec{\phi},x}$ and the constant polynomial 1 are linearly independent, the LIP theorem directly shows that F is Φ-RKA-secure. To illustrate this, we construct in Sect. 5.1 a PRF that is secure against permutations of the secret key using this method.

However, to assume that all polynomials $P_{\vec{\phi},x}$ are linearly independent is a very strong property and, in general, this is not the case for all x and $\vec{\phi}$. Hence, in Sect. 5.2, we consider the less restrictive case where the polynomials $P_{\vec{\phi}_1,x_1}, \ldots, P_{\vec{\phi}_q,x_q}$ are linearly independent as long as the inputs $x_1, \ldots, x_q$ are distinct (in which case the adversary is said to be unique-input). More precisely, we first design a new algebraic framework that extends the one from [1], when the PRF F is of the form $[P_x(\vec{a})]$ and the RKD functions are multivariate poly-

nomials, and then use it to construct RKA-secure PRFs from F for new and larger classes of RKD functions.

5.1 Direct Constructions of RKA-Secure PRFs

In this section, we show how the LIP theorem can be used to prove the Φ-RKA-PRF security in the particular case where all polynomials $P_{\vec{\phi},x}$ are linearly independent, for any $\vec{\phi} \in \Phi$ and any input x.

Specifically, we consider the class $\Phi_{\mathfrak{S}_n}$ of functions defined as $\{\sigma \mid \sigma \in \mathfrak{S}_n\}$ such that, applying a function $\sigma \in \Phi_{\mathfrak{S}_n}$ to a key $\vec{a} = (a_1, \ldots, a_n) \in \mathbb{Z}_p^n$ leads to the key $\sigma(\vec{a}) = (a_{\sigma^{-1}(1)}, \ldots, a_{\sigma^{-1}(n)})$, so the i-th component of $\vec{a}$ becomes the $\sigma(i)$-th component of the key $\sigma(\vec{a})$.

It is clear that BMR is not $\Phi_{\mathfrak{S}_n}$-RKA-secure, since we can distinguish BMR from a random function with only 2 queries. Indeed, let id be the identity function and (12) be the permutation which switches the first two components of the key. Then, one can just first query $(\mathsf{id}, 100\ldots0)$ and $((12), 010\ldots0)$ and check whether the output of these queries are the same, which is the case in the real case while they are independent in the random case. However, we show in what follows that a particular case of WBMR, defined below, is a $\Phi_{\mathfrak{S}_n}$-RKA-secure PRF.

Linear WBMR PRF. We define $\mathsf{WBMR}^{\mathsf{lin}}$ as the particular case of WBMR, where $w_i = (i-1)(d+1)$, for $i = 1, \ldots, n$. Please refer to Table 2 for details.

Theorem 2. *Let $\mathbb{G} = \langle g \rangle$ be a group of prime order p and let $\mathsf{WBMR}^{\mathsf{lin}}$ be the function defined above. Then we can reduce the $\Phi_{\mathfrak{S}_n}$-RKA-PRF security of $\mathsf{WBMR}^{\mathsf{lin}}$ to the hardness of the $(n(d+1)-1)$-DDHI problem in $\mathbb{G}$, with a loss of a factor $n(n(d+1)-1)$. Moreover, the time overhead of this reduction is polynomial in n, d and in the number of queries made by the adversary.*

The proof is given in the full version and is very similar to the proof of security of WBMR sketched in Sect. 4. The construction can actually be extended to also tolerates small additive factors in addition to permutations (see the full version).

5.2 Constructions via Unique-Input RKA-Secure PRFs

In this section, we address the less restrictive case where the polynomials $P_{\vec{\phi}_1,x_1}, \ldots, P_{\vec{\phi}_q,x_q}$ are linearly independent for any $\vec{\phi}_1, \ldots, \vec{\phi}_q$ only when the inputs $x_1, \ldots, x_q$ are all distinct. Please notice that this is the case for all the classes considered in [1,8]. We now denote by M the "original" PRF: $M(\vec{a}, x) = [P_x(\vec{a})]$.

In order to build RKA-secure PRFs from such PRFs, we would like to apply the ABPP generic framework [1] that allows to transform a PRF M which is RKA-secure with respect to unique-input adversaries (UI-RKA-secure) into an RKA-secure PRF F, when M is key-collision and statistical-key-collision secure. The latter means that it is hard to find two functions $\phi_1, \phi_2 \in \Phi$ such that

$\phi_1(K) = \phi_2(K)$, even with access to an oracle $(\phi, x) \mapsto f(\phi(K), x)$, when $f = M$ (key-collision security), and when f is a random function (statistical key-collision security). The framework consists in transforming this UI-RKA-secure PRF M into an RKA-secure PRF F, as follows:

$$F(K, x) = M(K, H(x, M(K, \vec{\omega}))),$$

where H is a compatible collision-resistant hash function, and the vector $\vec{\omega}$ is a strong key fingerprint, meaning that it is a vector of inputs such that the vector of outputs $M(K, \vec{\omega})$ completely defines K (recall that $M(K, \vec{\omega}) = (M(K, \omega_1), \ldots, M(K, \omega_{|\vec{\omega}|}))$. As defined in [8], a hash function is said to be compatible if it guarantees that the inner calls to M in the construction above will never collide with the outer calls to M even under related keys.

Unfortunately, if we consider the PRF $\mathsf{WNR}^{\vec{w}}$ with some $w_i > 1$, then it is not clear how to find a strong key fingerprint, which can be used to apply the ABPP framework. Furthermore, this ABPP framework requires to prove several non-algebraic properties (statistical or computational), namely key-collision, statistical-key-collision, and UI-RKA securities.

For this reason, we design a new algebraic framework, that generalizes the ABPP framework in the particular case of PRFs of the shape $M(\vec{a}, x) = [P_x(\vec{a})]$ and of RKD functions which are multivariate polynomials. For completeness, a more general framework, which does not make any assumptions about the shape of a PRF, is also given in the full version. Afterwards, we use our algebraic framework to design new RKA-secure PRFs based on WNR for larger classes for which previous constructions from [1,8] are not secure.

An Algebraic Framework for Related-Key Security. Here, we describe a new framework that transforms any PRF that satisfies that $P_{\vec{\phi}_1, x_1}, \ldots, P_{\vec{\phi}_q, x_q}$ are linearly independent, for any $\vec{\phi}_1, \ldots, \vec{\phi}_q$ as long as $x_1, \ldots, x_q$ are all distinct inputs, into a RKA-secure PRF. To do so, we first introduce three new notions, termed *algebraic fingerprint*, *helper information*, and *expansion function*, and defined as follows.

Group Generator. In this framework and its applications, we assume for simplicity that the generator used in the PRF construction, that is revealed to the adversary, is $[a']$.

Algebraic Fingerprint. In order to overcome the eventual lack of a strong key fingerprint, we introduce algebraic fingerprint, which will be used to replace $M(K, \vec{\omega})$ in the construction in [1], where $\vec{\omega}$ is a strong fingerprint. An algebraic fingerprintis simply an injective function $\vec{\Omega}$: $\mathbb{Z}_p^n \to \mathbb{G}^m$ such that the image $\vec{\Omega}(\vec{a})$ is a vector of group elements $([\Omega_1(\vec{a})a'], \ldots, [\Omega_m(\vec{a})a'])$ with $\Omega_1, \ldots, \Omega_m$ being polynomials in $\mathbb{Z}_p[T_1, \ldots, T_n]$ and $a' \in \mathbb{Z}_p$. In our applications, we will simply have $\vec{\Omega}(\vec{a}) = ([a_1 a'], \ldots, [a_n a'])$, so $m = n$ and $\Omega_i(\vec{T}) = T_i$ for $i = 1, \ldots, n$.

Helper Information. In order to prove the security of our framework, we need to be able to compute the image of the algebraic fingerprint, $\vec{\Omega}(\vec{\phi}(\vec{a})) = ((\Omega_1 \circ \vec{\phi})(\vec{a}), \ldots, (\Omega_m \circ \vec{\phi})(\vec{a}))$, for any related key $\vec{\phi}(\vec{a}) \in \mathbb{Z}_p^n$, with $\vec{\phi} \in \Phi$, from

some information which can somehow be made public without hurting security. We call this information a helper information, write it $\mathsf{Help}_\Phi(\vec{a})$, and call Help_Φ the helper function. We suppose that $\mathsf{Help}_\Phi(\vec{a}) = ([\mathsf{help}_1(\vec{a})a'], \ldots, [\mathsf{help}_l(\vec{a})a'])$, with $\mathsf{help}_1, \ldots, \mathsf{help}_l$ linearly independent polynomials which generate a vector subspace of $\mathbb{Z}_p[T_1, \ldots, T_n]$ containing the polynomials $\Omega_i \circ \vec{\phi}$ for $i = 1, \ldots, m$, and $\vec{\phi} \in \Phi$.

Hash Function and Expansion Function. Let $\overline{\mathcal{D}} = \mathcal{D} \times \mathbb{G}^m$ where $\mathcal{D}$ is the domain of the PRF M, and let h be a collision-resistant hash function $h\colon \overline{\mathcal{D}} \to \mathsf{hSp}$ (definition recalled in the full version), where hSp is a large enough space. The last thing we need to define is an expansion function, which is simply an injective function $\mathsf{E}\colon \mathsf{hSp} \to \mathcal{S} \subseteq \mathcal{D}$ such that for any sequence $(\vec{\phi}_1, x_1), \ldots, (\vec{\phi}_q, x_q)$ where $x_1, \ldots, x_q$ are distinct inputs in $\mathcal{S}$ and $\vec{\phi}_1, \ldots, \vec{\phi}_q$ are RKD functions, polynomials $\mathsf{help}_1, \ldots, \mathsf{help}_l$ and polynomials $P_{\vec{\phi}_1, x_1}, \ldots, P_{\vec{\phi}_q, x_q}$ and 1 (which needs to be queried to define the generator $[a']$) are linearly independent over $\mathbb{Z}_p$ (in particular, E has to be injective).

Using these new tools, we obtain the following framework.

Theorem 3. *Let $\mathbb{G}$ be a group of prime order p. We use the above definitions, with $M\colon \mathbb{Z}_p^n \times \mathcal{D} \to \mathbb{G}$ defined by $M(\vec{a}, x) = [P_x(\vec{a})]$. Let d be a upper bound for the maximum degree in any indeterminate of polynomials in $\{\mathsf{help}_1, \ldots, \mathsf{help}_l\} \cup \{P_{x,\vec{\phi}} \mid x \in \mathcal{S}, \vec{\phi} \in \Phi\}$. Define $F\colon \mathbb{Z}_p^n \times \mathcal{D} \to \mathbb{G}$ by*

$$F(\vec{a}, x) = M(\vec{a}, \mathsf{E}(h(x, \vec{\Omega}(\vec{a}))))$$

for all $\vec{a} \in \mathbb{Z}_p^n$ and $x \in \mathcal{D}$. Then, we can reduce the Φ-RKA-PRF security of F to the (n, d)-LIP security, the collision-resistance security of h without any loss, and to the [-SDLd] assumption with a loss of a factor $2n$. The running time overhead of this reduction is polynomial in n, d and q.

Proof Overview. The proof of the above theorem is detailed in the full version and relies on the sequence of 10 games (games $\mathrm{G}_0 - \mathrm{G}_9$). We first prove an intermediate statement whose proof is very similar to the proof of Theorem 3.1 from [1], under a notion termed extended key-collision security (that states the hardness of finding key collisions given access to PRF values and helper information) which is defined in the appendix. Afterwards, we reduce this notion to the hardness of the SDL in $\mathbb{G}$. Here we provide a brief overview of the proof of the intermediate statement.

We start by giving the generator used for the PRF by querying polynomial 1. Hence, the generator is simply $[a']$. Since we may have key collisions (i.e., two RKD functions $\phi_1 \neq \phi_2$, such that $\phi_1(\vec{a}) = \phi_2(\vec{a})$), we start by dealing with possible collisions on the related keys in the RKAPRFReal case, using the extended key-collision notion (games $\mathrm{G}_0 - \mathrm{G}_2$). These claws can be detected by looking for collisions on images of $\vec{\Omega}$ for different RKD functions.

Then, in games $G_3 - G_4$, we deal with possible collisions on hash values in order to ensure that the inputs $t = \mathsf{E}(h(x, \vec{\Omega}(\vec{a})))$ used to compute the output y are distinct (recall that E is injective).

Then, we use the (n, d)-LIP security notion to show that it is hard to distinguish the output of F and the helper information from uniformly random values (games $G_5 - G_6$).

Finally, we use once again the extended-key-collision security notion to deal with possible key collisions in the RKAPRFRand case (games $G_7 - G_9$) so that G_9 matches the description of the RKAPRFRand game. These key collisions can still be detected in these games by making crucial use of the helper information.

RKA-PRFs for Permutations of Univariate Polynomial Functions. We now apply our framework to a particular case of WNR and build the first RKA-secure PRF secure against permutations of univariate polynomials. We chose to set w_0 to 0 in our construction in order to ease the readability so that the key space of the PRF stays $\mathbb{Z}_p^n$, but similar results can be proven with $w_0 = 1$ or set to a prime number $p_0 > d$ (and distinct to $p_1, \ldots, p_n$ defined below).

For $d \geq 1$, let Φ_d be the class of degree at most d non-constant univariate polynomials defined as $\Phi_d = \{\vec{\phi}\colon\ \mathbb{Z}_p^n \to \mathbb{Z}_p^n \mid \phi_i : \vec{T} \mapsto \sum_{j=0}^{d} \alpha_{i,j} T_i^j, (\alpha_{i,1}, \ldots, \alpha_{i,d}) \neq 0^d, \forall i = 1, \ldots, n\}$. Then we consider the class $\Phi_{\mathfrak{S}_n,d}$ of permutations of degree at most d non-constant univariate polynomials, defined as follows:

$$\Phi_{\mathfrak{S}_n,d} = \{\sigma \circ \vec{\phi} \mid (\sigma, \vec{\phi}) \in \mathfrak{S}_n \times \Phi_d\} .$$

For a key $\vec{a} = (a_1, \ldots, a_n) \in \mathbb{Z}_p^n$, applying an RKD function $\sigma \circ \vec{\phi} \in \Phi_{\mathfrak{S}_n,d}$, where $\vec{\phi} = (\phi_1, \ldots, \phi_n)$ leads to the key $(\phi_{\sigma^{-1}(1)}(\vec{a}), \ldots, \phi_{\sigma^{-1}(n)}(\vec{a})) \in \mathbb{Z}_p^n$, so the i-th component a_i of the key is changed into $\phi_i(\vec{a})$ and becomes the $\sigma(i)$-th component of the related key.

Before explaining our construction, we would like to point out that, even if we just consider the simple class of permutations $\Phi_{\mathfrak{S}_n} \subset \Phi_{\mathfrak{S}_n,1}$ introduced in Sect. 5.1, we can already show that NR and NR^* are not $\Phi_{\mathfrak{S}_n}$-RKA secure, even with respect to unique-input adversaries.

Indeed, let us consider NR^*: let id be the identity function and (12) be the permutation which switches the first two components of the key. Then, the output of the queries $(\mathsf{id}, 100\ldots0)$ and $((12), 010\ldots0)$ will be the same in the real case and independent in the random case.

In fact, we can generalize the attack above to show that there even exists a compatible collision-resistant hash function h such that the PRF that one obtains when applying the Bellare-Cash (or ABPP) transform to NR^* would not be RKA-secure with respect to the class of permutations. Indeed, let h' be a collision-resistant hash function. The counter-example for h could be as follows (where x_1 and x_2 are two arbitrary distinct inputs):

$$h(x, [a_1], \ldots, [a_n]) = \begin{cases} 1110 \,\|\, h'(x_1, [a_1], \ldots, [a_n]) & \text{if } x = x_1 \\ 1101 \,\|\, h'(x_1, [a_2], [a_1], [a_3], \ldots, [a_n]) & \text{if } x = x_2 \\ 1111 \,\|\, h'(x, [a_1], \ldots, [a_n]) & \text{otherwise.} \end{cases}$$

Note that h is a compatible collision-resistant hash function. It is easy to see that the output of the queries (id, x_1) and $((12), x_2)$ will be the same in the real case and independent in the random case. The same kind of attack can be mounted against NR.

However, while NR and NR^* are not RKA-secure against permutations attacks, we show in what follows that a particular case of WNR, defined below, yields a $\Phi_{\mathfrak{S}_n,d}$-RKA-secure PRF.

d-Linear Weighted NR PRF. Let $d \geq 1$. Let $p_1 < p_2 < \cdots < p_n$ be distinct prime numbers such that $p_1 > d$. We define $\mathsf{WNR}^{d\text{-}\mathsf{lin}}$ as the particular case of WNR, where $w_0 = 0$ and $w_i = p_i$. Please refer to Table 2 for details. Using standard inequalities over prime numbers, it is easy to see that we can find $p_1, \ldots, p_n$ such that $p_n = \tilde{O}(d+n)$.

In order to apply the framework from Theorem 3 to $\mathsf{WNR}^{d\text{-}\mathsf{lin}}$ and $\Phi_{\mathfrak{S}_n,d}$, we define:

- $[a'] \in \mathbb{G}$ is the generator used for the PRF construction
- $\vec{\Omega}$: $\vec{a} \in \mathbb{Z}_p^n \mapsto ([a_1 a'], \ldots, [a_n a']) \in \mathbb{G}^n$
- $\mathsf{Help}_{\Phi_{\mathfrak{S}_n,d}}$: $\vec{a} \in \mathbb{Z}_p^n \mapsto ([a'], [a_1 a'], \ldots, [a_1^d a'], \ldots, [a_n a'], \ldots, [a_n^d a']) \in \mathbb{G}^{nd+1}$
- h can be any collision-resistant hash function h: $\{0,1\}^n \times \mathbb{G}^n \to \{0,1\}^{n-2}$
- E: $z \in \{0,1\}^{n-2} \mapsto 11 \,\|\, z \in \{0,1\}^n$.

We just need to prove that E satisfies the linear independence property required to apply the framework, which is done in the full version, and sketched here. We order monomials of multivariate polynomials, with any order respecting the total degree of polynomials (e.g., the graded lexicographic order). The leading monomial (i.e., the first monomial for that order) of the polynomial $P_{\vec{\phi},x}$ is $T_1^{x_{\sigma(1)} p_{\sigma(1)} d_1} \cdots T_n^{x_{\sigma(n)} p_{\sigma(n)} d_n}$, with $d_i > 0$ the degree of ϕ_i. The polynomials for the helper information (help_k) are T_i^j. Therefore, the leading monomials of $\mathsf{help}_1, \ldots, \mathsf{help}_l, P_{\vec{\phi_1},x_1}, \ldots, P_{\vec{\phi_q},x_q}, 1$ are all distinct, when $x_1, \ldots, x_q$ are distinct inputs. This means that the matrix whose columns correspond to monomials (ordered as specified above) and whose rows correspond to the polynomials $\mathsf{help}_1, \ldots, \mathsf{help}_l, P_{\vec{\phi_1},x_1}, \ldots, P_{\vec{\phi_q},x_q}, 1$ (ordered according to their leading monomial) is in echelon form. Hence, the latter polynomials are linearly independent. Finally, by combining Theorem 3 and the LIP theorem, we obtain the following theorem.

Theorem 4. *Let $\vec{\Omega}$, h and E be defined as above. Define F: $\mathbb{Z}_p^n \times \{0,1\}^n \to \mathbb{G}$ by $F(\vec{a}, x) = \mathsf{WNR}^{d\text{-}\mathsf{lin}}(\vec{a}, \mathsf{E}(h(x, \vec{\Omega}(\vec{a}))))$, for all $\mathbf{a} \in \mathbb{Z}_p^n$ and $x \in \{0,1\}^n$. Then we can reduce the $\Phi_{\mathfrak{S}_n,d}$-RKA-PRF security of F to the hardness of the $p_n d$-DDHI problem in $\mathbb{G}$ and the $p_n d$-SDL problem in $\mathbb{G}$, respectively with a loss of a factor $np_n d$ and of a factor n, and to the* CR *security of h. Moreover, the time overhead of this reduction is polynomial in n, d, p_n and in the number of queries made by the adversary attacking the $\Phi_{\mathfrak{S}_n,d}$-RKA-PRF security of F.*

6 Extension to PRFs in Symmetric Bilinear Groups

6.1 High-Level Overview of Existing Constructions and Challenges

All the previous constructions (of classical PRF and RKA-secure PRF) require at least DDH to hold. In particular, they are insecure if there exists a symmetric pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. In this section, we investigate how to adapt our linearly independent polynomials framework and the corresponding LIP theorem to handle constructions of PRFs under weaker assumptions, which may hold in symmetric bilinear groups.

The first algebraic PRF based on DLin is the Lewko-Waters PRF [26], which is defined as follows:

$$\mathsf{LW}(\vec{\boldsymbol{A}}, x) = \left[\prod_{i=1}^{n} \boldsymbol{A}_i^{x_i} \cdot \boldsymbol{A}'\right],$$

with $\vec{\boldsymbol{A}} = (\boldsymbol{A}_1, \ldots, \boldsymbol{A}_n)$ being a vector of n uniformly random matrices in $\mathbb{Z}_p^{2\times 2}$ and $\boldsymbol{A}'$ a uniformly random matrix in $\mathbb{Z}_p^{2\times m}$, for some $m \geq 1$. $\boldsymbol{A}'$ was actually in $\mathbb{Z}_p^{2\times 1}$ (i.e., $m = 1$) in [26] (with only the first group element being returned). This PRF is secure under DLin, and even under a weaker assumption, namely the $\mathcal{U}_2$-MDDH-assumption of Escala et al. [18]. In the latter paper, this PRF is extended to any MDDH-assumption, which particularly encompasses DDH and DLin. These instantiations differ by the size of the matrices and their distribution. Except for constructions using multilinear maps and lattices [5,16] or trivial variants, we are not aware of any other construction.

Commutation Challenge. From a high level point of view, these PRFs are very similar to the one considered in our algebraic framework in Sect. 3, except elements of keys are now matrices. Unfortunately, matrices do not commute in general, and this lack of commutativity makes everything more complex.

One naive solution would be to extend the LIP theorem by considering non-commutative polynomials, or in other words elements of the free algebra $\mathbb{Z}_p\langle T_1, \ldots, T_n\rangle$. In this algebra, for example, T_1T_2 and T_2T_1 are distinct and linearly independent elements. The problem is that, as proven by Amitsur and Levitzki [4], for any matrices $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_4 \in \mathbb{Z}_p^{2\times 2}$, $\sum_{\sigma\in\mathfrak{S}_4} \mathrm{sgn}(\sigma) \cdot \boldsymbol{A}_{\sigma(1)} \cdot \boldsymbol{A}_{\sigma(2)} \cdot \boldsymbol{A}_{\sigma(3)} \cdot \boldsymbol{A}_{\sigma(4)} = 0$, with $\mathrm{sgn}(\sigma)$ being the parity of the permutation σ. Thus, while the family of non-commutative polynomials $(P_\sigma = T_{\sigma(1)}T_{\sigma(2)}T_{\sigma(3)}T_{\sigma(4)})_{\sigma\in\mathcal{S}_4}$ is linearly independent in the free algebra, the PRF of domain $\mathcal{D} = \mathcal{S}_4$, the PRF defined by $F(\vec{\boldsymbol{A}}, \sigma) = \left[\boldsymbol{A}_{\sigma(1)}\boldsymbol{A}_{\sigma(2)}\boldsymbol{A}_{\sigma(3)}\boldsymbol{A}_{\sigma(4)}\boldsymbol{A}'\right]$ would clearly be insecure.

Assumption Challenge and Generic Symmetric Bilinear Group. The second challenge is to prove the hardness of the $\mathcal{E}_{2,d}$-MDDH assumption in the generic bilinear group, which is done in the full version, using a non-trivial technical lemma. Notably, contrary to the cyclic group case, it is not straightforward to check whether a PRF defined by $F(\vec{\boldsymbol{A}}, x) = \left[P_x(\vec{\boldsymbol{A}}) \cdot \boldsymbol{A}'\right]$ is secure in the generic bilinear group model, where $(P_x)_{x\in\mathcal{D}}$ is a family of non-commutative polynomials, $\vec{\boldsymbol{A}}$ is a vector of matrices from $\mathbb{Z}_p^{2\times 2}$, and $\boldsymbol{A}'$ is a matrix from $\mathbb{Z}_p^{2\times m}$, for some $m \geq 1$.

6.2 Generalized Polynomial Framework

Let us show how we address these challenges.

Generalized Polynomial (GP) Security. Let us introduce the (k, n, d)-GP security of a cyclic group $\mathbb{G} = \langle g \rangle$ as a generalization of the (n, d)-LIP security in Sect. 3.2, where the secret scalar $a' \xleftarrow{\$} \mathbb{Z}_p$ and the secret vector of scalars $\vec{a} \xleftarrow{\$} \mathbb{Z}_p^n$ are replaced by a secret matrix $\boldsymbol{A'} \xleftarrow{\$} \mathbb{Z}_p^{k \times m}$ (for some $m \geq 1$; for the sake of simplicity, in the sequel, we choose $k = m$) and a secret vector of matrices $\vec{\boldsymbol{A}} \xleftarrow{\$} (\mathbb{Z}_p^{k \times k})^n$, respectively.

Result under $\mathcal{E}_{2,d}$-MDDH. To extend Theorem 1 to symmetric bilinear groups and avoid the commutativity problem, we suppose that all indeterminates appear "in the same order when multiplied together" in each subexpression of the representation of the non-commutative polynomials P_j (e.g., $P_1 = T_1T_3 + T_3T_2$ and $P_2 = T_3 + T_1T_2$, where T_1 appears before T_3 which appears before T_2). The condition is quite natural and is formally defined in the full version. That makes these non-commutative polynomials behave very similarly to commutative polynomial, and we get the following theorem.

Theorem 5. *Let $\mathbb{G} = \langle g \rangle$ be a group of prime order p. Let $\mathscr{A}$ be an adversary against the $(2, n, d)$-GP security of $\mathbb{G}$ that makes q oracle queries $P_1, \ldots, P_q$. We suppose that all indeterminates appear in the same order in each monomial of each non-commutative polynomials P_j. Then we can build an adversary $\mathscr{B}$ against the $\mathcal{E}_{2,d}$-MDDH problem in $\mathbb{G}$, such that $\mathbf{Adv}_{\mathbb{G}}^{(2,n,d)\text{-gp}}(\mathscr{A}) \leq n \cdot d \cdot \mathbf{Adv}_{\mathbb{G}}^{\mathcal{E}_{2,d}\text{-mddh}}(\mathscr{B}) + O(ndq/p)$. The running time of $\mathscr{B}$ is that of $\mathscr{A}$ plus the time to perform a polynomial number (in q, n, and d) of operations in $\mathbb{Z}_p$ and $\mathbb{G}$.*

The proof is similar to the proof of the LIP theorem (with some additional care when partially evaluating polynomials to avoid having polynomials with matrix coefficients) and is given in the full version. Actually, this theorem can trivially be extended to the (k, n, d)-GP security and the $\mathcal{E}_{k,d}$-MDDH assumption. But for $k \geq 3$ and $n \geq 2$, it is not known if the latter assumption is secure in the generic k-linear group model.

Results in the Generic Bilinear Group Model. We may wonder whether the $(2, k, d)$-GP security still holds in the generic bilinear group model, when indeterminates do not necessarily appear in the same order in each polynomial P_j. As seen before, it is not sufficient to suppose that $(P_j)_{j=1,\ldots,q}$ is a linearly independent family. But we show here that under a relatively natural condition, the *DLM* (distinct leading monomial) condition, the $(2, k, d)$-GP security still holds.

To formally state our result, we need to introduce some notions, which are formally defined in the full version and which are informally described here. We consider a monomial order for $\mathbb{Z}_p[T_1, \ldots, T_n]$, which is a total order on monomials $T_1^{i_1} \cdots T_n^{i_n}$ compatible with multiplications and where 1 is the smallest monomial. We then define the commutative leading monomials of a non-commutative

Table 3. Summary of our results related to generalized polynomial security

Cyclic Group $\mathbb{G}$	**Symmetric Bilinear Group** (pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$)
$P_j \in \mathbb{Z}_p[T_1, \ldots, T_n]$ commutative polynomial $(a', a_1, \ldots, a_n) \xleftarrow{\$} \mathcal{K} = \mathbb{Z}_p^{n+1}$	$P_j \in \mathbb{Z}_p\langle T_1, \ldots, T_n\rangle$ non-commutative polynomial $(a', a_1, \ldots, a_n) \xleftarrow{\$} \mathcal{K} = (\mathbb{Z}_p^{2\times 2})^{n+1}$
In generic cyclic group: $(1, n, d)$-GP security $\Leftrightarrow (P_j)_j$ satisfies the DLM condition $\Leftrightarrow (P_j)_j$ is linearly independent (easy)	In generic bilinear group: $(2, n, d)$-GP security $\Leftarrow (P_j)_j$ satisfies the DLM condition (Theorem 6)
Under $\mathcal{E}_{1,d}$-MDDH: $(1, n, d)$-GP security $\Leftrightarrow$ same condition as above (Theorem 1, the LIP theorem)	Under $\mathcal{E}_{2,d}$-MDDH: $(2, n, d)$-GP security $\Leftarrow$ same condition as above + same order for indeterminates or equivalently, $(P_j)_j$ is linearly independent + smae order for indeterminates (Theorem 5)

polynomial as the monomials which are the highest for our monomial order, when considered as commutative monomials. There may be many commutative leading monomials for a given polynomial (for example $T_1T_2^2 + 5T_2T_1T_2$ has two commutative leading monomials: $T_1T_2^2$ and $T_2T_1T_2$). We say a polynomial has a unique commutative leading monomial if there is only one such monomial.

Finally, we say that a family of polynomials $(P_j)_j$ satisfies the DLM condition, if there exists a monomial order and an invertible matrix $\boldsymbol{M} \in \mathbb{Z}_p^{q\times q}$ such that $\boldsymbol{M} \cdot (P_j)_j$ is a vector of non-commutative polynomials with unique and distinct commutative leading monomials, where $(P_j)_j$ is the column vector of polynomials P_j.

Theorem 6. *Let $\mathbb{G} = \langle g\rangle$ be a group of prime order p. Let $\mathscr{A}$ be an adversary against the $(2, n, d)$-GP security of $\mathbb{G}$ that makes q oracle queries $P_1, \ldots, P_q$. We suppose that $(P_j)_j$ satisfies the DLM condition. Then, the advantage $\mathbf{Adv}_{\mathbb{G}}^{(2,n,d)\text{-gp}}(\mathscr{A})$ is negligible in the generic bilinear group model.*

The proof of Theorem 6 is given in the full version. We remark that, in the case of commutative polynomials (i.e., LIP theorem), the DLM condition is exactly the same as saying that the polynomials P_j are linearly independent (using the Gauss reduction). However, this is not the case with non-commutative polynomials (e.g., consider $P_1 = T_1T_2$ and $P_2 = T_2T_1$ which are linearly independent but which have the same leading monomial).

Summary. Table 3 provides a summary of all our results about GP security.

6.3 Applications

RKA-PRFs in Generic Bilinear Groups. The RKA-PRF for permutation of univariate polynomial functions based on WNR (Sect. 5.2) can easily be transformed into an RKA-secure PRF for symmetric bilinear groups for the same set of RKD functions. It is sufficient to change keys from $\vec{a} \xleftarrow{\$} \mathbb{Z}_p^n$ to $\vec{\boldsymbol{A}} \xleftarrow{\$} (\mathbb{Z}_p^{2\times 2})^n$. Indeed, the RKA framework extends to this case easily, and the polynomials family we considered verifies the DLM condition as non-commutative polynomials. Actually, our proof of their linear independence can be seen as exhibiting a monomial order (namely the graded lexicographic order) for which these polynomials have distinct leading monomials. In addition, their leading monomials are always unique even as non-commutative polynomials.

RKA-PRFs under $\mathcal{E}_{2,d}$-MDDH. Unfortunately, Theorem 5 does not apply to RKA-PRF for permutation, as permutation change the order of the indeterminates. However, it still easily enables to construct the first RKA-PRF for univariate polynomial functions, secure in symmetric bilinear groups, using the construction of Sect. 5.2 (or a slightly more efficient variant thereof in the full version). Again, the construction is straightforward and so is the proof.

Acknowledgments. This work was supported by the French ANR-10-SEGI-015 PRINCE Project, the *Direction Générale de l'Armement* (DGA), the CFM Foundation, and the European Research Council under the European Union's Seventh Framework Program (FP7/2007–2013 Grant Agreement 339563 – CryptoCloud).

References

1. Abdalla, M., Benhamouda, F., Passelègue, A., Paterson, K.G.: Related-key security for pseudorandom functions beyond the linear barrier. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 77–94. Springer, Heidelberg (2014)
2. Abdalla, M., Catalano, D., Fiore, D.: Verifiable random functions from identity-based key encapsulation. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 554–571. Springer, Heidelberg (2009)
3. Advanced Encryption Standard (AES). National Institute of Standards and Technology (NIST), FIPS PUB 197, U.S. Department of Commerce, November 2001
4. Amitsur, A.S., Levitzki, J.: Minimal identities for algebras. Proc. Am. Math. Soc. **1**(4), 449–463 (1950)
5. Banerjee, A., Peikert, C.: New and improved key-homomorphic pseudorandom functions. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 353–370. Springer, Heidelberg (2014)
6. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 719–737. Springer, Heidelberg (2012)
7. Bellare, M., Canetti, R., Krawczyk, H.: Pseudorandom functions revisited: The cascade construction and its concrete security. In: 37th FOCS, October 1996, pp. 514–523. IEEE Computer Society Press (1996)

8. Bellare, M., Cash, D.: Pseudorandom functions and permutations provably secure against related-key attacks. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 666–684. Springer, Heidelberg (2010)
9. Bellare, M., Kohno, T.: A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In: Biham, Eli (ed.) EUROCRYPT 2003. LNCS, vol. 2656. Springer, Heidelberg (2003)
10. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer, Heidelberg (2006)
11. Biham, E.: New types of cryptanalytic attacks using related keys. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 398–409. Springer, Heidelberg (1994)
12. Biham, E., Dunkelman, O., Keller, N.: Related-key boomerang and rectangle attacks. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 507–525. Springer, Heidelberg (2005)
13. Biryukov, A., Khovratovich, D., Nikolić, I.: Distinguisher and related-key attack on the full AES-256. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 231–249. Springer, Heidelberg (2009)
14. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
15. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
16. Boneh, D., Lewi, K., Montgomery, H., Raghunathan, A.: Key homomorphic PRFs and their applications. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 410–428. Springer, Heidelberg (2013)
17. Boneh, D., Montgomery, H.W., Raghunathan, A.: Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In: Al-Shaer, E., Keromytis, A.D., Shmatikov, V. (eds.) ACM CCS 10, October 2010, pp. 131–140. ACM Press (2010)
18. Escala, A., Herold, G., Kiltz, E., Ràfols, C., Villar, J.: An algebraic framework for diffie-hellman assumptions. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 129–147. Springer, Heidelberg (2013)
19. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. J. ACM **33**(4), 792–807 (1986)
20. Goyal, V., O'Neill, A., Rao, V.: Correlated-input secure hash functions. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 182–200. Springer, Heidelberg (2011)
21. Hofheinz, D., Kiltz, E.: Secure hybrid encryption from weakened key encapsulation. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 553–571. Springer, Heidelberg (2007)
22. Hohenberger, S., Waters, B.: Constructing Verifiable Random Functions with Large Input Spaces. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 656–672. Springer, Heidelberg (2010)
23. Kim, J.-S., Hong, S.H., Preneel, B.: Related-key rectangle attacks on reduced AES-192 and AES-256. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 225–241. Springer, Heidelberg (2007)
24. Knudsen, L.R.: Cryptanalysis of LOKI91. In: Zheng, Yuliang, Seberry, Jennifer (eds.) AUSCRYPT 1992. LNCS, vol. 718. Springer, Heidelberg (1993)
25. Lewi, K., Montgomery, H., Raghunathan, A.: Improved constructions of PRFs secure against related-key attacks. In: Boureanu, I., Owesarski, P., Vaudenay, S. (eds.) ACNS 2014. LNCS, vol. 8479, pp. 44–61. Springer, Heidelberg (2014)

26. Lewko, A.B., Waters, B.: Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In: Al-Shaer, E., Jha, S., Keromytis, A.D. (eds.) ACM CCS 09, November 2009, pp. 112–120. ACM Press (2009)
27. Naor, M., Reingold, O.: Number-theoretic constructions of efficient pseudo-random functions. In: 38th FOCS, October 1997, pp. 458–467. IEEE Computer Society Press (1997)
28. Naor, M., Reingold, O.: Synthesizers and their application to the parallel construction of pseudo-random functions. J. Comput. Syst. Sci. **58**(2), 336–375 (1999)
29. Shacham, H.: A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074 (2007). http://eprint.iacr.org/2007/074
30. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg (1997)