



**HAL**  
open science

# Algebraic XOR-RKA-Secure Pseudorandom Functions from Post-Zeroizing Multilinear Maps

Michel Abdalla, Fabrice Benhamouda, Alain Passelègue

► **To cite this version:**

Michel Abdalla, Fabrice Benhamouda, Alain Passelègue. Algebraic XOR-RKA-Secure Pseudorandom Functions from Post-Zeroizing Multilinear Maps. ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Dec 2019, Kobe, Japan. pp.386-412, 10.1007/978-3-030-34621-8\_14 . hal-02375594

**HAL Id: hal-02375594**


**<https://inria.hal.science/hal-02375594>**

Submitted on 12 Nov 2020

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Algebraic XOR-RKA-Secure Pseudorandom Functions from Post-Zeroizing Multilinear Maps

Michel Abdalla<sup>1,2</sup>, Fabrice Benhamouda<sup>3</sup>, and Alain Passelègue<sup>4</sup>

<sup>1</sup> DIENS, École normale supérieure, CNRS, PSL Research University, Paris, France  
[michel.abdalla@ens.fr](mailto:michel.abdalla@ens.fr)

<sup>2</sup> Inria, Paris, France

<sup>3</sup> Algorand Foundation, New York, NY, USA  
[fabrice.benhamouda@normalesup.org](mailto:fabrice.benhamouda@normalesup.org)

<sup>4</sup> Inria, ENS Lyon, Lyon, France  
[alain.passelegue@inria.fr](mailto:alain.passelegue@inria.fr)

**Abstract.** Due to the vast number of successful related-key attacks against existing block-ciphers, related-key security has become a common design goal for such primitives. In these attacks, the adversary is not only capable of seeing the output of a function on inputs of its choice, but also on related keys. At Crypto 2010, Bellare and Cash proposed the first construction of a pseudorandom function that could provably withstand such attacks based on standard assumptions. Their construction, as well as several others that appeared more recently, have in common the fact that they only consider linear or polynomial functions of the secret key over complex groups. In reality, however, most related-key attacks have a simpler form, such as the XOR of the key with a known value. To address this problem, we propose the first construction of RKA-secure pseudorandom function for XOR relations. Our construction relies on multilinear maps and, hence, can only be seen as a feasibility result. Nevertheless, we remark that it can be instantiated under two of the existing multilinear-map candidates since it does not reveal any encodings of zero. To achieve this goal, we rely on several techniques that were used in the context of program obfuscation, but we also introduce new ones to address challenges that are specific to the related-key-security setting.

**Keywords.** Pseudorandom Functions, Related-Key Security, Multilinear Maps, Post-Zeroizing Constructions.

## 1 Introduction

**Context.** Most of the security models used to prove the security of cryptographic schemes usually assume that an adversary has only a black-box access to the cryptosystem. In particular, the adversary has no information about the secret key, nor can it modify the latter. Unfortunately, it has been shown that this is not always true in practice. For instance, an adversary may learn information from physical measures such as the running time of the protocol or its energy consumption, or may also be able to inject faults in the cryptosystem. In the

specific case of fault attacks, in addition to possibly being able to learn partial information about the key, the adversary may be able to force the cryptosystem to run with a different, but related, secret key. Then, by observing the outcomes of the cryptosystem with this new related key, an adversary may be able to break it. Such an attack is called a related-key attack (RKA) and has often been used against concrete blockciphers [Knu93, Bih94, BDK05, BDK<sup>+</sup>10].

**Formalization of RKA security.** Following the seminal cryptanalytic works by Biham and Knudsen, theoreticians have defined new security models in order to capture such attacks. In 2003, Bellare and Kohno formalized the foundations for RKA security [BK03]. Specifically, let  $F: \mathcal{K} \times \mathcal{D} \rightarrow \mathcal{R}$  be a pseudorandom function and let  $\Phi \subseteq \text{Fun}(\mathcal{K}, \mathcal{K})$  be a set of functions on the key space  $\mathcal{K}$ , called a class of related-key deriving (RKD) functions. We say that  $F$  is  $\Phi$ -RKA secure if it is hard to distinguish an oracle which, on input a pair  $(\phi, x) \in \Phi \times \mathcal{D}$ , outputs  $F(\phi(K), x)$ , from an oracle which, on the same input pair, outputs  $G(\phi(K), x)$ , where  $K \in \mathcal{K}$  is a random target key and  $G: \mathcal{K} \times \mathcal{D} \rightarrow \mathcal{R}$  is a random function.

**Existing Constructions.** Building provably RKA-secure pseudorandom functions in non-idealized model has been a long-standing open question until the work of Bellare and Cash in 2010 [BC10]. Their construction is adapted from the Naor-Reingold PRF [NR97] and is obtained by applying a generic framework to it. This generic framework has led to other constructions based under different assumptions [LMR14] and been recently extended to circumvent its limitations [ABPP14, ABP15, ABPP18]. However, despite being simple and elegant, the existing frameworks crucially rely on the algebraic structure of the pseudorandom functions and thus only allow to build RKA-secure pseudorandom functions for algebraic classes of RKD functions. More precisely, existing constructions use a key whose components are elements in  $\mathbb{Z}_p$ , and are proven secure against an attacker that is given the capability to perform operations (additions, multiplications, or even polynomial evaluation) modulo  $p$ , with  $p$  super-polynomial in the security parameter for instantiations over cyclic or multilinear groups [BC10, LMR14, ABPP14, ABP15], or  $p$  polynomial but still much larger than 2, for instantiations based on lattices [LMR14, BP14].

Unfortunately, the algebraic classes of RKD functions above are not very natural as they seem difficult to implement from the perspective of an attacker<sup>5</sup>. Moreover, they also do not seem to match attacks against concrete blockciphers such as AES (e.g., [BDK05, BDK<sup>+</sup>10]). To address these shortcomings, we focus in this paper on the XOR class of RKD functions, which seems more relevant for practice, as suggested by Bellare and Kohno [BK03]. In this class, which corresponds to the class of functions  $\Phi_{\oplus} = \{\phi_s: K \in \{0, 1\}^k \mapsto K \oplus s \in \{0, 1\}^k \mid s \in \{0, 1\}^k\}$  with  $\mathcal{K} = \{0, 1\}^k$  being the keyspace, the adversary is allowed to flip bits of the secret key.

<sup>5</sup> In practice, laser fault attacks can be used to switch bits inside a chip, but it seems unlikely that such a fault attack can be used to apply an operation modulo some prime number on a register.

In the context of pseudorandom functions, there have been a few proposals for protecting against weak versions of XOR-related-key attacks. In [AW14], for instance, Applebaum and Widder proposed several schemes that can provably resist XOR attacks by restricting the capabilities of the adversary. In particular, they consider models where the adversary only uses a bounded number of related keys or where it only performs random XOR operations. In [JW15], Jafargholi and Wichs proposed constructions of pseudorandom functions based on continuous non-malleable codes that can resist several forms of related-key attacks, including XOR-related-key attacks. Their solutions, however, only guarantee pseudorandomness under the original key and not under both the original and related keys as in standard notions of related-key security. Another advantage of our construction compared to theirs is that the key in our case is just a bit string.

In the random-oracle model, there is a straightforward construction of a pseudorandom function secure against XOR-related-key attacks:  $F(K, x) = H(K \| x)$ , with  $H$  being a hash function modeled as a random oracle, and  $\|$  being the string concatenation operator. However, from a theoretical perspective, having a construction in the random-oracle model does not provide any guarantee that the primitive can be realized under non-interactive (and falsifiable) assumptions. To the best of our knowledge, building a pseudorandom function that provably resists XOR-related-key attacks in non-idealized models still remains a major open problem. Furthermore, we would like to point out that we do not know of any construction of a pseudorandom function secure against XOR-related-key attacks in the generic multilinear group model.

**Our Contributions.** In this paper, we provide the first provably secure construction of an RKA-secure pseudorandom function for XOR relations. To achieve this goal, our construction departs significantly from previous RKA-secure pseudorandom functions in that the secret key no longer lies in an algebraic group. As in prior constructions (e.g., [BC10, ABPP14, ABP15, LMR14]), our new scheme also requires public parameters, such as generators of a cyclic group, which cannot be tampered with. However, unlike these constructions, the secret key in our scheme is just a bit string, whose values are used to select a subset of the common parameters which will be used in the evaluation of the pseudorandom function.<sup>6</sup> The evaluation is performed using an asymmetric multilinear map [GGH13a, CLT13, GGH15].

In particular, we prove our construction under two non-interactive assumptions. We further prove that it is hard to come up with encodings of zero (in the generic multilinear map model) given only the encodings revealed by these assumptions. Therefore these assumptions are plausible under some current instantiations of multilinear maps. To the best of our knowledge, this is the first construction using multilinear maps with such a level of security. In [BMSZ16], Badrinarayanan et al. were the first to take into account zeroizing attacks (i.e., attacks using encodings

<sup>6</sup> Please note that these common parameters are public and can therefore be stored in a non-secure read-only part of the memory which can potentially more easily be made tamper-proof.

of zero) against current multilinear maps constructions into their constructions and to propose a scheme which is proven not to reveal encodings of zero, in the generic multilinear map model. But, contrary to us, they did not provide a proof of their scheme under a non-interactive assumption.

**Overview of our techniques.** As mentioned above, our construction is based on asymmetric multilinear maps [GGH13a, CLT13, GGH15]. Informally speaking, a multilinear map is a generalization of bilinear maps. It allows to “encode” scalars  $a$  (in some finite field  $\mathbb{Z}_p$ ) into some encodings  $[a]_{\mathcal{S}}$  with respect to an index set (also called an index)  $\mathcal{S} \subseteq \mathcal{U}$ , where  $\mathcal{S}$  indicates the level of the encoding  $[a]_{\mathcal{S}}$  and  $\mathcal{U}$  denotes the top-level index set. We can add two elements  $[a]_{\mathcal{S}}$  and  $[b]_{\mathcal{S}}$  belonging to the same index set  $\mathcal{S}$  to obtain  $[a + b]_{\mathcal{S}}$ . We can also multiply elements  $[a]_{\mathcal{S}_1}$  and  $[b]_{\mathcal{S}_2}$  to compute  $[a \cdot b]_{\mathcal{S}_1 \cup \mathcal{S}_2}$  to obtain the encoding of  $a \cdot b$  with respect to the index set  $\mathcal{S}_1 \cup \mathcal{S}_2$ , as long as the two index sets are disjoint ( $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$ ). Finally, it is possible to test whether an element at level  $\mathcal{U}$  is an encoding of 0.

Let  $k$  and  $n$  be the lengths of the secret key  $K$  and input  $x$ , respectively, and let  $\mathcal{U} = \{1, \dots, k + n\}$ . Next, let  $\{a_{i,b}\}_{i \in [k], b \in \{0,1\}}$  and  $\{c_{j,b}\}_{j \in [n], b \in \{0,1\}}$  be random scalars in  $\mathbb{Z}_p$  and let  $\hat{a}_{i,b} = [a_{i,b}]_{\{i\}}$  be the encoding of  $a_{i,b}$  at index level  $\{i\}$  and  $\hat{c}_{j,b} = [c_{j,b}]_{\{j+k\}}$  be the encoding of  $c_{j,b}$  at index level  $\{j+k\}$ . The starting point of our construction is the function

$$F_{\mathbf{pp}}(K, x) = \left[ \prod_{i=1}^k a_{i,K_i} \prod_{j=1}^n c_{j,x_j} \right]_{\mathcal{U}},$$

where the public parameters  $\mathbf{pp}$  include  $\{\hat{a}_{i,b}\}_{i \in [k], b \in \{0,1\}}$  and  $\{\hat{c}_{j,b}\}_{j \in [n], b \in \{0,1\}}$  as well as the public parameters of the multilinear map.

Since the encodings of the scalars  $a_{i,b}$  and  $c_{j,b}$  are included in  $\mathbf{pp}$ , it is not hard to see that the user in possession of the secret  $K$  can efficiently evaluate this function at any point  $x$  by computing the multilinear map function with the help of the encodings  $\hat{a}_{i,K_i}$  and  $\hat{c}_{j,x_j}$ . To prove it secure and be able to instantiate the scheme with existing multilinear map candidates, however, is not straightforward as one needs to show that the adversary cannot mix too many existing encodings and create an encoding of zero.

While a proof in the generic multilinear map model [Sho97, GGH<sup>+</sup>13b, BR14, BGK<sup>+</sup>14, Zim15] is possible, we would like to rely on non-interactive complexity assumptions over multilinear maps which are likely to hold in existing multilinear map constructions. To achieve this goal, we change the way in which the index sets are defined using techniques from program obfuscation [GGH<sup>+</sup>13b, BGK<sup>+</sup>14, Zim15]. More precisely, we make use of the notion of strong straddling sets [BGK<sup>+</sup>14, MSW14, Zim15, BMSZ16], which informally allows to partition a set into two disjoint sets of subsets so that they cannot be mixed. As in [MSW14, Zim15, BMSZ16], we first construct strong straddling set systems over sets  $\mathcal{S}_i$  of  $k^2$  fresh symbols, for  $i \in \{0, \dots, k\}$  and use  $\mathcal{S}_i$  for  $i \geq 1$  to prevent the adversary from mixing an exponential number of inputs. In addition to that, and unlike [MSW14, Zim15, BMSZ16], we use  $\mathcal{S}_0$  in the proof to prevent the

adversary from mixing an internal private representation of the key (used by the reduction) with the parameters.

As we show in Section 4, the resulting scheme can be proven secure in the generic multilinear map model. In particular, we are actually able to prove that no polynomial-time adversary can construct a (non-trivial) encoding of zero.<sup>7</sup> In addition to that, as we show in Section 5, one benefit of using (strong) straddling sets is that it allows us to prove the security of our construction under non-interactive assumptions and avoid the use of idealized models.<sup>8</sup> Finally, we also prove the plausibility of these new assumptions in the full version [ABP17] by showing that they hold in the generic multilinear map model.

We would like to stress that the security proof in Section 5 is a much stronger result qualitatively than a direct proof in the generic multilinear map model since the latter model is only used to show that our (non-interactive) assumptions are plausible.

**Concrete Instantiations.** The security of our scheme relies on two new non-interactive assumptions on multilinear maps. However, contrary to most classical assumptions (such as the Decisional Diffie-Hellman assumption for low-level group elements) or the multilinear subgroup assumption used to construct witness encryption and indistinguishable obfuscation in [GLW14, GLSW15], our new assumptions do not reveal any encoding of zero. More precisely, similarly to what Badrinarayanan et al. did in [BMSZ16], we show in the ideal multilinear map model, that given an instance for one of our assumptions, the adversary cannot construct any encoding of zero (even at the top level). In particular, this implies that our assumptions holds in the hybrid graded encoding model [GMM<sup>+</sup>16], as this model just restricts the shape of the encodings of zero that the adversary can construct, while we prove that the adversary cannot construct any such encoding.

Our assumptions are therefore not broken by most of the attacks against multilinear maps [GGH13a, CHL<sup>+</sup>15, CGH<sup>+</sup>15, HJ16, CLLT16] including the annihilation attacks [MSZ16, CGH17] and recent attacks such as [CLLT17, CVW18]. Hence, while the GGH13 and CLT15 multilinear map candidates [GGH13a, CLT15] might not be used for our construction because of recent attacks without encodings of zero [CFL<sup>+</sup>16, ABD16, CJL16], our assumptions are plausible when implemented with the GGH15<sup>9</sup> or the CLT13 multilinear map candidates [GGH15, CLT13].

<sup>7</sup> Note that, to prove that two games are indistinguishable in the generic multilinear map model, it suffices to prove that the adversary cannot generate encodings of zeros in either game as it would only obtain random handles which carry no information.

<sup>8</sup> The security result in Section 4 is therefore implied by the security result in Section 5. Section 4 should be seen as a warm-up for Section 5.

<sup>9</sup> The GGH15 multilinear map is defined over a graph. Index sets (and in particular straddling set indexes) are defined independently of the graph structure of GGH15, hence they can be implemented as on GGH13. Specifically, index sets are based on the  $z_i$ 's, as for GGH13, and not on the graph structure. More precisely, the construction relies on distinct  $z_i$ 's for every index and an encoding with index set  $S$  depends on  $\prod_{i \in S} z_i$ , exactly as on GGH13. This is completely independent of the graph structure and then we can use a simple chain as a graph, as in the case of indistinguishable

**Additional Related Work.** In addition to the work mentioned above, a few other constructions of pseudorandom functions against related-key attacks for linear and polynomial functions have been proposed in [BLMR13, LMR14]. While their RKA-secure pseudorandom functions also require multilinear maps, their security proofs are based on assumptions which reveal encodings of zero and hence are subject to the multiple attacks mentioned above. Moreover, these works do not consider XOR-related-key attacks.

Related-key security with respect to XOR relations has also been considered in the context of Even-Mansour ciphers [CS15, FP15, Men16, WLZZ16]. Unlike our work, which aims to prove security under well specified non-interactive complexity assumptions, all these works rely on idealized models, which we want to avoid.

In [FX15], the authors proposed efficient constructions of identity-based encryption and key encapsulation schemes that remain secure against related-key attacks for a large class of functions, which include XOR relations. While their results are interesting, we remark that achieving RKA security for randomized primitives appears to be significantly easier than for deterministic ones, as already noted by Bellare and Cash [BC10].

Finally, it is worth mentioning here that, due to the results of Bellare, Cash, and Miller [BCM11], RKA security for pseudorandom functions can be transferred to several other primitives, including identity-based encryption, signatures, and chosen-ciphertext-secure public-key encryption.

**Organization.** The rest of the paper is organized as follows. Section 2 presents standard definitions of (related-key) pseudorandom functions and multilinear maps. It also introduces the generic multilinear map model and the notion of straddling set system. Section 3 explains our construction of XOR-RKA PRF. Section 4 provides a first security proof of our new scheme in the generic multilinear map model. It also shows that is not feasible for an adversary to generate non-trivial encodings of zero. Finally, Section 5 describes our main result, which is a proof of security for our new construction under two new non-interactive assumptions. The formal proof of security, as well as the proofs of our assumptions are detailed in the appendix. In particular, we prove that our assumptions are not only secure in the generic multilinear map model, but also that it is not feasible for an adversary to generate (non-trivial) encodings of zero. Hence, our assumptions are plausible given some current instantiations of multilinear maps.

---

obfuscation candidate. A clean exposition of how to implement straddling sets for GGH15 is given in [Hal15, Sec. 4.2]. Note also that we consider multilinear maps with plaintext space corresponding to a prime-order finite field. This can be obtained using [Hal15, Sec. 4.1]. Attacks such as [CHKL18] do not apply in this setting.

## 2 Definitions

### 2.1 Notation and Games

**Notation.** We denote by  $\kappa$  the security parameter. Let  $F: \mathcal{K} \times \mathcal{D} \rightarrow \mathcal{R}$  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  $F: \mathcal{K} \times \mathcal{D} \rightarrow \mathcal{R}$  is then denoted by  $\text{Fun}(\mathcal{K}, \mathcal{D}, \mathcal{R})$ . Likewise,  $\text{Fun}(\mathcal{D}, \mathcal{R})$  denotes the set of all functions mapping  $\mathcal{D}$  to  $\mathcal{R}$ . If  $S$  is a set, then 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  $\vec{x} = (x_1, \dots, x_{|\vec{x}|})$ . For a binary string  $x$ , we denote its length by  $|x|$ ,  $x_i$  its  $i$ -th bit, so  $x \in \{0, 1\}^{|x|}$  and  $x = x_1 \parallel \dots \parallel x_n$ .

**Games [BR06].** 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  $\mathcal{A}$ , we proceed as follows. First, **Initialize** is executed and its outputs become the input of  $\mathcal{A}$ . When  $\mathcal{A}$  executes, its oracle queries are answered by the corresponding procedures of  $G$ . When  $\mathcal{A}$  terminates, its outputs become the input of **Finalize**. The output of the latter, denoted  $G^{\mathcal{A}}$  is called the output of the game, and we let “ $G^{\mathcal{A}} \Rightarrow 1$ ” 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.

### 2.2 Pseudorandom Functions

Our definitions of pseudorandom functions and related-key secure pseudorandom functions include a **Setup** algorithm that is used to generate public parameters. For classical PRFs, the public parameters could actually just be included in the key. However, to our knowledge, all the known proven RKA-secure PRFs [BC10, ABPP14, LMR14, ABP15], use public parameters. Contrary to the key itself, the public parameters cannot be modified by the related-key deriving function. In our case, the **Setup** algorithm is specified explicitly in the construction for clarity.

**PRFs [GGM86, BC10].** Consider a pseudorandom function  $F_{\text{pp}}: \mathcal{K} \times \mathcal{D} \rightarrow \mathcal{R}$  with public parameters  $\text{pp}$ . The advantage of an adversary  $\mathcal{A}$  in attacking the standard PRF security of a function  $F_{\text{pp}}$  is defined via

$$\text{Adv}_{F_{\text{pp}}}^{\text{prf}}(\mathcal{A}) = \Pr \left[ \text{PRFReal}_{F_{\text{pp}}}^{\mathcal{A}} \Rightarrow 1 \right] - \Pr \left[ \text{PRFRand}_{F_{\text{pp}}}^{\mathcal{A}} \Rightarrow 1 \right].$$

Game  $\text{PRFReal}_{F_{\text{pp}}}$  first runs the **Setup** algorithm to generate the public parameters  $\text{pp}$  which it outputs. It then picks  $K \xleftarrow{\$} \mathcal{K}$  at random, and responds to oracle query  $\mathbf{Fn}(x)$  via  $F_{\text{pp}}(K, x)$ . Game  $\text{PRFRand}_{F_{\text{pp}}}$  runs **Setup** and outputs the public parameters  $\text{pp}$ . It then picks  $f \xleftarrow{\$} \text{Fun}(\mathcal{D}, \mathcal{R})$  and responds to oracle query  $\mathbf{Fn}(x)$  via  $f(x)$ .



**RKA-PRFs [BK03, BC10].** Consider a pseudorandom function  $F_{\text{pp}}: \mathcal{K} \times \mathcal{D} \rightarrow \mathcal{R}$  with public parameters  $\text{pp}$ . Let  $\Phi \subseteq \text{Fun}(\mathcal{K}, \mathcal{K})$ ; the members of  $\Phi$  are called RKD (Related-Key Deriving) functions. An adversary is said to be  $\Phi$ -restricted if its oracle queries  $(\phi, x)$  satisfy  $\phi \in \Phi$ . The advantage of a  $\Phi$ -restricted adversary  $\mathcal{A}$  in attacking the RKA-PRF security of  $F_{\text{pp}}$  is defined via

$$\text{Adv}_{\Phi, F_{\text{pp}}}^{\text{prf-rka}}(\mathcal{A}) = \Pr \left[ \text{RKPRFReal}_{F_{\text{pp}}}^{\mathcal{A}} \Rightarrow 1 \right] - \Pr \left[ \text{RKPRFRand}_{F_{\text{pp}}}^{\mathcal{A}} \Rightarrow 1 \right].$$

Game  $\text{RKPRFReal}_{F_{\text{pp}}}$  first runs the **Setup** algorithm to generate the public parameters which it outputs. It then picks a key  $K \xleftarrow{\$} \mathcal{K}$  at random, and responds to oracle query  $\text{RKFn}(\phi, x)$  via  $F_{\text{pp}}(\phi(K), x)$ . Game  $\text{RKPRFRand}_{F_{\text{pp}}}$  runs **Setup** to generate the public parameters  $\text{pp}$  and outputs them. It then picks  $G \xleftarrow{\$} \text{Fun}(\mathcal{K}, \mathcal{D}, \mathcal{R})$  and  $K \xleftarrow{\$} \mathcal{K}$  at random, and responds to oracle query  $\text{RKFn}(\phi, x)$  via  $G(\phi(K), x)$ . We say that  $F_{\text{pp}}$  is a  $\Phi$ -RKA-secure PRF if for any  $\Phi$ -restricted adversary, its advantage in attacking the RKA-PRF security is negligible.

**XOR-RKA-PRFs.** Let  $F_{\text{pp}}: \mathcal{K} \times \mathcal{D} \rightarrow \mathcal{R}$  be a pseudorandom function with public parameters  $\text{pp}$  and  $\mathcal{K} = \{0, 1\}^k$  for some integer  $k \geq \kappa$ . We say that  $F_{\text{pp}}$  is XOR-RKA-secure if it is a  $\Phi_{\oplus}$ -RKA-secure PRF according to the above definition, where  $\Phi_{\oplus} = \{\phi_s: K \in \{0, 1\}^k \mapsto K \oplus s \in \{0, 1\}^k \mid s \in \{0, 1\}^k\}$ .

### 2.3 Multilinear Maps

We informally introduced multilinear maps in the introduction. Let us now introduce formal definitions, following the notations of [Zim15].

**Definition 1 (Formal Symbol).** A formal symbol is a bitstring in  $\{0, 1\}^*$ . Distinct variables denote distinct bitstrings, and we call a fresh formal symbol any bitstring in  $\{0, 1\}^*$  that has not already been assigned to a formal symbol.

**Definition 2 (Index Sets).** An index set (also called index) is a set of formal symbols.

**Definition 3 (Multilinear Map).** A multilinear map is defined by a tuple of six algorithms ( $\text{MM.Setup}$ ,  $\text{MM.Encode}$ ,  $\text{MM.Add}$ ,  $\text{MM.Mult}$ ,  $\text{MM.ZeroTest}$ ,  $\text{MM.Extract}$ ) with the following properties:

- $\text{MM.Setup}$  takes as inputs the security parameter  $\kappa$  in unary and an index set  $\mathcal{U}$ , termed the top-level index set, and generates public parameters  $\text{mm.pp}$ , secret parameters  $\text{mm.sp}$ , and a prime number  $p$ ;
- $\text{MM.Encode}$  takes as inputs secret parameters  $\text{mm.sp}$ , a scalar  $x \in \mathbb{Z}_p$ , and an index set  $\mathcal{S} \subseteq \mathcal{U}$  and outputs:

$$\text{MM.Encode}(\text{mm.sp}, x, \mathcal{S}) \rightarrow [x]_{\mathcal{S}} ;$$

For the index set  $\mathcal{S} = \emptyset$ ,  $[x]_{\emptyset}$  is simply the scalar  $x \in \mathbb{Z}_p$ .

- **MM.Add** takes as inputs public parameters  $\text{mm.pp}$  and two encodings with same index set  $\mathcal{S} \subseteq \mathcal{U}$  and outputs:

$$\text{MM.Add}(\text{mm.pp}, [x]_{\mathcal{S}}, [y]_{\mathcal{S}}) \rightarrow [x + y]_{\mathcal{S}} ;$$

- **MM.Mult** takes as inputs public parameters  $\text{mm.pp}$  and two encodings with index sets  $\mathcal{S}_1, \mathcal{S}_2 \subseteq \mathcal{U}$  respectively and outputs:

$$\text{MM.Mult}(\text{mm.pp}, [x]_{\mathcal{S}_1}, [y]_{\mathcal{S}_2}) \rightarrow \begin{cases} [xy]_{\mathcal{S}_1 \cup \mathcal{S}_2} & \text{if } \mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset ; \\ \perp & \text{otherwise} \end{cases} ;$$

- **MM.ZeroTest** takes as inputs public parameters  $\text{mm.pp}$  and a top-level encoding (with index set  $\mathcal{U}$ ) and outputs:

$$\text{MM.ZeroTest}(\text{mm.pp}, [x]_{\mathcal{S}}) \rightarrow \begin{cases} \text{“zero”} & \text{if } \mathcal{S} = \mathcal{U} \text{ and } x = 0 ; \\ \text{“non-zero”} & \text{otherwise} \end{cases} ;$$

- **MM.Extract** takes public parameters  $\text{mm.pp}$  and a top-level encoding  $[x]_{\mathcal{U}}$  as inputs and outputs a canonical representation of  $[x]_{\mathcal{U}}$ .

*Remark 4.* The **MM.Extract** algorithm is needed for our pseudorandom function to be deterministic with all currently known instantiations of multilinear maps [GGH13a, CLT13, GGH15, CLT15]. Indeed, in these instantiations, the same group element has many different representations, and the extraction procedure enables to extract a unique representation from any top-level group element (i.e., of index  $\mathcal{U}$ ).

This extraction is necessary for our proof under non-interactive assumptions in Section 5 to work. For our proof in the generic multilinear map model, this is not required. For this reason, our generic multilinear map model does not support extraction for the sake of simplicity. Actually, this only strengthens the result, as before extraction, the adversary still has to possibility to add top-level group elements while extracted values are not necessarily homomorphic.

**Conventions.** In order to ease the reading, we adopt the following conventions in the rest of the paper:

- Scalars are noted with lowercase letter, e.g.  $a, b, \dots$
- Encodings are noted either as their encoding at index set  $\mathcal{S}$ ,  $[a]_{\mathcal{S}}$  or simply with a hat, when the index set is clear from the context, e.g.  $\hat{a}, \hat{b}, \dots$ . In particular,  $\hat{a}$  is an encoding of the scalar  $a$ .
- Index sets as well as formal variables are noted with uppercase letters, e.g.  $X, \mathcal{S}, \mathcal{S}', \dots$
- We denote by  $S_1 \cdot S_2$  or  $S_1 S_2$  the union of sets  $S_1$  and  $S_2$ . This notation implicitly assumes that the two sets are disjoint. If  $S_1$  is an element, then  $S_1 \cdot S_2$  stands for  $\{S_1\} \cdot S_2$ .
- The top-level index set is referred as  $\mathcal{U}$ .

We also naturally extend these notations when clear from the context, so for instance  $\hat{a} + \hat{b} = \text{MM.Add}(\text{mm.pp}, \hat{a}, \hat{b})$  and  $\hat{a} \cdot \hat{b} = \text{MM.Mult}(\text{mm.pp}, \hat{a}, \hat{b})$ .

## 2.4 Generic Multilinear Map Model

Our construction is proven secure under two non-interactive assumptions. One is very classical and is a variant of DDH. The other is relatively simple but is not classical and we prove it in the generic multilinear map model to show its plausibility. That is why we need to introduce the generic multilinear map model, in addition to the fact that we also prove in this model that our construction does not enable the adversary to produce encodings of zero.

The generic multilinear map model is similar to the generic group model [Sho97]. Roughly speaking, the adversary has only the capability to apply operations (add, multiply, and zero-test) of the multilinear map to encodings. A scheme is secure in the generic multilinear map model if for any adversary breaking the real scheme, there is a generic adversary that breaks a modified scheme in which encodings are replaced by fresh nonces, called *handles*, that it can supply to a stateful oracle  $\mathcal{M}$ , defined as follows:

**Definition 5 (Generic Multilinear Map Oracle).** *A generic multilinear map oracle is a stateful oracle  $\mathcal{M}$  that responds to queries as follows:*

- On a query  $\text{MM.Setup}(1^\kappa, \mathcal{U})$ ,  $\mathcal{M}$  generates a prime number  $p$  and parameters  $\text{mm.pp}, \text{mm.sp}$  as fresh nonces chosen uniformly at random from  $\{0, 1\}^\kappa$ . It also initializes an internal table  $T \leftarrow []$  that it uses to store queries and handles. It finally returns  $(\text{mm.pp}, \text{mm.sp}, p)$  and set internal state so that subsequent  $\text{MM.Setup}$  queries fail.
- On a query  $\text{MM.Encode}(z, x, \mathcal{S})$ , with  $z \in \{0, 1\}^\kappa$  and  $x \in \mathbb{Z}_p$ , it checks that  $z = \text{mm.sp}$  and  $\mathcal{S} \subseteq \mathcal{U}$  and outputs  $\perp$  if the check fails, otherwise it generates a fresh handle  $h \xleftarrow{\$} \{0, 1\}^\kappa$ , adds  $h \mapsto (x, \mathcal{S})$  to  $T$ , and returns  $h$ .
- On a query  $\text{MM.Add}(z, h_1, h_2)$ , with  $z, h_1, h_2 \in \{0, 1\}^\kappa$ , it checks that  $z = \text{mm.pp}$ , that  $h_1$  and  $h_2$  are handles in  $T$  which are mapped to values  $(x_1, \mathcal{S}_1)$  and  $(x_2, \mathcal{S}_2)$  such that  $\mathcal{S}_1 = \mathcal{S}_2 = \mathcal{S} \subseteq \mathcal{U}$ , and returns  $\perp$  if the check fails. If it passes, it generates a fresh handle  $h \xleftarrow{\$} \{0, 1\}^\kappa$ , adds  $h \mapsto (x_1 + x_2, \mathcal{S})$  to  $T$ , and returns  $h$ .
- On a query  $\text{MM.Mult}(z, h_1, h_2)$ , with  $z, h_1, h_2 \in \{0, 1\}^\kappa$ , it checks  $z = \text{mm.pp}$ , that  $h_1$  and  $h_2$  are handles in  $T$  which are mapped to values  $(x_1, \mathcal{S}_1)$  and  $(x_2, \mathcal{S}_2)$  such that  $\mathcal{S}_1 \cup \mathcal{S}_2 \subseteq \mathcal{U}$  and  $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$ , and returns  $\perp$  if the check fails. If it passes, it generates a fresh handle  $h \xleftarrow{\$} \{0, 1\}^\kappa$ , adds  $h \mapsto (x_1 x_2, \mathcal{S}_1 \cup \mathcal{S}_2)$  to  $T$ , and returns  $h$ .
- On a query  $\text{MM.ZeroTest}(z, h)$ , with  $z, h \in \{0, 1\}^\kappa$ , it checks  $z = \text{mm.pp}$ , that  $h$  is a handle in  $T$  such that it is mapped to a value  $(x, \mathcal{U})$ , and returns  $\perp$  if the check fails. If it passes, it returns “zero” if  $x = 0$  and “non-zero” otherwise.

*Remark 6.* In order to ease the reading, we actually use a slightly different and more intuitive characterization of the generic multilinear map oracle in our proofs. Informally, instead of considering encodings as nonces, we consider these as formal polynomials (that can be computed easily), whose formal variables are substituted with their join value distribution from the real game. In our construction, formal

variables are  $\hat{a}_{i,b}, \hat{c}_{j,b}, \hat{z}_{i_1, i_2, b_1, b_2}$  —please refer to the construction in Section 3 for details. This variant characterization follows the formalization from [Zim15, Appendix B], please refer to this section for more formal definitions.

## 2.5 Actual Instantiations

While Definition 3 is a very natural definition and is what we actually would like as a multilinear map, up to now, we still do not know any such construction. Known constructions [GGH13a, CLT13, GGH15, CLT15] of multilinear maps are actually “noisy” variants of our formal definition. That is, each encoding includes a random error term, and similarly to what happens for lattice-based constructions, this error term grows when performing operations (addition or multiplication). Eventually, this error term becomes too big and the `MM.ZeroTest` can no longer recover the correct answer. This noise implicitly restricts the number of operations that can be performed. Intuitively, in current constructions, the errors are added when performing an addition and multiplied when performing a multiplication. However, the fact that current instantiations are noisy does not pose any problem regarding our construction, as the number of operations for evaluating our pseudorandom function is fixed and independent from the instantiation of the multilinear map.

## 2.6 Straddling Sets

Our construction and its proofs use strong straddling sets [BGK<sup>+</sup>14, MSW14, Zim15, BMSZ16], in order to prevent the adversary from mixing too many encodings and creating encodings of zero. We recall their definition below. We first recall that, for a set  $\mathcal{S}$ , we say that  $\{S_1, \dots, S_k\}$ , for some integer  $k$ , is a partition of  $\mathcal{S}$ , if and only if  $\cup_{i=1}^k S_i = \mathcal{S}$ ,  $S_i \neq \emptyset$  and  $S_i \cap S_j = \emptyset$ , for any  $1 \leq i, j \leq k$ ,  $i \neq j$ .

**Definition 7 ((Strong) Straddling Set System).** *For  $k \in \mathbb{N}$ , a  $k$ -straddling set system over a set  $\mathcal{S}$  consists of two partitions  $S_0 = \{S_{0,1}, \dots, S_{0,k}\}$  and  $S_1 = \{S_{1,1}, \dots, S_{1,k}\}$  of  $\mathcal{S}$  such that the following holds: for any  $\mathcal{T} \subseteq \mathcal{S}$ , if  $T_0, T_1$  are distinct subsequences of  $S_{0,1}, \dots, S_{0,k}, S_{1,1}, \dots, S_{1,k}$  such that  $T_0$  and  $T_1$  are partitions of  $\mathcal{T}$ , then  $\mathcal{T} = \mathcal{S}$  and  $T_0 = S_b$  and  $T_1 = S_{1-b}$  for some  $b \in \{0, 1\}$ .*

*Moreover, we say that  $S_0, S_1$  is a strong  $k$ -straddling set system if for any  $1 \leq i, j \leq k$ ,  $S_{0,i} \cap S_{1,j} \neq \emptyset$ .*

A strong  $k$ -straddling set system is clearly also a  $k$ -straddling set system. Intuitively, a  $k$ -straddling set system ensures that the only two solutions to build a partition of  $\mathcal{S}$  from combining sets in  $S_0$  or  $S_1$  are to use either every element in  $S_0$  or every element in  $S_1$ .

We are only using strong straddling set systems in this paper, for the sake of simplicity. However, we only rely on the straddling set property in all our proofs, except the proof of one of our non-interactive assumptions, namely the `Sel-Prod`

assumption. Let us now recall the construction of strong straddling set systems from [MSW14]. For the sake of completeness, we also recall the construction of straddling set systems from [BGK<sup>+</sup>14] in the full version [ABP17].

**Constructions 8 (Strong Straddling Set Systems [MSW14]).** Let  $k$  be a fixed integer and let  $\mathcal{S} = \{1, \dots, k^2\}$ . Then the following partitions  $S_b = (S_{b,1}, \dots, S_{b,k})$ , for  $b \in \{0, 1\}$ , form a strong  $k$ -straddling set system over  $\mathcal{S}$ :

$$\begin{aligned} S_{0,i} &= \{k(i-1) + 1, k(i-1) + 2, \dots, ki\} \\ S_{1,i} &= \{i, k + i, 2k + i, \dots, k(k-1) + i\}. \end{aligned}$$

This construction naturally extends to any set with  $k^2$  elements.

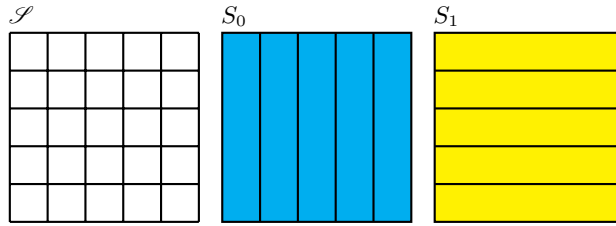


Fig. 1. Construction of strong 5-straddling set systems

Strong straddling set systems also satisfy the following lemma. Please refer to [BMSZ16] for proofs of constructions and lemma.

**Lemma 9.** *Let  $S_0, S_1$  be a strong  $k$ -straddling set system over a set  $\mathcal{U}$ . Then for any  $\mathcal{T} \subsetneq \mathcal{U}$  that can be written as a disjoint union of sets from  $S_0, S_1$ , there is a unique  $b \in \{0, 1\}$  such that  $\mathcal{T} = \cup_{i \in I} S_{b,i}$  for some  $I \subseteq \{1, \dots, k\}$ .*

### 3 Our Construction

Let us now describe our construction of an XOR-RKA-secure pseudorandom function, for security parameter  $\kappa$ , key set  $\mathcal{K} = \{0, 1\}^k$ , with  $k = 2\kappa$ , and domain  $\mathcal{D} = \{0, 1\}^n$  for some integer  $n$ .

#### 3.1 Intuition

**Construction Overview.** The starting point of our construction is the Naor-Reingold pseudorandom function, defined as  $\text{NR} : (\vec{a}, x) \in \mathbb{Z}_p^{2n} \times \{0, 1\}^n \mapsto g^{\prod_{i=1}^n a_i x_i}$ , where  $\vec{a}$  is the secret key. As we are interested in XOR relations, we want the key to be a bitstring. A simple solution is to tweak this construction by considering the function  $f_{\vec{a}, \vec{c}} : (\mathcal{K}, x) \in \{0, 1\}^k \times \{0, 1\}^n \mapsto g^{\prod_{i=1}^k a_i \kappa_i \cdot \prod_{j=1}^n c_{i,x_j}}$ ,

with  $\vec{a} \in \mathbb{Z}_p^k$  and  $\vec{c} \in \mathbb{Z}_p^n$ . It is easy to see that without knowing  $\vec{a}$  nor  $\vec{c}$ , the outputs of this function are computationally indistinguishable from random (they correspond to NR evaluations with key  $(\vec{a}, \vec{c})$  and on input  $(K, x)$ ). However, given a key  $K \in \{0, 1\}^k$ , one needs the values  $\vec{a}, \vec{c}$  in order to be able to evaluate this function, so these values need to be made public. Then, it becomes very easy, even without knowing  $K$ , to distinguish this function from a random one.

That is why we use a multilinear map: this allows us to publicly reveal low-level encodings of elements in  $\vec{a}$  and  $\vec{c}$ . These encodings let anyone evaluate the function on any key  $K$  and any input  $x$ , while keeping the outputs of the function computationally indistinguishable from random to an adversary that does not know the secret key  $K$ . Formally, we let  $\mathcal{U} = \{1, \dots, k+n\}$  be the set of indices for a multilinear map,  $(a_{i,b})_{i \in \{1, \dots, k\}, b \in \{0, 1\}}$  and  $(c_{j,b})_{j \in \{1, \dots, n\}, b \in \{0, 1\}}$  be random scalars in  $\mathbb{Z}_p$  and  $\hat{a}_{i,b} = [a_{i,b}]_{\{i\}}$  be an encoding of  $a_{i,b}$  at index  $i$  and  $\hat{c}_{j,b} = [c_{j,b}]_{\{j+k\}}$  be an encoding of  $c_{j,b}$  at index level  $j+k$ . We then consider the function:

$$F_{\text{pp}}(K, x) = \left[ \prod_{i=1}^k a_{i, K_i} \prod_{j=1}^n c_{j, x_j} \right]_{\mathcal{U}} = \prod_{i=1}^k \hat{a}_{i, K_i} \cdot \prod_{j=1}^n \hat{c}_{j, x_j} ,$$

with public parameters  $\text{pp}$  including the public parameters of the multilinear map as well as  $\{\hat{a}_{i,b}\}_{i \in \{1, \dots, k\}, b \in \{0, 1\}}$  and  $\{\hat{c}_{j,b}\}_{j \in \{1, \dots, n\}, b \in \{0, 1\}}$ .

This construction can be easily proven to be an XOR-RKA secure pseudo-random function in the generic multilinear map model, and it is also easy to show that it does not let an adversary create encodings of zero. However, it seems very hard to prove that this construction is secure under a non-interactive assumption<sup>10</sup>. Hence, we modify this construction by using a more complex set of indices and straddling sets. While this makes the proof in the generic multilinear map model a bit harder, this allows us to prove the security of our construction under non-interactive assumptions, whose hardness seems plausible even with current instantiations of multilinear maps. In particular, we prove in the full version [ABP17] that these assumptions are secure in the generic multilinear map model and do not let an adversary generate (non-trivial) encodings of zero.

**Proof Overview.** In the proof, we need to show that an oracle  $(s, x) \mapsto F_{\text{pp}}(K \oplus s, x)$  (where  $K$  is chosen secretly uniformly at random) looks indistinguishable from a random oracle. We first remark that we can write  $F_{\text{pp}}(K \oplus s, x)$  as:

$$F_{\text{pp}}(K \oplus s, x) = \prod_{i=1}^k \hat{\gamma}_{i, s_i} \cdot \prod_{j=1}^n \hat{c}_{j, x_j} ,$$

<sup>10</sup> In particular, the strategy we would like to use (replacing  $\hat{a}_{i,b}$  by independent random elements  $\hat{\gamma}_i$  when evaluating the function) does not work, as the adversary would be able to distinguish between the two games by comparing  $\hat{a}_i, 0$  and  $\hat{a}_i, 1$  to  $\hat{\gamma}_i$  (by computing the differences and applying `MM.ZeroTest`).

where  $\hat{\gamma}_{i,b} = [a_{i,K_i \oplus b}]_{\{i\}}$  is an encoding of  $a_{i,K_i \oplus b}$  for  $i \in \{1, \dots, k\}$ . Thus, instead of using  $K \in \{0, 1\}^k$  as the key, we can use an alternative private representation:  $(\hat{\gamma}_{i,b})_{i \in \{1, \dots, k\}}$ .

The main idea of our reduction is to be able to replace this private representation of the key, by completely random group elements, independent of the public parameters. We remark that the function

$$((\hat{\gamma}_{i,b})_{i \in \{1, \dots, k\}}, s) \mapsto \prod_{i=1}^k \hat{\gamma}_{i,s_i} ,$$

is a slight variant of the Naor-Reingold pseudorandom function with  $(\hat{\gamma}_{i,b})_{i \in \{1, \dots, k\}}$  being the key and  $s \in \{0, 1\}^k$  the input. It is actually possible to prove it is a pseudorandom function under a DDH-like assumption, and from that to prove that our construction is XOR-RKA-secure.

Unfortunately, it is obviously not true that given the public parameters  $(\hat{a}_{i,b})_{i \in \{1, \dots, k\}, b \in \{0, 1\}}$  (which are encodings of  $a_{i,b}$ ), the real group elements  $(\hat{\gamma}_{i,b})_{i \in \{1, \dots, k\}}$  (which are encodings  $a_{i,K_i \oplus b}$ ) are indistinguishable from encodings of independent uniform values: it is straightforward to check that  $\hat{a}_{i,b}$  corresponds to the same scalar as  $\hat{\gamma}_{i,b}$  or  $\hat{\gamma}_{i,1-b}$  (depending whether  $K_i = 0$  or  $1$ ), using `MM.ZeroTest` (and after multiplying these group elements by the same group elements to get a top-level encodings). Our solution is to use more complex index sets based on strong straddling sets to solve this issue.

The first step consists in using a first strong  $k$ -straddling set  $\mathcal{S}_0$ : we use the indices of the first partition for  $\hat{a}_{i,0}$  and  $\hat{a}_{i,1}$  (for each  $i$ ,  $\hat{a}_{i,0}$  and  $\hat{a}_{i,1}$  get the same index), and the indices of the second partition for  $\hat{\gamma}_{i,0}$  and  $\hat{\gamma}_{i,1}$ . This prevents the adversary from comparing one group element  $\hat{a}_{i,b}$  with a group element  $\hat{\gamma}_{i,b'}$  directly. But this is not sufficient, as the adversary still could check whether:

$$\prod_{i=1}^k (\hat{a}_{i,0} + \hat{a}_{i,1}) = \prod_{i=1}^k (\hat{\gamma}_{i,0} + \hat{\gamma}_{i,1}) ,$$

for example. When  $(\hat{\gamma}_{i,b})_{i \in \{1, \dots, k\}}$  are correctly generated, the equality is satisfied, while otherwise, it is not with overwhelming probability. More generally, the adversary can generate expression which contains an exponential number of monomials when expanded. We do not know how to prove anything reasonable on these expressions, so instead, we are using  $k$  additional strong  $k$ -straddling sets to prevent this from happening, in a similar way as they are used in [Zim15].

### 3.2 Actual Construction

**Index set.** First, similarly to [Zim15], for each  $i \in \{0, \dots, k\}$ , we construct a strong  $k$ -straddling set system over a set  $\mathcal{S}_i$  of  $2k - 1$  fresh formal symbols. We denote by  $S_{i,b}$  the two partitions forming each of this straddling set, for  $b \in \{0, 1\}$ , and by  $S_{i,b,j}$  their elements, for  $1 \leq j \leq k$ . We also define:

$$\text{BitCommit}_{i,b} = S_{i,b,i} \quad \text{BitFill}_{i_1, i_2, b_1, b_2} = S_{i_1, b_1, i_2} \cdot S_{i_2, b_2, i_1}$$

for any  $i, i_1, i_2 \in \{1, \dots, k\}$  and  $b, b_1, b_2 \in \{0, 1\}$ . Intuitively, the straddling set systems  $\mathcal{S}_i$  for  $i \geq 1$  play the same role as in [Zim15] (preventing the adversary from mixing an exponential number of inputs), while  $\mathcal{S}_0$  is used in the proof to prevent the adversary from mixing the private representation of the key with the public parameters.

Let  $X_j$  be fresh formal symbols for  $j \in \{1, \dots, n\}$ . We then define the top-level index set as follows:

$$\mathcal{U} = \prod_{i=0}^k \mathcal{S}_i \prod_{j=1}^n X_j .$$

**Setup.** The algorithm **Setup** first generates the parameters  $(\text{mm.pp}, \text{mm.sp}, p)$  for the multilinear map by running  $\text{MM.Setup}(1^\kappa, \mathcal{U})$ . Then it generates the following elements:

$$\begin{aligned} a_{i,b} &\stackrel{\$}{\leftarrow} \mathbb{Z}_p && \text{for } i \in \{1, \dots, k\} \text{ and } b \in \{0, 1\} \\ \hat{a}_{i,b} &\leftarrow [a_{i,b}]_{S_{0,0,i} \text{BitCommit}_{i,b}} && \text{for } i \in \{1, \dots, k\} \text{ and } b \in \{0, 1\} \\ c_{j,b} &\stackrel{\$}{\leftarrow} \mathbb{Z}_p && \text{for } j \in \{1, \dots, n\} \text{ and } b \in \{0, 1\} \\ \hat{c}_{j,b} &\leftarrow [c_{j,b}]_{X_j} && \text{for } j \in \{1, \dots, n\} \text{ and } b \in \{0, 1\} \\ \hat{z}_{i_1, i_2, b_1, b_2} &\leftarrow [1]_{\text{BitFill}_{i_1, i_2, b_1, b_2}} && \text{for } i_1, i_2 \in \{1, \dots, k\} \text{ and } b_1, b_2 \in \{0, 1\} \\ &&& \text{with } i_1 < i_2, \end{aligned}$$

and outputs the following parameters:

$$\text{pp} = \left( \text{mm.pp}, (\hat{a}_{i,b})_{i,b}, (\hat{c}_{j,b})_{j,b}, (\hat{z}_{i_1, i_2, b_1, b_2})_{i_1, i_2, b_1, b_2} \right) .$$

Intuitively,  $\text{BitCommit}_{i,b} = S_{i,b,i}$  is associated to the (public) encoding used to evaluate the function if the  $i$ -th bit of the key is  $b$ . By definition of a straddling set, the only way to reach the top-level  $\mathcal{U}$ , which contains  $\mathcal{S}_i$ , once we have used an encoding with index  $S_{i,b,i}$  is to use every index  $S_{i,b,j}$  with  $j \neq i$ . These is done by multiplying the terms  $\hat{z}_{i,j,K_i,K_j}$ . Therefore, using  $K_i = b$  is like ‘‘committing’’ to the partition  $S_{i,b}$  of  $\mathcal{S}_i$ , and terms  $\hat{z}_{i,j,K_i,K_j}$  are then used to ‘‘fill’’ this partition.

*Remark 10.* For the sake of simplicity, we set  $\hat{z}_{i_1, i_2, b_1, b_2}$  to be encodings of 1, but one could also simply set it to encodings of a random value, as long as they are all encodings of the same value.

**Evaluation.** The output of the PRF on a key  $K \in \{0, 1\}^k$  and an input  $x \in \{0, 1\}^n$  is

$$F_{\text{pp}}(K, x) = \text{MM.Extract} \left( \prod_{i=1}^k \hat{a}_{i, K_i} \prod_{j=1}^n \hat{c}_{j, x_j} \prod_{i_1=1}^k \prod_{i_2=i_1+1}^k \hat{z}_{i_1, i_2, K_{i_1}, K_{i_2}} \right) .$$

We can re-write it as:

$$F_{\text{pp}}(K, x) = \text{MM.Extract} \left( \left[ \prod_{i=1}^k a_{i, K_i} \prod_{j=1}^n c_{j, x_j} \right]_{\mathcal{U}} \right) .$$



**Extraction.** As explained in Remark 4, the role of extraction (MM.Extract) is dual. First, it ensures the correctness of the PRF, as in currently known instantiations of multilinear maps [GGH13a, CLT13, GGH15, CLT15], a scalar has many different encodings. Second, it is used in our proof of security under non-interactive assumptions in Section 5, as in the security proof we change the way the group element

$$\prod_{i=1}^k \hat{a}_{i,K_i} \prod_{j=1}^n \hat{c}_{j,x_j} \prod_{i_1=1}^k \prod_{i_2=i_1+1}^k \hat{z}_{i_1,i_2,K_{i_1},K_{i_2}}$$

is computed. We indeed recall that due to the fact that a scalar has many different encodings, any group element (as the above one) leaks information on the exact computation used to obtain it, instead of just depending on its discrete logarithm. The usual way to solve this issue is to randomize the resulting group element using encodings of zero. However, in this paper, we do not want to use any encoding of zeros, hence the requirement for this extraction. For the proof in the generic multilinear map model in Section 4, this is not an issue, and we just ignore the extraction (see Remark 4).

## 4 Security in the Generic Multilinear Map Model

In this section, we prove the security of our construction in the generic multilinear map model. As already explained at the end of Section 3, we suppose in this section that no extraction is performed. We actually even prove that no polynomial-time adversary can construct a (non-trivial) encoding of zero, in any of the two experiments RKPRFReal and RKPRFRand, with non-negligible probability. This implies, in particular, that these two experiments cannot be distinguished by a polynomial-time adversary, in the generic multilinear map model, as an adversary only sees handles which only leak information when two of them correspond to the same (top-level) element.

This section is mainly a warm-up to familiarize with the model, since the fact that we prove our construction under some assumptions that are proven secure in the generic multilinear map model and proven to not let an adversary generate encodings of zero also implies the results below. However, it is very simple to modify the proof below in order to prove the security of the simplified construction proposed in Section 3.1, which is of independent interest.

We first need to formally define the notion of (non-trivial) encoding of zero. We follow the definition of Badrinarayanan et al. [BMSZ16].

**Definition 11 ((Non-trivial) encoding of zero).** *An adversary  $\mathcal{A}$  in the generic multilinear map model with multilinear map oracle  $\mathcal{M}$  returns a (non-trivial) encoding of zero if it returns a handle  $h$  (output by  $\mathcal{M}$ ) such that  $h$  corresponds to the element 0 in  $\mathcal{M}$ 's table and the polynomial corresponding to the handle is not identically null.*

**Theorem 12 (Impossibility of constructing encodings of zero).** *In the generic multilinear map model with oracle  $\mathcal{M}$ , for any adversary  $\mathcal{A}$  making at most  $q_{\mathcal{M}}$  queries to the oracle  $\mathcal{M}$  and  $q_{\mathbf{RKFn}}$  queries to the oracle  $\mathbf{RKFn}$ , we have:*

$$\Pr \left[ \text{PRFReal}_{F_{\text{pp}}}^{\mathcal{A}} \Rightarrow \text{an encoding of } 0 \right] \leq q_{\mathcal{M}} \left( \frac{q_{\mathbf{RKFn}}}{2^k} + \frac{k+n}{p} \right)$$

and

$$\Pr \left[ \text{PRFRand}_{F_{\text{pp}}}^{\mathcal{A}} \Rightarrow \text{an encoding of } 0 \right] \leq q_{\mathcal{M}} \frac{k+n}{p} .$$

*Proof (Theorem 12).* We first introduce a technical lemma, whose proof is given in the full version [ABP17].

**Lemma 13.** *Let  $k$  and  $n$  be two positive integers. Let  $\mathcal{U}$  be the index defined in Section 3. Let  $\hat{z}_{i_1, i_2, b_1, b_2} = [1]_{\text{BitFill}_{i_1, i_2, b_1, b_2}}$  for  $1 \leq i_1 < i_2 \leq k$  and  $b_1, b_2 \in \{0, 1\}$ . Let  $Z_1$  and  $Z_2$  be two subsets of  $\{(i_1, i_2) \mid 1 \leq i_1 < i_2 \leq k\} \times \{0, 1\}^2$ . If  $t_1 = \prod_{(i_1, i_2, b_1, b_2) \in Z_1} \hat{z}_{i_1, i_2, b_1, b_2}$  and  $t_2 = \prod_{(i_1, i_2, b_1, b_2) \in Z_2} \hat{z}_{i_1, i_2, b_1, b_2}$  have the same index set, then  $Z_1 = Z_2$ .*

We need to show that the adversary cannot generate a non-trivial encoding of zero.

$\text{RKPRFRand}_{F_{\text{pp}}}^{\mathcal{A}}$ . We start by proving it in the game  $\text{RKPRFRand}_{F_{\text{pp}}}^{\mathcal{A}}$ . In this game, except for  $\hat{z}_{i_1, i_2, b_1, b_2}$ , all the handles the adversary sees correspond to fresh new formal variables, as the oracle  $\mathbf{RKFn}$  only returns fresh new formal variables (of index  $\mathcal{U}$ ). The only polynomials the adversary can generate are therefore of the form:

$$P = \sum_{\ell=1}^L Q_{\ell} \prod_{(i_1, i_2, b_1, b_2) \in Z_{\ell}} \hat{z}_{i_1, i_2, b_1, b_2} ,$$

where  $Q_{\ell}$  are polynomials over all the elements except  $\hat{z}_{i_1, i_2, b_1, b_2}$ , and  $Z_{\ell}$  are distinct subsets of  $\{(i_1, i_2) \mid 1 \leq i_1 < i_2 \leq k\} \times \{0, 1\}^2$  ( $L$  might be exponential in  $q_{\mathcal{M}}$ , but that does not matter for what follows).

Let us now show that if  $P$  is not the zero polynomial, then when replacing  $\hat{z}_{i_1, i_2, b_1, b_2}$  by 1, the resulting polynomial is still a non-zero polynomial. From Lemma 13, one can assume that elements  $\prod_{(i_1, i_2, b_1, b_2) \in Z_{\ell}} \hat{z}_{i_1, i_2, b_1, b_2}$  all have distinct indices. Therefore, the polynomials  $Q_{\ell}$  all have distinct indices too. No monomial in two different  $Q_{\ell}$  of the sum  $\sum_{\ell} Q_{\ell}$  (when forgetting the indices) can therefore cancel out, otherwise this would mean that the adversary can construct two equal monomials (without  $\hat{z}_{i, b}$ ) with two different indices. This is impossible as, except for  $\hat{z}_{i, b}$ , two distinct handles correspond to two fresh variables (or in other words, all the handles except  $\hat{z}_{i, b}$  are encodings of scalars chosen uniformly and independently at random).

We therefore simulate the oracle  $\mathcal{M}$  as follows: we do everything as normal, but we make the zero-testing oracle always output “non-zero” except when its

input corresponds to the zero polynomial. The Schwarz-Zippel lemma ensures that any non-zero polynomial of degree at most  $k + n$  and whose variables are fixed to uniformly random values in  $\mathbb{Z}_p$  does not evaluate to zero, except with probability at most  $(k + n)/p$ . In other words, the zero-testing oracle outputs “zero” on a non-zero polynomial with probability at most  $(k + n)/p$ , as this polynomial remains non-zero and has degree at most  $(k + n)$ , when we replace  $\hat{z}_{i_1, i_2, b_1, b_2}$  by 1. As we can suppose that the zero-testing oracle is queried with the output of the adversary without loss of generality, using at most  $q_{\mathcal{M}}$  hybrid games (replacing one-by-one every output of the zero-testing oracle with “non-zero”) we get that:

$$\Pr \left[ \text{PRFRand}_{F_{pp}}^{\mathcal{A}} \Rightarrow \text{an encoding of } 0 \right] \leq q_{\mathcal{M}} \frac{k + n}{p} .$$

$\text{RKPRFReal}_{F_{pp}}^{\mathcal{A}}$ . Let us now look at the game  $\text{RKPRFReal}_{F_{pp}}^{\mathcal{A}}$ . The analysis is more complicated as the adversary has access to new formal variables

$$\hat{y}_{s,x} = F_{pp}(K \oplus s, x)$$

returned by queries  $\mathbf{RKFn}(\phi_s, x)$ .

We use the same simulator as in the previous case. We need to show that if a polynomial  $P$  produced by the adversary is not zero, it remains non-zero when  $\hat{z}_{i_1, i_2, b_1, b_2}$  is replaced by 1 and  $\hat{y}_{s,x}$  is replaced by its value.

We first consider the case where  $P$  is not a top-level polynomial. In this case,  $P$  cannot contain these new variables  $\hat{y}_{s,x}$  as these variables are top-level. Then, as in  $\text{RKPRFRand}_{F_{pp}}^{\mathcal{A}}$ , the zero-testing oracle outputs “non-zero” except with probability at most  $(k + n)/p$ .

Let us now suppose that  $P$  is a top-level polynomial. This polynomial has the form:

$$P = \sum_{\ell=1}^L Q_{\ell} \prod_{i=1}^k \hat{a}_{i, K'_{\ell, i}} \prod_{i_1=1}^k \prod_{i_2=i_1+1}^k \hat{z}_{i_1, i_2, K'_{\ell, i_1}, K'_{\ell, i_2}} + \sum_{j=1}^{q'} \lambda_j \hat{y}_{s_j, x_j} ,$$

with  $L$  being a non-negative integer (possibly exponential in  $q_{\mathcal{M}}$ ),  $Q_{\ell}$  being non-zero polynomials in the formal variables  $\hat{c}_{j,b}$ ,  $K'_{\ell}$  being distinct bitstrings in  $\{0, 1\}^k$  (chosen by the adversary),  $q'$  an integer less or equal to  $q_{\mathbf{RKFn}}$ ,  $(s_1, x_1), \dots, (s_{q'}, x_{q'})$  queries to  $\mathbf{RKFn}$ , and  $\lambda_j$  some scalar in  $\mathbb{Z}_p$ . Indeed, the adversary can ask for an encoding of any polynomial of the form  $Q_{\ell} \prod_{i=1}^k \hat{a}_{i, K'_{\ell, i}}$ , and by definition of straddling set systems, the unique way to obtain a top-level encoding from such a polynomial is by multiplying it with an encoding of  $\prod_{i_1=1}^k \prod_{i_2=i_1+1}^k \hat{z}_{i_1, i_2, K'_{\ell, i_1}, K'_{\ell, i_2}}$ .

Let us suppose that  $P$  is not zero but becomes zero when  $\hat{z}_{i_1, i_2, b_1, b_2}$  is replaced by 1 and  $\hat{y}_{s,x}$  is replaced by its value. In this case, in particular, the first monomial (for any order) of the term

$$Q_1 \prod_{i=1}^k \hat{a}_{i, K'_{1, i}} \prod_{i_1=1}^k \prod_{i_2=i_1+1}^k \hat{z}_{i_1, i_2, K'_{1, i_1}, K'_{1, i_2}}$$

necessarily needs to be canceled out by some  $\hat{y}_{s_j, x_j}$ . The probability over  $K \xleftarrow{\$} \{0, 1\}^k$  that this happens is at most:

$$\Pr[\exists j' \in \{1, \dots, q'\}, K'_1 = K \oplus s_{j'}] \leq \frac{q'}{2^k} \leq \frac{q_{\mathbf{RKFn}}}{2^k} .$$

As before, thanks to the Schwarz-Zippel lemma, we get that the zero-testing oracle outputs “zero”, on input a non-zero polynomial, with probability at most:

$$\frac{q_{\mathbf{RKFn}}}{2^k} + \frac{k+n}{p} .$$

This concludes the proof of Theorem 12. □

*Remark 14.* We never use the properties of the straddling set system  $\mathcal{S}_0$  in this proof. These properties are only used in our proof under non-interactive assumptions in Section 5.

We obtain the following immediate corollary.

**Corollary 15 (Security in the generic multilinear map model).** *Let  $\mathcal{A}$  be an adversary in the generic multilinear map model with oracle  $\mathcal{M}$  against the XOR-RKA security of the PRF  $F$  defined in Section 3. If  $\mathcal{A}$  makes at most  $q_{\mathcal{M}}$  queries to the oracle  $\mathcal{M}$  and  $q_{\mathbf{RKFn}}$  queries to the oracle  $\mathbf{RKFn}$ , then:*

$$\mathbf{Adv}_{\Phi_{\oplus}, F_{\text{pp}}}^{\text{prf-rka}}(\mathcal{A}) \leq \frac{q_{\mathcal{M}} q_{\mathbf{RKFn}}}{2^k} + \frac{2q_{\mathcal{M}}(k+n)}{p} .$$

*Proof (Corollary 15).* We just consider an intermediate game where we simulate everything as before except the zero-testing oracle which always outputs “non-zero” unless its input is zero, as a polynomial. This game is indistinguishable from both  $\text{RKPRFReal}_{F_{\text{pp}}}^{\mathcal{A}}$  and  $\text{RKPRFRand}_{F_{\text{pp}}}^{\mathcal{A}}$  according to Theorem 12 (up to the bounds in this lemma). Corollary 15 follows. □

## 5 Security Under Non-Interactive Assumptions

In this section, we show that our construction is an XOR-RKA PRF under two non-interactive assumptions defined below.

### 5.1 Assumptions

We use two assumptions that we call the  $(k, n, X, Y)$ -XY-DDH assumption, which is roughly a generalization of the standard DDH assumption, and the  $(k, n)$ -Sel-Prod assumption. We show in the full version [ABP17] that both these assumptions are secure in the generic multilinear map model, and even that an adversary against these assumptions cannot generate encodings of zero. As explained in the “concrete instantiations” paragraph of Section 1, contrary to

most assumptions considered on multilinear maps (e.g., classical DDH-like assumptions and the multilinear subgroup assumption [GLW14, GLSW15]), these assumptions are therefore plausible at least with two current instantiations of multilinear maps [CLT13, GGH15].

To ensure the impossibility of generating encodings of zero, in these two assumptions, we restrict the adversary’s capabilities as follows: it is only provided parameters `mm.pp`, so it can only run `MM.Add`, `MM.Mult` and `MM.ZeroTest` (and of course use the elements generated by the assumption), but we do not allow the adversary to generate new encodings of a chosen scalar. In particular, this forces us to let the assumption contain the group elements  $\hat{z}_{i_1, i_2, b_1, b_2}$ . It is straightforward to get rid of these additional elements by allowing the adversary to generate any element of the multilinear map, at the cost of getting an implausible assumption under current instantiations of multilinear maps.

Finally, our assumption implicitly contains a list  $\mathcal{L}$  of a polynomial number of encodings of independent uniform random values at non-zero index, index being implicit parameters of the assumption. We could avoid this artifact with the previous proposition as well, or by giving a sufficient number of encodings of 0 and 1, but once again, in that case, the assumption would most likely not hold with currently known multilinear maps instantiations. We believe this is a small price to pay to get plausible assumptions, as the resulting assumptions are still non-interactive.

We insist on the fact that the encodings in  $\mathcal{L}$  are encodings of independent uniformly random scalars. At least in the generic multilinear group model, our assumptions hold whatever the list of indices of these encodings is. We do not have any constraint on this list of indices.

**Definition 16** ( $(k, n, X, Y)$ -XY-DDH). *Let  $k$  and  $n$  be two positive integers. Let  $X$  and  $Y$  be two non-empty and disjoint indices in the index set  $\mathcal{U}$  of our construction in Section 3. The advantage of an adversary  $\mathcal{D}$  against the  $(k, n, X, Y)$ -XY-DDH problem is:*

$$\mathbf{Adv}^{(k, n, X, Y)\text{-XY-DDH}}(\mathcal{D}) = \Pr [(k, n, X, Y)\text{-XY-DDH-L}^{\mathcal{D}} \Rightarrow 1] - \Pr [(k, n, X, Y)\text{-XY-DDH-R}^{\mathcal{D}} \Rightarrow 1],$$

where the games  $(k, n, X, Y)$ -XY-DDH-L $^{\mathcal{D}}$  and  $(k, n, X, Y)$ -XY-DDH-R $^{\mathcal{D}}$  are defined in Fig. 2. The  $(n, k, X, Y)$ -XY-DDH assumption holds when this advantage is negligible for any polynomial-time adversary  $\mathcal{D}$ .

This assumption is very close to the classical DDH assumption with indices, with two main differences: the presence of elements  $\hat{z}_{i_1, i_2, b_1, b_2}$  which are necessary to prove our construction and the implicit presence of encodings of random values at non-zero indices (list  $\mathcal{L}$  described previously) instead of a polynomial number of encodings of 0 and 1. Without the elements  $\hat{z}_{i_1, i_2, b_1, b_2}$ , the proof of this assumption in the generic multilinear map model would be completely straightforward. The difficulty of the proof is to deal with these elements.

In the security proof of our construction, this assumption is used in a similar way as the DDH assumption in the proof of the Naor-Reingold PRF.

$(k, n, X, Y)$ -XY-DDH-L <sup><math>\mathcal{D}</math></sup> <hr/> <p><b>proc Initialize</b>  <math>(\text{mm.sp}, \text{mm.pp}) \xleftarrow{\\$} \text{MM.Setup}(1^\kappa, \mathcal{U})</math>  <math>d_0, d_1, e \xleftarrow{\\$} \mathbb{Z}_p</math>  <math>\hat{d}_0 \leftarrow [d_0]_X; \quad \hat{d}_1 \leftarrow [d_1]_X</math>  <math>\hat{e}_0 \leftarrow [ed_0]_{XY}; \quad \hat{e}_1 \leftarrow [ed_1]_{XY}</math>  For <math>i_1, i_2 \in \{1, \dots, k\}</math> and <math>b_1, b_2 \in \{0, 1\}</math>  <math>\hat{z}_{i_1, i_2, b_1, b_2} \leftarrow [1]_{\text{BitFill}_{i_1, i_2, b_1, b_2}}</math>  Return <math>(\mathcal{L}, \text{mm.pp}, \hat{d}_0, \hat{d}_1, \hat{e}_0, \hat{e}_1,</math>  <math>\quad\quad\quad (\hat{z}_{i_1, i_2, b_1, b_2}))</math></p> <p><b>proc Finalize(<math>b</math>)</b>  Return <math>b</math></p>	$(k, n, X, Y)$ -XY-DDH-R <sup><math>\mathcal{D}</math></sup> <hr/> <p><b>proc Initialize</b>  <math>(\text{mm.sp}, \text{mm.pp}) \xleftarrow{\\$} \text{MM.Setup}(1^\kappa, \mathcal{U})</math>  <math>d_0, d_1, e_0, e_1 \xleftarrow{\\$} \mathbb{Z}_p</math>  <math>\hat{d}_0 \leftarrow [d_0]_X; \quad \hat{d}_1 \leftarrow [d_1]_X</math>  <math>\hat{e}_0 \leftarrow [e_0]_{XY}; \quad \hat{e}_1 \leftarrow [e_1]_{XY}</math>  For <math>i_1, i_2 \in \{1, \dots, k\}</math> and <math>b_1, b_2 \in \{0, 1\}</math>  <math>\hat{z}_{i_1, i_2, b_1, b_2} \leftarrow [1]_{\text{BitFill}_{i_1, i_2, b_1, b_2}}</math>  Return <math>(\mathcal{L}, \text{mm.pp}, \hat{d}_0, \hat{d}_1, \hat{e}_0, \hat{e}_1,</math>  <math>\quad\quad\quad (\hat{z}_{i_1, i_2, b_1, b_2}))</math></p> <p><b>proc Finalize(<math>b</math>)</b>  Return <math>b</math></p>
$(k, n)$ -Sel-Prod-L <sup><math>\mathcal{D}</math></sup> <hr/> <p><b>proc Initialize</b>  <math>(\text{mm.sp}, \text{mm.pp}) \xleftarrow{\\$} \text{MM.Setup}(1^\kappa, \mathcal{U})</math>  <math>K \xleftarrow{\\$} \{0, 1\}^k</math>  For <math>i \in \{1, \dots, k\}</math> and <math>b \in \{0, 1\}</math>  <math>a_{i,b} \xleftarrow{\\$} \mathbb{Z}_p; \quad \gamma_{i, K_i \oplus b} \leftarrow a_{i,b}</math>  <math>\hat{a}_{i,b} \leftarrow [a_{i,b}]_{S_{0,0,i} \text{BitCommit}_{i,b}}</math>  <math>\hat{\gamma}_{i, K_i \oplus b} \leftarrow [\gamma_{i, K_i \oplus b}]_{S_{0,1,i} \text{BitCommit}_{i, K_i \oplus b}}</math>  For <math>i_1, i_2 \in \{1, \dots, k\}</math> and <math>b_1, b_2 \in \{0, 1\}</math>  <math>\hat{z}_{i_1, i_2, b_1, b_2} \leftarrow [1]_{\text{BitFill}_{i_1, i_2, b_1, b_2}}</math>  Return <math>(\mathcal{L}, \text{mm.pp}, (\hat{a}_{i,b}), (\hat{\gamma}_{i,b}),</math>  <math>\quad\quad\quad (\hat{z}_{i_1, i_2, b_1, b_2}))</math></p> <p><b>proc Finalize(<math>b</math>)</b>  Return <math>b</math></p>	$(k, n)$ -Sel-Prod-R <sup><math>\mathcal{D}</math></sup> <hr/> <p><b>proc Initialize</b>  <math>(\text{mm.sp}, \text{mm.pp}) \xleftarrow{\\$} \text{MM.Setup}(1^\kappa, \mathcal{U})</math>  For <math>i \in \{1, \dots, k\}</math> and <math>b \in \{0, 1\}</math>  <math>a_{i,b} \xleftarrow{\\$} \mathbb{Z}_p; \quad \gamma_{i,b} \xleftarrow{\\$} \mathbb{Z}_p</math>  <math>\hat{a}_{i,b} \leftarrow [a_{i,b}]_{S_{0,0,i} \text{BitCommit}_{i,b}}</math>  <math>\hat{\gamma}_{i,b} \leftarrow [\gamma_{i,b}]_{S_{0,1,i} \text{BitCommit}_{i,b}}</math>  For <math>i_1, i_2 \in \{1, \dots, k\}</math> and <math>b_1, b_2 \in \{0, 1\}</math>  <math>\hat{z}_{i_1, i_2, b_1, b_2} \leftarrow [1]_{\text{BitFill}_{i_1, i_2, b_1, b_2}}</math>  Return <math>(\mathcal{L}, \text{mm.pp}, (\hat{a}_{i,b}), (\hat{\gamma}_{i,b}),</math>  <math>\quad\quad\quad (\hat{z}_{i_1, i_2, b_1, b_2}))</math></p> <p><b>proc Finalize(<math>b</math>)</b>  Return <math>b</math></p>

**Fig. 2.** Games defining the advantage of an adversary  $\mathcal{D}$  against the XY-DDH and Sel-Prod problems.

**Definition 17** ( $(k, n)$ -Sel-Prod). *Let  $k$  and  $n$  be two positive integers. The advantage of an adversary  $\mathcal{D}$  against the  $(k, n, X, Y)$ -Sel-Prod problem is:*

$$\text{Adv}^{(k,n)\text{-Sel-Prod}}(\mathcal{D}) = \Pr[(k, n)\text{-Sel-Prod-L}^{\mathcal{D}} \Rightarrow 1] - \Pr[(k, n)\text{-Sel-Prod-R}^{\mathcal{D}} \Rightarrow 1],$$

where the games  $(k, n)$ -Sel-Prod-L $^{\mathcal{D}}$  and  $(k, n)$ -Sel-Prod-R $^{\mathcal{D}}$  are defined in Fig. 2. The  $(n, k)$ -Sel-Prod assumption holds when this advantage is negligible for any polynomial-time adversary  $\mathcal{D}$ .

Intuitively, this assumption states that, given a low-level encodings of  $a_{i,0}$  and  $a_{i,1}$  at indices  $S_{0,i}$  from the first partition of the straddling set  $\mathcal{S}$ , where  $i \in \{1, \dots, k\}$ , then it is hard to distinguish low-level encodings of  $\gamma_{i,0} = a_{i,K_i}$  and  $\gamma_{i,1} = a_{i,1-K_i}$  (where  $K_i$  is a random bit) at indices  $S_{1,i}$  from encodings of fresh random values (at the same index). The hardness of this assumption crucially relies on two facts: First, one can only compare top-level encodings, and thus, the only way to compare elements whose index is in the first partition of  $\mathcal{S}$  with elements whose index is in the second partition is to combine  $k$  such elements to reach the same index. Second, the latter can be hard only if  $k$ , the size of the partitions, is big. Indeed, assume  $k = 2$ , then one can just guess  $K_1$  and  $K_2$  and check if the relation carries on between encodings of  $a_{i,b}$  and encodings of  $\gamma_{i,b}$ . Therefore, we show that if  $k$  is big enough, this assumption holds in the generic multilinear map model.

As explained in Section 3.1, this assumption is used to switch from the key  $K$  to a private independent key represented by the encodings  $\gamma_{i,b}$ . More precisely, under this assumption, we can replace the encodings  $\hat{a}_{i,K_i \oplus s_i}$  at index from the first partition of the straddling set  $\mathcal{S}_0$ , used in the computation of the output with relation  $s$ , to encodings of uniformly random scalars at index from the second partition of  $\mathcal{S}_0$ . In particular, doing this change, we no longer need to know the key  $K$  to simulate correctly the output, but only the relations  $s$  for each query.

*Remark 18.* For the sake of simplicity, we do not explicitly specify the noise level in our assumptions. It can easily be made to work with our proof.

## 5.2 Security of our Construction

In this whole section, we set  $\mathcal{S} = \prod_{i=0}^k \mathcal{S}_i$  and  $\mathcal{S}' = \prod_{i=1}^k \mathcal{S}_i$ , so  $\mathcal{S} = \mathcal{S}_0 \cdot \mathcal{S}'$ . Please refer to Section 3 for notation.

**Theorem 19 (Security under non-interactive assumptions).** *Let  $\mathcal{A}$  be a polynomial-time adversary against the XOR-RKA security of the PRF  $F$  defined in Section 3. We suppose that  $\mathcal{A}$  makes at most  $q_{\mathbf{RKF}_n}$  queries to the oracle  $\mathbf{RKF}_n$ . We can define an adversary  $\mathcal{D}$  against the  $(k, n)$ -Sel-Prod problem,  $(k-1)$  adversaries  $\mathcal{B}_{i'}$  against the  $(k, n, \mathcal{S}' \prod_{i=1}^{i'-1} S_{0,1,i}, S_{0,1,i'})$ -XY-DDH problem*

for  $i' \in \{2, \dots, k\}$ , and  $n$  adversaries  $\mathcal{C}_{j'}$  against the  $(k, n, \mathcal{S} \prod_{j=1}^{j'-1} X_j, X_{j'})$ -XY-DDH problem for  $j' \in \{1, \dots, n\}$ , such that:

$$\begin{aligned} \text{Adv}_{\Phi_{\oplus}, F_{\text{pp}}}^{\text{prf-rka}}(\mathcal{A}) &\leq \text{Adv}^{(k,n)\text{-Sel-Prod}}(\mathcal{D}) + \\ &\sum_{i'=2}^k q_{\text{RKFn}} \cdot \text{Adv}^{(k,n,\mathcal{S}' \prod_{i=1}^{i'-1} S_{0,1,i}, S_{0,1,i'})\text{-XY-DDH}}(\mathcal{B}_{i'}) + \\ &\sum_{j'=1}^n q_{\text{RKFn}} \cdot \text{Adv}^{(k,n,\mathcal{S} \prod_{j=1}^{j'-1} X_j, X_{j'})\text{-XY-DDH}}(\mathcal{C}_{j'}) . \end{aligned}$$

Furthermore, all these adversaries run in polynomial time (their running time is approximately the same as  $\mathcal{A}$ ).

Below, we provide a sketch of the proof. The full proof is given in the full version [ABP17].

**Sketch of proof.** The proof follows a sequence of hybrid games. The first hybrid corresponds exactly to  $\text{RKPRFReal}_F^{\mathcal{A}}$ , while the last game corresponds to  $\text{RKPRFRand}_F^{\mathcal{A}}$ . Here is how we proceed. First, instead of computing the output using encodings  $\hat{a}_{i,b}$  of  $a_{i,b}$  with index  $S_{0,0,i} \text{BitCommit}_{i,b}$ , we use encodings  $\hat{\gamma}_{i,b}$  of  $a_{i,K_i \oplus b}$  with index  $S_{0,1,i} \text{BitCommit}_{i,K_i \oplus b}$ . That is, we use the second partition  $S_{0,1}$  of the straddling set  $\mathcal{S}_0$  instead of the first one ( $S_{0,0}$ ) to reach top-level index (which contains  $\mathcal{S}_0$ ). Also, we now compute the output using only the relation  $s$  instead of the key  $K$ . More precisely, the output on a query  $(s, x)$  is computed as:

$$\text{MM.Extract} \left( \prod_{i=1}^k \hat{\gamma}_{i,s_i} \prod_{j=1}^n \hat{c}_{j,x_j} \prod_{i_1=1}^k \prod_{i_2=i_1+1}^k \hat{z}_{i_1,i_2,K_{i_1},K_{i_2}} \right),$$

which can be computed without knowing  $K$ . This does not change anything regarding the output (thanks to the extraction), so these two games are indistinguishable.

However, using the  $(k, n)$ -Sel-Prod assumption, we can now switch the encodings  $\hat{\gamma}_{i,b}$  to encodings of fresh random scalars  $\gamma_{i,b} \in \mathbb{Z}_p$ . The rest of the proof is very similar to the proof of the Naor-Reingold pseudorandom function. We do  $k + n$  hybrid games, where in the  $j$ -th hybrid, we just switch products of encodings  $\prod_{i=1}^j \hat{\gamma}_{i,s_i}$  to encodings of uniformly fresh random values using the XY-DDH assumption with the proper indices. These modifications are done in a lazy fashion to obtain a polynomial-time reduction.

**Acknowledgements.** The first author was supported in part by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013 Grant Agreement 339563 – CryptoCloud). The second author was supported in part by the Defense Advanced Research Projects Agency (DARPA) and Army Research Office (ARO) under Contract No. W911NF-15-C-0236. Part



of this work was done while the second author was at École normale supérieure, Paris, France, and at IBM Research, Yorktown Heights, NY, USA. Part of this work was done while the third author was at École normale supérieure, Paris, France, and at UCLA, Los Angeles, CA, USA.

## References

- ABD16. Martin R. Albrecht, Shi Bai, and Léo Ducas. A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 153–178. Springer, Heidelberg, August 2016.
- ABP15. Michel Abdalla, Fabrice Benhamouda, and Alain Passelègue. An algebraic framework for pseudorandom functions and applications to related-key security. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 388–409. Springer, Heidelberg, August 2015.
- ABP17. Michel Abdalla, Fabrice Benhamouda, and Alain Passelègue. Algebraic XOR-RKA-secure pseudorandom functions from post-zeroizing multilinear maps. Cryptology ePrint Archive, Report 2017/500, 2017. <http://eprint.iacr.org/2017/500>.
- ABPP14. Michel Abdalla, Fabrice Benhamouda, Alain Passelègue, and Kenneth G. Paterson. Related-key security for pseudorandom functions beyond the linear barrier. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 77–94. Springer, Heidelberg, August 2014.
- ABPP18. Michel Abdalla, Fabrice Benhamouda, Alain Passelègue, and Kenneth G. Paterson. Related-key security for pseudorandom functions beyond the linear barrier. *Journal of Cryptology*, 31(4):917–964, October 2018.
- AW14. Benny Applebaum and Eyal Widder. Related-key secure pseudorandom functions: The case of additive attacks. Cryptology ePrint Archive, Report 2014/478, 2014. <http://eprint.iacr.org/2014/478>.
- BC10. Mihir Bellare and David Cash. Pseudorandom functions and permutations provably secure against related-key attacks. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 666–684. Springer, Heidelberg, August 2010.
- BCM11. Mihir Bellare, David Cash, and Rachel Miller. Cryptography secure against related-key attacks and tampering. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 486–503. Springer, Heidelberg, December 2011.
- BDK05. Eli Biham, Orr Dunkelman, and Nathan Keller. Related-key boomerang and rectangle attacks. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 507–525. Springer, Heidelberg, May 2005.
- BDK<sup>+</sup>10. Alex Biryukov, Orr Dunkelman, Nathan Keller, Dmitry Khovratovich, and Adi Shamir. Key recovery attacks of practical complexity on AES-256 variants with up to 10 rounds. In Henri Gilbert, editor, *Advances*

- in *Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 299–319. Springer, Heidelberg, May / June 2010.
- BGK<sup>+</sup>14. Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Protecting obfuscation against algebraic attacks. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 221–238. Springer, Heidelberg, May 2014.
- Bih94. Eli Biham. New types of cryptanalytic attacks using related keys (extended abstract). In Tor Helleseth, editor, *Advances in Cryptology – EUROCRYPT’93*, volume 765 of *Lecture Notes in Computer Science*, pages 398–409. Springer, Heidelberg, May 1994.
- BK03. Mihir Bellare and Tadayoshi Kohno. A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 491–506. Springer, Heidelberg, May 2003.
- BLMR13. Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic PRFs and their applications. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 410–428. Springer, Heidelberg, August 2013.
- BMSZ16. Saikrishna Badrinarayanan, Eric Miles, Amit Sahai, and Mark Zhandry. Post-zeroizing obfuscation: New mathematical tools, and the case of evasive circuits. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 764–791. Springer, Heidelberg, May 2016.
- BP14. Abhishek Banerjee and Chris Peikert. New and improved key-homomorphic pseudorandom functions. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 353–370. Springer, Heidelberg, August 2014.
- BR06. Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426. Springer, Heidelberg, May / June 2006.
- BR14. Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In Yehuda Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 1–25. Springer, Heidelberg, February 2014.
- CFL<sup>+</sup>16. Jung Hee Cheon, Pierre-Alain Fouque, Changmin Lee, Brice Minaud, and Hansol Ryu. Cryptanalysis of the new CLT multilinear map over the integers. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 509–536. Springer, Heidelberg, May 2016.
- CGH<sup>+</sup>15. Jean-Sébastien Coron, Craig Gentry, Shai Halevi, Tancrede Lepoint, Hemanta K. Maji, Eric Miles, Mariana Raykova, Amit Sahai, and Mehdi Tibouchi. Zeroizing without low-level zeroes: New MMAP attacks and their limitations. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part I*, volume 9215 of *Lecture*

- Notes in Computer Science*, pages 247–266. Springer, Heidelberg, August 2015.
- CGH17. Yilei Chen, Craig Gentry, and Shai Halevi. Cryptanalyses of candidate branching program obfuscators. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part III*, volume 10212 of *Lecture Notes in Computer Science*, pages 278–307. Springer, Heidelberg, April / May 2017.
- CHKL18. Jung Hee Cheon, Minki Hhan, Jiseung Kim, and Changmin Lee. Cryptanalysis on the hhss obfuscation arising from absence of safeguards. *IEEE Access*, 6:40096–40104, 2018.
- CHL<sup>+</sup>15. Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 3–12. Springer, Heidelberg, April 2015.
- CJL16. Jung Hee Cheon, Jinhyuck Jeong, and Changmin Lee. An algorithm for ntru problems and cryptanalysis of the ggh multilinear map without a low-level encoding of zero. *LMS Journal of Computation and Mathematics*, 19(A):255–266, 2016.
- CLLT16. Jean-Sébastien Coron, Moon Sung Lee, Tancrede Lepoint, and Mehdi Tibouchi. Cryptanalysis of GGH15 multilinear maps. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 607–628. Springer, Heidelberg, August 2016.
- CLLT17. Jean-Sébastien Coron, Moon Sung Lee, Tancrede Lepoint, and Mehdi Tibouchi. Zeroizing attacks on indistinguishability obfuscation over CLT13. In Serge Fehr, editor, *PKC 2017: 20th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 10174 of *Lecture Notes in Computer Science*, pages 41–58. Springer, Heidelberg, March 2017.
- CLT13. Jean-Sébastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 476–493. Springer, Heidelberg, August 2013.
- CLT15. Jean-Sébastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. New multilinear maps over the integers. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 267–286. Springer, Heidelberg, August 2015.
- CS15. Benoit Cogliati and Yannick Seurin. On the provable security of the iterated Even-Mansour cipher against related-key and chosen-key attacks. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 584–613. Springer, Heidelberg, April 2015.
- CVW18. Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. GGH15 beyond permutation branching programs: Proofs, attacks, and candidates. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 577–607. Springer, Heidelberg, August 2018.

- FP15. Pooya Farshim and Gordon Procter. The related-key security of iterated Even-Mansour ciphers. In Gregor Leander, editor, *Fast Software Encryption – FSE 2015*, volume 9054 of *Lecture Notes in Computer Science*, pages 342–363. Springer, Heidelberg, March 2015.
- FX15. Eiichiro Fujisaki and Keita Xagawa. Efficient RKA-secure KEM and IBE schemes against invertible functions. In Kristin E. Lauter and Francisco Rodríguez-Henríquez, editors, *Progress in Cryptology - LATINCRYPT 2015: 4th International Conference on Cryptology and Information Security in Latin America*, volume 9230 of *Lecture Notes in Computer Science*, pages 3–20. Springer, Heidelberg, August 2015.
- GGH13a. Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17. Springer, Heidelberg, May 2013.
- GGH<sup>+</sup>13b. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual Symposium on Foundations of Computer Science*, pages 40–49. IEEE Computer Society Press, October 2013.
- GGH15. Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 498–527. Springer, Heidelberg, March 2015.
- GGM86. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, October 1986.
- GLSW15. Craig Gentry, Allison Bishop Lewko, Amit Sahai, and Brent Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. In Venkatesan Guruswami, editor, *56th Annual Symposium on Foundations of Computer Science*, pages 151–170. IEEE Computer Society Press, October 2015.
- GLW14. Craig Gentry, Allison B. Lewko, and Brent Waters. Witness encryption from instance independent assumptions. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 426–443. Springer, Heidelberg, August 2014.
- GMM<sup>+</sup>16. Sanjam Garg, Eric Miles, Pratyay Mukherjee, Amit Sahai, Akshayaram Srinivasan, and Mark Zhandry. Secure obfuscation in a weak multilinear map model. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B: 14th Theory of Cryptography Conference, Part II*, volume 9986 of *Lecture Notes in Computer Science*, pages 241–268. Springer, Heidelberg, October / November 2016.
- Hal15. Shai Halevi. Graded encoding, variations on a scheme. Cryptology ePrint Archive, Report 2015/866, 2015. <http://eprint.iacr.org/2015/866>.
- HJ16. Yupu Hu and Huiwen Jia. Cryptanalysis of GGH map. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 537–565. Springer, Heidelberg, May 2016.
- JW15. Zahra Jafargholi and Daniel Wichs. Tamper detection and continuous non-malleable codes. In Yevgeniy Dodis and Jesper Buus Nielsen, editors,

- TCC 2015: 12th Theory of Cryptography Conference, Part I*, volume 9014 of *Lecture Notes in Computer Science*, pages 451–480. Springer, Heidelberg, March 2015.
- Knu93. Lars R. Knudsen. Cryptanalysis of LOKI91. In Jennifer Seberry and Yuliang Zheng, editors, *Advances in Cryptology – AUSCRYPT’92*, volume 718 of *Lecture Notes in Computer Science*, pages 196–208. Springer, Heidelberg, December 1993.
- LMR14. Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Improved constructions of PRFs secure against related-key attacks. In Ioana Boureanu, Philippe Owesarski, and Serge Vaudenay, editors, *ACNS 14: 12th International Conference on Applied Cryptography and Network Security*, volume 8479 of *Lecture Notes in Computer Science*, pages 44–61. Springer, Heidelberg, June 2014.
- Men16. Bart Mennink. XPX: Generalized tweakable Even-Mansour with improved security guarantees. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 64–94. Springer, Heidelberg, August 2016.
- MSW14. Eric Miles, Amit Sahai, and Mor Weiss. Protecting obfuscation against arithmetic attacks. Cryptology ePrint Archive, Report 2014/878, 2014. <http://eprint.iacr.org/2014/878>.
- MSZ16. Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 629–658. Springer, Heidelberg, August 2016.
- NR97. Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th Annual Symposium on Foundations of Computer Science*, pages 458–467. IEEE Computer Society Press, October 1997.
- Sho97. Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT’97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266. Springer, Heidelberg, May 1997.
- WLZZ16. Peng Wang, Yuling Li, Liting Zhang, and Kaiyan Zheng. Related-key almost universal hash functions: Definitions, constructions and applications. In Thomas Peyrin, editor, *Fast Software Encryption – FSE 2016*, volume 9783 of *Lecture Notes in Computer Science*, pages 514–532. Springer, Heidelberg, March 2016.
- Zim15. Joe Zimmerman. How to obfuscate programs directly. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 439–467. Springer, Heidelberg, April 2015.