Adaptive Security of Constrained PRFs

Georg Fuchsbauer[1], Momchil Konstantinov[2], Krzysztof Pietrzak[1⋆], and Vanishree Rao[3]

[1] IST Austria, {gfuchsbauer,pietrzak}ist.ac.at
[2] London School of Geometry and Number Theory, UK
[3] UCLA, USA

Abstract. Constrained pseudorandom functions have recently been introduced independently by Boneh and Waters (Asiacrypt'13), Kiayias et al. (CCS'13), and Boyle et al. (PKC'14). In a standard pseudorandom function (PRF) a key K is used to evaluate the PRF on all inputs in the domain. Constrained PRFs additionally offer the functionality to delegate "constrained" keys K_S which allow to evaluate the PRF only on a subset S of the domain.

The three above-mentioned papers all show that the classical GGM construction (J.ACM'86) of a PRF from a pseudorandom generator (PRG) directly yields a constrained PRF where one can compute constrained keys to evaluate the PRF on all inputs with a given prefix. This constrained PRF has already found many interesting applications. Unfortunately, the existing security proofs only show selective security (by a reduction to the security of the underlying PRG). To achieve full security, one has to use complexity leveraging, which loses an exponential factor 2^N in security, where N is the input length.

The first contribution of this paper is a new reduction that only loses a quasipolynomial factor $q^{\log N}$, where q is the number of adversarial queries. For this we develop a new proof technique which constructs a distinguisher by interleaving simple guessing steps and hybrid arguments a small number of times. This approach might be of interest also in other contexts where currently the only technique to achieve full security is complexity leveraging.

Our second contribution is concerned with another constrained PRF, due to Boneh and Waters, which allows for constrained keys for the more general class of bit-fixing functions. Their security proof also suffers from a 2^N loss, which we show is inherent. We construct a meta-reduction which shows that any "simple" reduction of full security from a non-interactive hardness assumption must incur an exponential security loss.

Keywords: Constrained PRF, complexity leveraging, full security, meta-reduction.

1 Introduction

PRFs. Pseudorandom functions (PRFs) were introduced by Goldreich, Goldwasser and Micali [GGM86]. A PRF is an efficiently computable keyed function $\mathsf{F}\colon \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, where $\mathsf{F}(K, \cdot)$, instantiated with a random key $K \xleftarrow{*} \mathcal{K}$, cannot be distinguished from a function randomly chosen from the set of all functions $\mathcal{X} \to \mathcal{Y}$ with non-negligible probability.

Constrained PRFs. Recently, the notion of constrained PRFs (CPRFs) was introduced independently by Boneh and Waters [BW13], Boyle, Goldwasser and Ivan [BGI14] and Kiayias, Papadopoulos, Triandopoulos and Zacharias [KPTZ13].[4]

A constrained PRF is defined with respect to a set system $\mathcal{S} \subseteq 2^{\mathcal{X}}$ and supports the functionality to "delegate" (short) keys that can only be used to evaluate the function $\mathsf{F}\colon \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ on inputs specified by a subset $S \in \mathcal{S}$. Concretely, there is a "constrained" keyspace $\mathcal{K}_c$ and additional algorithms $\mathsf{F.constrain}\colon \mathcal{K} \times \mathcal{S} \to \mathcal{K}_c$ and $\mathsf{F.eval}\colon \mathcal{K}_c \times \mathcal{X} \to \mathcal{Y}$, which for all $K \in \mathcal{K}$, $S \in \mathcal{S}$, $x \in S$ and $K_S \leftarrow \mathsf{F.constrain}(K, S)$, satisfy $\mathsf{F.eval}(K_S, x) = \mathsf{F}(K, x)$ if $x \in S$ and $\mathsf{F.eval}(K_S, x) = \perp$ otherwise.

⋆ Research supported by ERC starting grant (259668-PSPC)

[4] The name "constrained PRF" is from [BW13]; in [KPTZ13] and [BGI14] these objects are called "delegatable PRFs" and "functional PRFs", respectively. In this paper we follow the naming and notation of [BW13].

The GGM and the Boneh-Waters construction. All three papers [BW13,BGI14,KPTZ13] show that the classical GGM construction [GGM86] of the PRF $\mathsf{GGM}\colon \{0,1\}^\lambda \times \{0,1\}^N \to \{0,1\}^\lambda$ from a length-doubling pseudorandom generator (PRG) $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ directly yields a constrained PRF, where for any key K and input prefix $z \in \{0,1\}^{\leq N}$, one can generate a constrained key K_z that allows to evaluate $\mathsf{GGM}(K,x)$ for any x with prefix z. This simple constrained PRF has found many applications; apart from those discussed in [BW13,BGI14,KPTZ13], it can be used to construct so-called "punctured" PRFs, which are a key ingredient in almost all the recent proofs of indistinguishability obfuscation [SW14,BCPR13,HSW14].

Boneh and Waters [BW13] construct a constrained PRF for a much more general set of constraints, where one can delegate keys that fix any subset of bits of the input (not just the prefix, as in GGM). The construction is based on leveled multilinear maps [GGH13] and its security is proven under a generalization of the decisional Diffie-Hellman assumption.

Security of constrained PRFs. The security definition for standard PRFs is quite intuitive. One considers two experiments: the "real" experiment and the "random" experiment, in both of which an adversary A gets access to an oracle $\mathcal{O}(\cdot)$ and then outputs a bit. In the real experiment $\mathcal{O}(\cdot)$ implements the PRF $\mathsf{F}(K,\cdot)$ using a random key; in the random experiment $\mathcal{O}(\cdot)$ implements a random function. The PRF is secure if every efficient A outputs 1 in both experiments with (almost) the same probability.

Defining the security of constrained PRFs requires a bit more thought. We want to give an adversary access not only to $\mathsf{F}(K,\cdot)$, but also to the constraining function $\mathsf{F.constrain}(K,\cdot)$. But now we cannot expect the values $\mathsf{F}(K,\cdot)$ to look random, as an adversary can always ask for a key $K_S \leftarrow \mathsf{F.constrain}(K,S)$ and then for any $x \in S$ check whether $\mathsf{F}(K,x) = \mathsf{F.eval}(K_S,x)$.

Instead, security is formalized by defining the experiments in two phases. In the first phase of both experiments the adversary gets access to the same oracles $\mathsf{F}(K,\cdot)$ and $\mathsf{F.constrain}(K,\cdot)$. The experiments differ in a second phase, where the adversary chooses some challenge query x^*. In the real experiment the adversary then obtains $\mathsf{F}(K,x^*)$, whereas in the random experiment she gets a random value. Intuitively, when no efficient adversary can distinguish these two games, this means that the outputs of $\mathsf{F}(K,\cdot)$ look random on all points that the adversary cannot compute by herself using the constrained keys she has received so far.

Selective vs. full security. In the above definition we let the adversary choose the challenge input x^* after she gets access to the oracles. This is the notion typically considered, and it is called "full security" or "adaptive security". One can also consider a weaker "selective security" notion, where the adversary must choose x^* before getting access to the oracles.

The reason to consider selective security notions, also for other objects like identity-based encryption [BF01,BB04,AFL12], is that it is often much easier to achieve. Although there exists a simple generic technique called "complexity leveraging", which translates any selective security guarantee into a security bound for full security, this technique (which really just consists of guessing the challenge) typically loses an exponential factor (in the length of the challenge) in the quality of the reduction, often making the resulting security guarantee meaningless for practical parameters.

1.1 Our Contributions

All prior works [BW13,BGI14,KPTZ13] only show selective security of the GGM constrained PRF, and [BW13] also only give a selective-security proof for their bit-fixing constrained PRF. In this paper we investigate the full security of these two constructions. For GGM we achieve a positive result, giving a reduction that only loses a quasipolynomial factor. For the Boneh-Waters bit-fixing CPRF we give a negative result, showing that for a large class of reductions, an exponential loss is necessary. We now elaborate on these results.

A quasipolynomial reduction for GGM. To prove full security of $\mathsf{GGM}\colon \{0,1\}^\lambda \times \{0,1\}^N \to \{0,1\}^\lambda$, the "standard" proof proceeds in two steps (we give a precise statement in Proposition 3).

1. A guessing step (a.k.a. complexity leveraging), which reduces full to selective security. This step loses an exponential factor 2^N in the input length N.
2. Now one applies a hybrid argument which loses a factor $2N$.

Readers not familiar with hybrid arguments can find a simple application of this technique in Appendix A.

The above two steps transform an adversary $\mathsf{A_f}$ that breaks the full security of GGM with advantage ϵ into a new adversary that breaks the security of the underlying pseudorandom generator G (used to construct the GGM function) with advantage $\epsilon/(2N \cdot 2^N)$. As a consequence, even if one makes a strong exponential hardness assumption on the PRG G, one must use a PRG whose domain is $\Theta(N)$ bits in order to get any meaningful security guarantee.

The reason for the huge security loss is the first step, in which one guesses the challenge $x^* \in \{0,1\}^N$ the adversary will choose, which is correct with probability 2^{-N}. To avoid this exponential loss, one must thus avoid guessing the entire x^*. Our new proof also consists of a guessing step followed by a hybrid argument.

1. A guessing step, where (for some ℓ) we guess which of the adversary's queries will be the *first* one that agrees with x^* in the first ℓ positions.[5] This step loses a factor q, which denotes the number of queries made by the adversary.
2. A hybrid argument which loses a constant factor 3.

The above two steps only lose a factor $3q$. Unfortunately, after one iteration of this approach we do not get a distinguisher for G right away. At a high level, these two steps achieve the following: We start with two games which in some sense are at distance N from each other, and we end up with two games which are at distance $N/2$. We can iterate the above process $n := \log N$ times to end up with games at distance $N/2^n = 1$. Finally, from any distinguisher for games at distance 1 we can get a distinguisher for the PRG G with the same advantage. Thus, starting from an adversary against the full security of GGM with advantage ϵ, we get a distinguisher for the PRG with advantage $\epsilon/(3q)^{\log N}$.

We can optimize this by combining this approach with the original proof, and therby obtain a quasipolynomial loss of $2q \log q \cdot (3q)^{\log N - \log\log q}$. To give some numerical example, let the input length be $N = 2^{10} = 1024$ and the number of queries be $q = 2^{32}$. Then we get a loss of $2q \log q \cdot (3q)^{\log N - \log\log q} = 2 \cdot 2^{32} \cdot 32 \cdot (3 \cdot 2^{32})^{10-5} = 2^{198} \cdot 3^5 \leq 2^{206}$, whereas complexity leveraging loses $2N2^N = 2^{1035}$.

Although our proof is somewhat tailored to the GGM construction, the general "fine-grained" guessing approach outlined above might be useful to improve the bounds for other constructions (like CPRFs, and even IBE schemes) where currently the only proof technique that can be applied is complexity leveraging.

A lower bound for the Boneh-Waters CPRF and Hofheinz's construction. We then turn our attention to the bit-fixing constrained PRF by Boneh and Waters [BW13]. For this construction too, complexity leveraging—losing an exponential factor—is the only known technique to prove full security. We give strong evidence that this is inherent (even when the construction is only used as a prefix-fixing CPRF).

Concretely, we prove that every "simple" reduction (which runs the adversary once without rewinding; see Section 5.2) of the full security of this scheme from any decisional (and thus also search) assumption

[5] This guessing is somewhat reminiscent of a proof technique from [HW09].

must lose an exponential factor. Our proof is a so-called meta-reduction [BV98,Cor02,FS10], showing that any reduction that breaks the underlying assumption when given access to any adversary that breaks the CPRF, could be used to break the underlying assumption without the help of an adversary.

This impossibility result is similar to existing results, the closest one being a result of Lewko and Waters [LW14] ruling out security proofs without exponential loss for so-called "prefix-encryption" schemes (which satisfy some special properties). Other related results are those of Coron [Cor02] and Hofheinz et al. [HJK12], which show that security reductions for certain signature schemes must lose a factor polynomial in the number of signing queries.

The above impossibility proofs are for public-key objects, where a public key uniquely determines the input/output distribution of the object. This property is crucially used in the proof, wherein one first gets the public key and then runs the reduction, rewinding the reduction multiple times to the point right after the public key has been received.

As we consider a secret-key primitive, the above approach seems inapplicable. We overcome this by observing that for the Boneh-Waters CPRF we can initially make some fixed "fingerprint" queries, which then uniquely determine the remaining outputs. We then use the responses to these fingerprint queries instead of a public key as in [LW14].

Hofheinz [Hof14] has (independently and concurrently with us) investigated the adaptive security of bit-fixing constrained PRFs. He gives a new construction of such PRFs which is more sophisticated than the Boneh-Waters construction, and for which he can give a security reduction that only loses a polynomial factor. The main tool that allows Hofheinz to overcome our impossibility result is the use of a random oracle $H(\cdot)$. Very informally, instead of evaluating the PRF on an input X, it is evaluated on $H(X)$ which forces an attacker to make every query X explicit. Unfortunately, this idea does not work directly as it destroys the structure of the preimages, and thus makes the construction of short delegatable keys impossible. Hofheinz deals with this problem using several other ideas.

2 Preliminaries

For $a \in \mathbb{N}$, we let $[a] := \{1, 2, \ldots, a\}$ and $[a]_0 := \{0, 1, \ldots, a\}$. By $\{0,1\}^{\leq a} = \bigcup_{i \leq a} \{0,1\}^i$ we denote the set of bitstrings of length at most a, including the empty string $\emptyset$. By U_a we denote the random variable with uniform distribution over $\{0,1\}^a$. We denote sampling s uniformly from a set $\mathcal{S}$ by $s \stackrel{*}{\leftarrow} \mathcal{S}$. We let $x \| y$ denote the concatenation of the bitstrings x and y. For sets $\mathcal{X}, \mathcal{Y}$, we denote by $\mathcal{F}[\mathcal{X}, \mathcal{Y}]$ the set of all functions $\mathcal{X} \to \mathcal{Y}$; moreover, $\mathcal{F}[a, b]$ is short for $\mathcal{F}[\{0,1\}^a, \{0,1\}^b]$. For $x \in \{0,1\}^*$, we denote by x_i the i-th bit of x, and by $x[i \ldots j]$ the substring $x_i \| x_{i+1} \| \ldots \| x_j$.

Definition 1 (Indistinguishability). *Two distributions X and Y are (ϵ, s)-**indistinguishable**, denoted $X \sim_{(\epsilon,s)} Y$, if no circuit D of size at most s can distinguish them with advantage greater than ϵ, i.e.,*

$$X \sim_{(\epsilon,s)} Y \iff \forall \mathsf{D}, |\mathsf{D}| \leq s : \left| \Pr[\mathsf{D}(X) = 1] - \Pr[\mathsf{D}(Y) = 1] \right| \leq \epsilon \ .$$

$X \sim_\delta Y$ denotes that the statistical distance of X and Y is δ (i.e., $X \sim_{(\delta,\infty)} Y$), and $X \sim Y$ denotes that they have the same distribution.

Definition 2 (PRG). *An efficient function $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ is an (ϵ, s)-secure (length-doubling)* **pseudorandom generator** *(PRG) if*

$$\mathsf{G}(U_\lambda) \sim_{(\epsilon,s)} U_{2\lambda} \ .$$

Definition 3 (PRF). *A keyed function* $\mathsf{F}\colon \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ *is an* (ϵ, s, q)*-secure* **pseudorandom function** *if for all adversaries* A *of size at most* s *making at most* q *oracle queries*

$$\left| \Pr_{K \xleftarrow{*} \mathcal{K}}[\mathsf{A}^{\mathsf{F}(K,\cdot)} \to 1] - \Pr_{f \xleftarrow{*} \mathcal{F}[\mathcal{X},\mathcal{Y}]}[\mathsf{A}^{f(\cdot)} \to 1] \right| \le \epsilon \ .$$

Constrained pseudorandom functions. Following [BW13], we say that a function $\mathsf{F}\colon \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is a *constrained PRF* for a set system $\mathcal{S} \subseteq 2^{\mathcal{X}}$, if there is a *constrained-key space* $\mathcal{K}_c$ and algorithms

$$\mathsf{F.constrain}\colon \mathcal{K} \times \mathcal{S} \to \mathcal{K}_c \quad \text{and} \quad \mathsf{F.eval}\colon \mathcal{K}_c \times \mathcal{X} \to \mathcal{Y} \ ,$$

which for all $K \in \mathcal{K}$, $S \in \mathcal{S}$, $x \in S$ and $K_S \leftarrow \mathsf{F.constrain}(K, S)$ satisfy

$$\mathsf{F.eval}(K_S, x) = \begin{cases} \mathsf{F}(K, x) & \text{if } x \in S \\ \perp & \text{otherwise} \end{cases} .$$

That is, $\mathsf{F.constrain}(K, S)$ outputs a key K_S that allows evaluation of $\mathsf{F}(K, \cdot)$ on all $x \in S$.

Informally, a constrained PRF F is secure, if no efficient adversary can distinguish $\mathsf{F}(K, x^*)$ from random, even given access to $\mathsf{F}(K, \cdot)$ and $\mathsf{F.constrain}(K, \cdot)$ which he can query on all $x \neq x^*$ and $S \in \mathcal{S}$ where $x^* \notin S$, respectively. We will always assume that $\mathcal{S}$ contains all singletons, i.e., $\forall\, x \in \mathcal{X} : \ \{x\} \in \mathcal{S}$; this way we do not have to explicitly give the adversary access to $\mathsf{F}(K, \cdot)$, as $\mathsf{F}(K, x)$ can be learned by querying for $K_x \leftarrow \mathsf{F.constrain}(K, \{x\})$ and computing $\mathsf{F.eval}(K_x, x)$.

We distinguish between selective and full security. In the selective security game the adversary must choose the challenge x^* before querying the oracles. Both games are parametrized by the maximum number q of queries the adversary makes, of which the last query is the challenge query.

$\mathbf{Exp}^{\mathsf{sel}}_{\mathsf{CPRF}}(\mathsf{A}, \mathsf{F}, b, q)$	$\mathbf{Exp}^{\mathsf{full}}_{\mathsf{CPRF}}(\mathsf{A}, \mathsf{F}, b, q)$	**Oracle** $\mathcal{O}(S)$
$K \xleftarrow{*} \mathcal{K}$, $\hat{S} := \emptyset$, $c := 0$ $x^* \leftarrow \mathsf{A}$ $\mathsf{A}^{\mathcal{O}(\cdot)}$ $C_0 \xleftarrow{*} \mathcal{Y}$, $C_1 := \mathsf{F}(K, x^*)$ A gets C_b $\tilde{b} \leftarrow \mathsf{A}$ if $x^* \in \hat{S}$, return 0 return $\tilde{b}$	$K \xleftarrow{*} \mathcal{K}$, $\hat{S} := \emptyset$, $c := 0$ $\mathsf{A}^{\mathcal{O}(\cdot)}$ $x^* \leftarrow \mathsf{A}$ $C_0 \xleftarrow{*} \mathcal{Y}$, $C_1 := \mathsf{F}(K, x^*)$ A gets C_b $\tilde{b} \leftarrow \mathsf{A}$ if $x^* \in \hat{S}$, return 0 return $\tilde{b}$	if $c = q - 1$, return $\perp$ $c := c + 1$ $\hat{\mathcal{S}} := \hat{\mathcal{S}} \cup S$ $K_S \leftarrow \mathsf{F.constrain}(K, S)$ return K_S

For $\mathsf{atk} \in \{\mathsf{sel}, \mathsf{full}\}$ we define A's advantage as

$$\mathsf{Adv}^{\mathsf{atk}}_{\mathsf{F}}(\mathsf{A}, q) = 2 \left| \Pr_{b \xleftarrow{*} \{0,1\}}[\mathbf{Exp}^{\mathsf{atk}}_{\mathsf{CPRF}}(\mathsf{A}, \mathsf{F}, b, q) = b] - \tfrac{1}{2} \right| \tag{1}$$

and denote with

$$\mathsf{Adv}^{\mathsf{atk}}_{\mathsf{F}}(s, q) = \max_{\mathsf{A}, |\mathsf{A}| \le s} \mathsf{Adv}^{\mathsf{atk}}_{\mathsf{F}}(\mathsf{A}, q)$$

the advantage of the best q-query adversary of size at most s.

Definition 4 (Selective and full security of CPRFs). *A constrained PRF* F *is*

- **selectively** (ϵ, s, q)**-secure** *if* $\mathsf{Adv}^{\mathsf{sel}}_{\mathsf{F}}(s, q) \le \epsilon$ *and*
- **fully** (ϵ, s, q)**-secure** *if* $\mathsf{Adv}^{\mathsf{full}}_{\mathsf{F}}(s, q) \le \epsilon$.

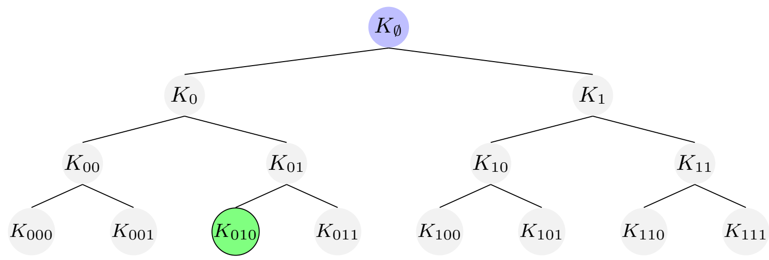

Fig. 1. Illustration of the GGM PRF. Every left child $K_{z\|0}$ of a node K_z is defined as the first half of $\mathsf{G}(K_z)$, the right child $K_{z\|1}$ as the second half. The circled node corresponds to $\mathsf{GGM}(K_\emptyset, 010)$.

Remark 1 (CCA1 vs. CCA2 security). In the selective and full security notion, we assume that the challenge query x^* is only made at the very end, when A has no longer access to the oracle (this is reminiscent of CCA1 security). All our positive results hold for stronger notions (reminiscent to CCA2 security) where A still has access to $\mathcal{O}(\cdot)$ after making the challenge query, but may not query it on any S where $x^* \in S$.

Remark 2 (Multiple challenge queries). We only allow the adversary one challenge query. As observed in [BW13], this implies security against any $t > 1$ challenge queries, losing a factor of t in the distinguishing advantage, by a standard hybrid argument.

Using what is sometimes called "complexity leveraging", one can show that selective security implies full security: given an adversary A against full security, we construct a selective adversary B, which at the beginning *guesses* a challenge x^* and outputs it, then runs A and aborts if the challenge that A eventually outputs is different from x^*. The distinguishing advantage drops thus by a factor of the domain size $|\mathcal{X}|$. The following is proved in Appendix B.1.

Lemma 1 (Complexity leveraging). *If a constrained PRF* $\mathsf{F}\colon \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ *is* (ϵ, s, q)-**selectively** *secure then it is* $(\epsilon|\mathcal{X}|, s', q)$-**fully** *secure (where* $s' = s - O(\log|\mathcal{X}|)$*), i.e.,*

$$\mathsf{Adv}^{\mathsf{full}}_{\mathsf{F}}(s', q) \leq |\mathcal{X}| \cdot \mathsf{Adv}^{\mathsf{sel}}_{\mathsf{F}}(s, q) \ .$$

3 The GGM Construction

The GGM construction, named after its inventors Goldreich, Goldwasser and Micali [GGM86], is a keyed function $\mathsf{GGM}^{\mathsf{G}}\colon \{0,1\}^\lambda \times \{0,1\}^* \to \{0,1\}^\lambda$ defined by any length-doubling pseudorandom generator $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ recursively as

$$\mathsf{GGM}(K_\emptyset, X) = K_X \ , \quad \text{where} \quad \forall\, Z \in \{0,1\}^{\leq N-1} : \ K_{Z\|0} \| K_{Z\|1} = \mathsf{G}(K_Z) \tag{2}$$

(cf. Figure 1). In [GGM86] it is shown that when the inputs are restricted to $\{0,1\}^N$ then $\mathsf{GGM}^{\mathsf{G}}$ is a secure PRF if G is a secure PRG. Their proof is one of the first applications of the so-called hybrid argument.[6] The proof loses a factor of $q \cdot N$ in distinguishing advantage (where q is the number of queries).

Proposition 1 (GGM is a PRF [GGM86]). *If* $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ *is an* $(\epsilon_{\mathsf{G}}, s_{\mathsf{G}})$*-secure PRG then (for any* N, q*)* $\mathsf{GGM}^{\mathsf{G}}\colon \{0,1\}^\lambda \times \{0,1\}^N \to \{0,1\}^\lambda$ *is an* (ϵ, s, q)*-secure PRF with*

$$\epsilon = \epsilon_{\mathsf{G}} \cdot q \cdot N \quad \text{and} \quad s = s_{\mathsf{G}} - O(q \cdot N \cdot |\mathsf{G}|) \ .$$

[6] The first application is in the "probabilistic encryption" paper [GM84].

We will not give a proof of this proposition here; however it follows from Proposition 2 below, for which we give a proof sketch. In his book [Gol01] Goldreich presents several generalizations of GGM, including a variant which is secure even if we allow the entire domain $\{0,1\}^*$ as inputs. Here, we'd like to mention that the original GGM construction is a secure "prefix-free" PRF as defined below. The reason for presenting this variant of GGM here is so we can later, in Remark 3, discuss why this variant of GGM does *not* already imply security of GGM as a constrained PRF. Instead of the number q of points queried by the adversary, the security of a prefix-free PRF with domain $\{0,1\}^*$ is parametrized by the sum m of the bitlengths of all queries.

Definition 5 (PF-PRF). *A keyed function* $\mathsf{F}\colon \mathcal{K} \times \{0,1\}^* \to \mathcal{Y}$ *is an* (ϵ, s, m)*-secure* **prefix-free pseudorandom function** *(PF-PRF) if for all adversaries* A *of size at most* s *making queries of total bitlength at most* m*, but where no query can be a prefix of another query,*

$$\left| \Pr_{K \xleftarrow{*} \mathcal{K}} [\mathsf{A}^{\mathsf{F}(K,\cdot)} \to 1] - \Pr_{f \xleftarrow{*} \mathcal{F}[\{0,1\}^*, \mathcal{Y}]} [\mathsf{A}^{f(\cdot)} \to 1] \right| \leq \epsilon \ .$$

Proposition 2 (GGM is a PF-PRF). *If* $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ *is an* $(\epsilon_\mathsf{G}, s_\mathsf{G})$*-secure PRG then (for any* m*)* $\mathsf{GGM}^\mathsf{G}\colon \{0,1\}^\lambda \times \{0,1\}^* \to \{0,1\}^\lambda$ *is an* (ϵ, s, m)*-secure PF-PRF with*

$$\epsilon = \epsilon_\mathsf{G} \cdot m \quad \textit{and} \quad s = s_\mathsf{G} - O(m \cdot |\mathsf{G}|) \ .$$

We prove this proposition in Appendix B.2. Note that if we drop the restriction that queries must be prefix-free, the construction is trivially insecure, as from $y = \mathsf{GGM}^\mathsf{G}(K, x)$ one can compute $y' = \mathsf{GGM}^\mathsf{G}(K, x \| z)$ for any z, thus y' is not pseudorandom given y. Also note that restricting the domain to $\{0,1\}^N$, we get $m = q \cdot N$, as stated in Proposition 1.

3.1 GGM is a Constrained PRF

As observed recently by three different works independently [BW13,BGI14,KPTZ13], the GGM construction can be used as a constrained PRF for the set system $\mathcal{S}_\mathsf{pre}$ of sets of inputs with common prefixes, defined as

$$\mathcal{S}_\mathsf{pre} = \{S_p \ : \ p \in \{0,1\}^{\leq N}\} \ , \quad \text{where} \quad S_p = \{p \| z \ : \ z \in \{0,1\}^{N-|p|}\} \ .$$

Thus, given a key K_p for the set S_p, one can evaluate $\mathsf{GGM}^\mathsf{G}(K, \cdot)$ on all inputs with prefix p. Formally, the constrained PRF with key $K = K_\emptyset$ is defined using (2) as follows:

$$\mathsf{GGM}^\mathsf{G}.\mathsf{constrain}(K_\emptyset, p) = \mathsf{GGM}^\mathsf{G}(K_\emptyset, p) = K_p \quad \text{and} \quad \mathsf{GGM}^\mathsf{G}.\mathsf{eval}(K_p, x = p\|z) = \mathsf{GGM}^\mathsf{G}(K_p, z) = K_x \ .$$

An interior node K_p in Figure 1 is thus the constrained key for the set S_p.

Remark 3. One might be tempted to think that the fact that GGM is a PF-PRF (Proposition 2), together with the fact that constrained-key derivation is simply the GGM function itself, already implies that it is a secure constrained PRF. Unfortunately, this is not sufficient, as the (selective and full) security notions for constrained PRFs do allow queries that are prefixes of previous queries.

The *selective* security of this construction can be proven using a standard hybrid argument, losing only a factor of $2N$ in the distinguishing advantage (see Proposition 3 below). Proving full security seems much more challenging, and prior to our work it was only achieved by complexity leveraging (cf. Lemma 1), which loses an additional exponential factor 2^N in the distinguishing advantage, as stated in Proposition 3 below.

Remark 4. In the proof of Proposition 3 and Theorem 1 we will slightly cheat, as in the security game when $b = 0$ (i.e., when the challenge output is random) we not only replace the challenge output K_{x^*}, but also its sibling $K_{x^*[1\ldots N-1]\overline{x}^*_N}$, with a random value. Thus, technically this only proves security for inputs of length $N-1$ (as we can e.g. simply forbid queries $x\|0, x \in \{0,1\}^{N-1}$, in which case it is irrelevant what the sibling is, as it will never be revealed). The proofs without this cheat require one extra hybrid, which requires a somewhat different treatment than all others hybrids and thus would complicate certain proofs and definitions. Hence, we chose to not include it. The bounds stated in Proposition 3 and Theorem 1 are the bounds we get *without* this cheat.

Proposition 3. *If* $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ *is an* $(\epsilon_\mathsf{G}, s_\mathsf{G})$*-secure PRG then (for any* N, q*)* $\mathsf{GGM}^\mathsf{G}\colon \{0,1\}^N \to \{0,1\}^\lambda$ *is a constrained PRF for* $\mathcal{S}_\mathsf{pre}$ *which is*

1. **selectively** (ϵ, s, q)*-secure, with*

$$\epsilon = \epsilon_\mathsf{G} \cdot 2N \quad \textit{and} \quad s = s_\mathsf{G} - O(q \cdot N \cdot |\mathsf{G}|) \ ;$$

2. **fully** (ϵ, s, q)*-secure, with*

$$\epsilon = \epsilon_\mathsf{G} \cdot 2^N 2N \quad \textit{and} \quad s = s_\mathsf{G} - O(q \cdot N \cdot |\mathsf{G}|) \ .$$

Full security as stated in Item *2.* of the proposition follows from selective security (Item *1.*) by complexity leveraging as explained in Lemma 1. To prove selective security, we let H_0 be the real game for selective security and let H_{2N-1} be the random game, that is, where K_{x^*} is random. We then define intermediate hybrid games $H_1, \ldots, H_{2N-2}$ by embedding random values along the path to K_{x^*}. (See Figure 5 in Appendix A for an illustration.) In particular, in hybrid H_i, for $1 \le i \le N$, the nodes $K_\emptyset, K_{x^*_1}, \ldots, K_{x^*[1\ldots i]}$ are random and for $N+1 \le i \le 2N-1$ the nodes $K_\emptyset, K_{x^*_1}, \ldots, K_{x^*[1\ldots 2N-1-i]}$ and K_{x^*} are random. Thus two consecutive games H_i, H_{i+1} differ in one node that is real in one game and random in the other, and the parent of that node is random, meaning we can embed a PRG challenge. From any distinguisher for two consecutive games we thus get a distinguisher for the PRG G with the same advantage. (See Appendix B.3 for a formal proof.)

This hybrid argument only loses a factor $2N$ in distinguishing advantage, but complexity leveraging loses a huge factor 2^N. In the next section we show how to prove full security avoiding such an exponential loss.

4 Full Security with Quasipolynomial Loss

Theorem 1. *If* $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ *is an* $(\epsilon_\mathsf{G}, s_\mathsf{G})$*-secure PRG then (for any* N, q*)* $\mathsf{GGM}^\mathsf{G}\colon \{0,1\}^N \to \{0,1\}^\lambda$ *is a* **fully** (ϵ, s, q)*-secure constrained PRF for* $\mathcal{S}_\mathsf{pre}$*, where*

$$\epsilon = \epsilon_\mathsf{G} \cdot (3q)^{\log N} \quad \textit{and} \quad s = s_\mathsf{G} - O(q \cdot N \cdot |\mathsf{G}|) \ .$$

At the end of this section we will sketch how to combine the proof of this theorem with the standard complexity leveraging proof from Proposition 3 to get a smaller loss of $\epsilon = \epsilon_\mathsf{G} \cdot 2q \log q \cdot (3q)^{\log N - \log\log q}$.

Proof idea. We can view the real and the random game for CPRF security as having distance N, in the sense that from the only node in which they differ (which is the challenge node K_{x^*}) we have to walk up N nodes until we reach a node that was chosen uniformly at random (which here is the root $K_\emptyset$).

As outlined in Section 1.1, our goal is to halve that distance. For this, we could define two intermediate hybrids which are defined as the real and the random games, except that the node half way down the path to x^*, i.e., $K_{x^*[1...N/2]}$, is a random node. This is illustrated in Figure 2, where a row depicts the path from the root (labeled '0') to x^* (labeled '8') and where blue nodes correspond to random values. The path at the top of the figure is the real game and the one at the bottom is the random game (ignore anything in the boxes for now), and the intermediate hybrids are the 2nd and the 3rd path. Of these 4 hybrids, each pair of consecutive hybrids has the following property: they differ in one node and its distance to the closest random node above is $N/2$.

There is a problem with this approach because the intermediate hybrid games we have just constructed are not even well-defined, as the value $x^*[1 \dots N/2]$ is only known when the adversary makes his challenge query. This is also the case for x^* itself, but K_{x^*} only needs to be computed once x^* is queried; in contrast, $K_{x^*[1...N/2]}$ could have been computed earlier in the game, if the value of some constrained-key query is a descendant of it. In order to avoid possible inconsistencies, we do the following: we guess which of the adversary's queries will be the first one with a prefix $x^*[1 \dots N/2]$. As there are at most q queries and there always exists a query with this property (at latest the challenge query itself), the probability of guessing correctly is $1/q$. If we guess correctly then the node $x^*[1 \dots N/2]$ is known precisely when the value $K_{x^*[1...N/2]}$ is computed for the first time and we can correctly simulate the game. If our guess was wrong, we abort.

Assuming an attacker can distinguish between the real and the random game, there must be two consecutive hybrids of the 4 hybrids that it can distinguish with at least one third of his original advantage. Between these two hybrids, which differ in one node d, we can then embed two new intermediate hybrids, which have a random value half way between d and the closest random node above (cf. the outer box in Figure 2). We continue to do so until we reach two hybrids where there is a random node immediately above the differing node. A distinguisher between two such games can then be used to break the PRG.

Neighboring sets with low weight. Before starting with the proof, we introduce some notation. It will be convenient to work with ternary numbers, which we represent as strings of digits from $\{0, 1, 2\}$ within angular brackets $\langle \dots \rangle$. We denote repetition of digits as $0_n = 0 \dots 0$ (n times). Addition will also be in ternary, e.g., $\langle 202 \rangle + \langle 1 \rangle = \langle 210 \rangle$.

Let $N = 2^n$ be a power of 2. In the proof of Theorem 1 we will construct $3^n + 1$ subsets $\mathcal{S}_{\langle 0 \rangle}, \dots, \mathcal{S}_{\langle 10_n \rangle} \subset \{0, \dots, N\}$. These sets will define the positions in the path to the challenge where we make random guesses in a particular hybrid. The following definition measures how "close" sets (that differ in one element) are and will be useful in defining neighboring hybrids.

Definition 6 (Neighboring sets). *For $k \in \mathbb{N}^+$, two sets $\mathcal{S}, \mathcal{S}' \subset \mathbb{N}^0$ are k-**neighboring** if*

1. $\mathcal{S} \,\Delta\, \mathcal{S}' := (\mathcal{S} \cup \mathcal{S}') \setminus (\mathcal{S} \cap \mathcal{S}') = \{d\}$ *for some $d \in \mathbb{N}^0$, i.e., they differ in exactly one element d.*
2. $d - k \in \mathcal{S}$.
3. $\forall\, i \in [k-1] : \, d - i \notin \mathcal{S}$.

We define the first set (with index $0 = \langle 0 \rangle$) and the and last set (with index $3^n = \langle 10_n \rangle$) as

$$\mathcal{S}_{\langle 0 \rangle} := \{0\} \quad \text{and} \quad \mathcal{S}_{\langle 10_n \rangle} := \{0, N\} \ . \tag{3}$$

(They will correspond to the real game, where only the root at depth '0' is random, and the random game, where the value for x^* at depth N is random too.) The remaining intermediate sets are defined recursively as follows. For $\ell = 0, \dots, n$, we define the ℓ-th level of sets to be all the sets of the form $\mathcal{S}_{\langle ?0_{n-\ell} \rangle}$ (i.e., whose index in ternary ends with $(n - \ell)$ zeros). Thus, $\mathcal{S}_{\langle 0 \rangle}$ and $\mathcal{S}_{\langle 10_n \rangle}$ are the (only) level-0 sets.

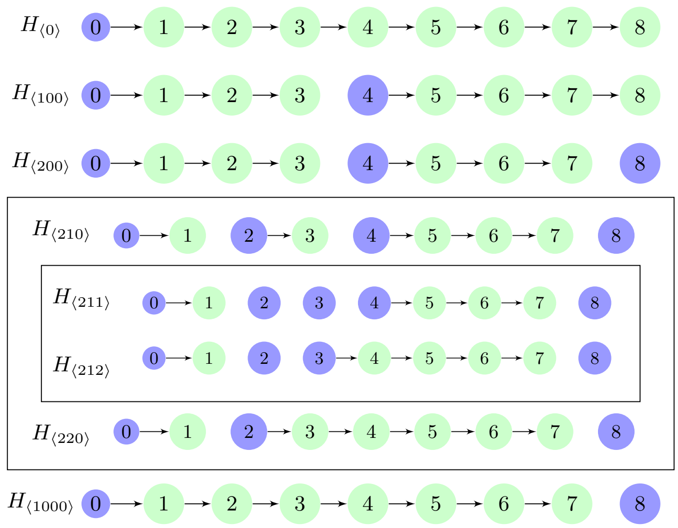

Fig. 2. Concrete example ($n = 3$) illustrating the iterative construction of hybrids in Theorem 1.

Let $\mathcal{S}_I, \mathcal{S}_{I'}$ be two consecutive level-ℓ sets, by which we mean that $I' = I + \langle 10_{n-\ell}\rangle$. By construction, these sets will differ in exactly one element $\{d\}$ (i.e., $\mathcal{S}_I \neq \mathcal{S}_{I'}$; and $\mathcal{S}_I \cup \{d\} = \mathcal{S}_{I'}$ or $\mathcal{S}_{I'} \cup \{d\} = \mathcal{S}_I$). Then the two level-$(\ell+1)$ sets between the level-ℓ sets $\mathcal{S}_I, \mathcal{S}_{I'}$ are defined as

$$\mathcal{S}_{I+\langle 10_{n-(\ell+1)}\rangle} := \mathcal{S}_I \cup \{d - \tfrac{N}{2^{\ell+1}}\} \quad \text{and} \quad \mathcal{S}_{I'-\langle 10_{n-(\ell+1)}\rangle} := \mathcal{S}_{I'} \cup \{d - \tfrac{N}{2^{\ell+1}}\} \ . \tag{4}$$

A concrete example for $N = 2^n = 2^3 = 8$ is illustrated in Figure 2 (where the blue nodes of H_I correspond to $\mathcal{S}_I$).

An important fact we will use is that consecutive level-ℓ sets are $(N/2^\ell)$-neighboring (see Definition 6); in particular, consecutive level-n sets (the 4 lines in the box in Figure 2 illustrate 4 consecutive sets) are thus 1-neighboring, i.e.,

$$\forall\, I \in \{\langle 0\rangle, \ldots, \langle 2_n\rangle\} : \ \mathcal{S}_I \,\Delta\, \mathcal{S}_{I+\langle 1\rangle} = \{d\} \quad \text{and} \quad d-1 \in \mathcal{S}_I \ . \tag{5}$$

Proof of Theorem 1. Below we prove two lemmata (2 and 3) concerning the games defined in Figure 3, from which the theorem follows quite immediately. As the games and the lemmata are rather technical, we first intuitively explain what is going on, going through a concrete example as illustrated in Figure 2.

To prove the theorem, we assume that there exists an adversary $\mathsf{A_f}$ that breaks the *full* security of $\mathsf{GGM^G}$ with some advantage ϵ, and from this, we want to construct a distinguisher for G with advantage at least $\epsilon/(3q)^n$, where $n = \log N$. Like in the proof of Proposition 3, we can think of the two games that $\mathsf{A_f}$ distinguishes as the games where we let $\mathsf{A_f}$ query $\mathsf{GGM^G}$, but along the path from the root $K_\emptyset$ down to the challenge K_{x^*} the nodes are either computed by G or they are random values. The position of the random values are defined by the set $\mathcal{S}_{\langle 0\rangle} = \{0\}$ for the real game and by $\mathcal{S}_{\langle 10_n\rangle} = \{0, N\}$ for the random game: in both cases the root $K_\emptyset$ is random, and in the latter game the final output K_{x^*} is also random. We call these two games $H^\emptyset_{\langle 0\rangle}$ and $H^\emptyset_{\langle 10_n\rangle}$, and they correspond to the games defined in Figure 3 with $\mathcal{P} = \emptyset$, and $I = \langle 0\rangle$ and $\langle 10_n\rangle$, respectively). As just explained, they satisfy

$$H^\emptyset_{\langle 0\rangle} \sim \mathbf{Exp}^{\mathsf{full}}_{\mathsf{CPRF}}(\mathsf{A_f}, \mathsf{GGM^G}, 0, q) \quad \text{and} \quad H^\emptyset_{\langle 10_n\rangle} \sim \mathbf{Exp}^{\mathsf{full}}_{\mathsf{CPRF}}(\mathsf{A_f}, \mathsf{GGM^G}, 1, q) \ .$$

Experiment $H_I^{\mathcal{P}}$

$\quad$ // $I \in \{\langle 0\rangle, \ldots, \langle 10_n \rangle\}$

$\quad$ // $\mathcal{P} = \{p_1, \ldots, p_t\} \subseteq \{1, \ldots, N-1\}$

$\quad$ // $\mathcal{S}_I \subseteq \mathcal{P} \cup \{0, N\}$, $\mathcal{S}_I$ as in eq. (3) and (4).

$\forall x \in \{0,1\}^{\leq N} : \ K_x := \perp$

$K_\emptyset \xleftarrow{*} \{0,1\}^\lambda$

$\quad$ // Initialize counters:

$\forall j = 1 \ldots N-1 : \ c_j = 0$

$\quad$ // Make a random guess for each

$\quad$ // element in $\mathcal{P} = \{p_1, \ldots, p_t\}$:

$\forall j \in [t] : \ q_{p_j} \xleftarrow{*} [q]$

$\quad$ // Below $\mathsf{A_f}$ can make exactly q distinct

$\quad$ // oracle queries $x_1, \ldots, x_q$. The last (chal-

$\quad$ // lenge) query $x_q = x^*$ must be in $\{0,1\}^N$.

$\mathsf{A_f}^{\mathcal{O}(\cdot)}$

$\tilde{b} \leftarrow \mathsf{A_f}$

$\quad$ // Only if guesses $q_{p_1}, \ldots, q_{p_t}$ were

$\quad$ // correct, return $\tilde{b}$, otherwise return 0:

If $\forall p \in \mathcal{P} : \ x^*[1 \ldots p-1] = z_{p-1}$ then

$\quad$ return $\tilde{b}$

Else return 0

$\mathcal{O}(x = x[1 \ldots \ell])$

$\quad$ // Return K_x if it is already defined:

if $K_x \neq \perp$ then return K_x

$\quad$ // Get parent of K_x recursively:

$K_{x[1\ldots\ell-1]} := \mathcal{O}(x[1 \ldots \ell - 1])$

$\quad$ // Increase counter for level $\ell - 1$:

$c_{\ell-1} = c_{\ell-1} + 1$

$\quad$ // Compute K_x and its sibling using G, unless its parent

$\quad$ // $K_{x[1\ldots\ell-1]}$ is a node which we guessed will be on the

$\quad$ // path from $K_\emptyset$ and K_{x^*} and as $\ell \in \mathcal{P}$ we must use a

$\quad$ // random value at this level; OR this is the challenge

$\quad$ // query $x_q = x^*$ and $N \in \mathcal{S}_I$, which means the answer

$\quad$ // to the challenge is random:

If ($\ell \in \mathcal{P}$ and $c_{\ell-1} = q_{\ell-1}$) OR ($x = x_q$ and $N \in \mathcal{S}_I$)

$\quad K_{x[1\ldots\ell-1]\|0} \| K_{x[1\ldots\ell-1]\|1} \xleftarrow{*} U_{2\lambda}$

$\quad\quad$ // Store this node to check if guess was correct later:

$\quad z_{\ell-1} = x[1 \ldots \ell - 1]$

else

$\quad K_{x[1\ldots\ell-1]\|0} \| K_{x[1\ldots\ell-1]\|1} := \mathsf{G}(K_{x[1\ldots\ell-1]})$

Return K_x

Fig. 3. Definition of the hybrid games from the proof of Theorem 1. The sets $\mathcal{S}_I$ are as in Equations (3) and (4). The hybrid $H_I^{\mathcal{P}}$ is defined like the full security game of a q-query adversary $\mathsf{A_f}$ against the constrained PRF GGM^G, but where we "guess", for any value in $p \in \mathcal{P}$, at which point in the experiment the node at depth p on the path from the root $K_\emptyset$ to the challenge K_{x^*} is computed (concretely, the guess is that it's the c_{p-1}th time we compute the children of an $p-1$ level node, we define the p level node $K_{x^*[1\ldots\ell]}$ on the path). At a subset of these points, namely $\mathcal{S}_I$, we embed random values. The final output is 0 unless all guesses were correct, in which case we forward $\mathsf{A_f}$'s output.

Thus, if $\mathsf{A_f}$ breaks the full security of GGM^G with advantage ϵ then

$$\left| \Pr[H^\emptyset_{\langle 0\rangle} = 1] - \Pr[H^\emptyset_{\langle 10_n\rangle} = 1] \right| \geq \epsilon \ . \tag{6}$$

In the proof of Proposition 3 we were able to "connect" the real and random experiments H_0 and H_{2N-1} via intermediate hybrids $H_1, \ldots, H_{2N-2}$, such that from a distinguisher for any two consecutive hybrids we can build a distinguisher for G with the same advantage.

We did this by using random values (instead of applying G) in some steps along the path from the root $K_\emptyset$ to the challenge K_{x^*}. Here we cannot use the same approach to connect $H^\emptyset_{\langle 0\rangle}$ and $H^\emptyset_{\langle 10_n\rangle}$, as these games consider full (and not selective) security, where we learn x^* only at the very end, and thus "the path to x^*" is not even defined until the adversary makes the challenge query.

We could reduce the problem from the full to the selective setting by guessing x^* at the beginning like in the proof of Lemma 1, but this would lose a factor 2^N, which is what we want to avoid.

Instead of guessing the entire x^*, we will guess something easier. During the experiment $H_{\langle 0\rangle}$ we have to compute at most q children $K_{z\|0} \| K_{z\|1} = \mathsf{G}(K_z)$ of nodes at level $N/2-1$, i.e., $z \in \{0,1\}^{N/2-1}$. One of these K_z satisfies $z = x^*[1 \ldots N/2 - 1]$, that is, it lies on the path from the root $K_\emptyset$ to the challenge K_{x^*} (potentially this happens only at the very last query $x_q = x^*$). We randomly guess $q_{N/2} \xleftarrow{*} [q]$ for which invocation of G this will be the case *for the first time*. Note that we have to wait until $\mathsf{A_f}$ makes its last query $x_q = x^*$ before we know whether our guess was correct. If the guess was wrong, we output 0; otherwise we output $\mathsf{A_f}$'s output. We will denote the position of the node down to which our guessed query should equal the path to x^* as superscript of the hybrid H. The experiment just described corresponds thus to hybrid $H^{\{N/2\}}_{\langle 0\rangle}$, as defined in Figure 3.

The games $H^{\{N/2\}}_{\langle 0\rangle}$ and $H^{\{N/2\}}_{\langle 10_n\rangle}$ behave *exactly* like $H^{\emptyset}_{\langle 0\rangle}$ and $H^{\emptyset}_{\langle 10_n\rangle}$, except for the final output, which in the former two hybrids is set to 0 with probability $1-1/q$, and left unchanged otherwise (namely, in case our random guess $q_{N/2} \xleftarrow{*} [q]$ turns out to be correct, which we know after learning x^*). This implies

$$\Pr[H^{\{N/2\}}_{\langle 0\rangle}=1] = \Pr[H^{\emptyset}_{\langle 0\rangle}=1]\cdot \tfrac{1}{q} \quad \text{and} \quad \Pr[H^{\{N/2\}}_{\langle 10_n\rangle}=1] = \Pr[H^{\emptyset}_{\langle 10_n\rangle}=1]\cdot \tfrac{1}{q}\ ,$$

and with (6)

$$\big|\Pr[H^{\{N/2\}}_{\langle 0\rangle}=1] - \Pr[H^{\{N/2\}}_{\langle 10_n\rangle}=1]\big| \geq \epsilon/q\ . \tag{7}$$

What did we gain? We paid a factor q in the advantage for aborting when our guess $q_{N/2}$ was wrong. What we gained is that when we guess correctly we know $x^*[1\ldots N/2]$, i.e., the node halfway in between the root and the challenge.

We use this fact to define two new hybrids $H^{\{N/2\}}_{\langle 10_{n-1}\rangle}, H^{\{N/2\}}_{\langle 20_{n-1}\rangle}$ which are defined like $H^{\{N/2\}}_{\langle 0\rangle}, H^{\{N/2\}}_{\langle 10_n\rangle}$, respectively, but where the children of $K_{x^*[1\ldots N/2-1]}$ are uniformly random instead of being computed by applying G to $K_{x^*[1\ldots N/2-1]}$.

Figure 2 (ignoring the boxes for now) illustrates the path from $K_{\emptyset}$ to K_{x^*} in the hybrids $H^{\{4\}}_{\langle 0\rangle}, H^{\{4\}}_{\langle 100\rangle}$, $H^{\{4\}}_{\langle 200\rangle}, H^{\{4\}}_{\langle 1000\rangle}$ assuming the guessing was correct (a node with label i corresponds to $K_{x^*[1\ldots i]}$, blue nodes are sampled at random, and green ones by applying G to the parent).

By (7) we can distinguish the first from the last hybrid with advantage ϵ/q, and thus there are two consecutive hybrids in the sequence $H^{\{N/2\}}_{\langle 0\rangle}, H^{\{N/2\}}_{\langle 10_{n-1}\rangle}, H^{\{N/2\}}_{\langle 20_{n-1}\rangle}, H^{\{N/2\}}_{\langle 10_n\rangle}$ that we can distinguish with advantage at least $\epsilon/(3q)$. For concreteness, let us fix parameters $N=8=2^3=2^n$ as in Figure 2 and assume that this is the case for the last two hybrids in the sequence, i.e.,

$$|\Pr[H^{\{4\}}_{\langle 200\rangle}=1] - \Pr[H^{\{4\}}_{\langle 1000\rangle}=1]| \geq \epsilon/(3q)\ . \tag{8}$$

The central observation here is that the above guessing step (losing a factor q) followed by a hybrid argument (losing a factor 3) transformed a distinguishing advantage ϵ for two hybrids $H^{\emptyset}_{\langle 0\rangle}, H^{\emptyset}_{\langle 1000\rangle}$ which have random values embedded along the path from $K_{\emptyset}$ to K_{x^*} on positions defined by N-neighboring sets $\mathcal{S}_{\langle 0\rangle}, \mathcal{S}_{\langle 1000\rangle}$, into a distinguishing advantage of $\epsilon/(3q)$ for two hybrids that correspond to $N/2$-neighboring sets, e.g. $\mathcal{S}_{\langle 200\rangle}$ and $\mathcal{S}_{\langle 1000\rangle}$.

We can now iterate this approach, losing in each iteration a factor $3q$ in distinguishing advantage, but obtaining hybrids that correspond to sets of half the neighboring distance. After $n=\log N$ iterations we end up with hybrids that correspond to 1-neighboring sets, and can be distinguished with advantage $\epsilon/(3q)^n$. (We will make this formal in Lemma 3 below.) From any distinguisher for hybrids corresponding to two 1-neighboring sets we can construct a distinguisher for G with the same advantage, as formally stated in Lemma 2 below. Let's continue illustrating the approach using the hybrids illustrated in Figure 2.

Recall that we assumed that we can distinguish $H^{\{4\}}_{\langle 200\rangle}$ and $H^{\{4\}}_{\langle 1000\rangle}$ as stated in eq. (8). We now embed hybrids corresponding to the sets $\mathcal{S}_{\langle 210\rangle}, \mathcal{S}_{\langle 220\rangle}$ in between, illustrated in the outer box in Figure 2 (ignore the inner box for now). Since $\mathcal{S}_{\langle 200\rangle}\,\Delta\,\mathcal{S}_{\langle 1000\rangle} = \{4\}$, by eq. (4) for $\ell=1$, we construct $\mathcal{S}_{\langle 200\rangle+\langle 10\rangle} = \mathcal{S}_{\langle 200\rangle}\cup\{4-\frac{8}{2^2}=2\}$ and $\mathcal{S}_{\langle 1000\rangle-\langle 10\rangle} = \mathcal{S}_{\langle 1000\rangle}\cup\{2\}$. We add this new element $\{2\}$ to the "guessing set" $\{4\}$, at the price of losing a factor q in distinguishing advantage compared to eq. (8):

$$\big|\Pr[H^{\{2,4\}}_{\langle 200\rangle}=1] - \Pr[H^{\{2,4\}}_{\langle 1000\rangle}=1]\big| \geq \epsilon/(3q^2)\ . \tag{9}$$

We can now consider the sequence of hybrids $H^{\{2,4\}}_{\langle 200\rangle}, H^{\{2,4\}}_{\langle 210\rangle}, H^{\{2,4\}}_{\langle 220\rangle}, H^{\{2,4\}}_{\langle 1000\rangle}$. There must be two consecutive hybrids that can be distinguished with advantage $\epsilon/(3^2q^2)$. Let's assume this is the case for the middle two.

$$\big|\Pr[H^{\{2,4\}}_{\langle 210\rangle}=1]-\Pr[H^{\{2,4\}}_{\langle 220\rangle}=1]\big| \geq \epsilon/(3^2q^2)\ . \tag{10}$$

Now $\mathcal{S}_{\langle 210\rangle}\,\Delta\,\mathcal{S}_{\langle 220\rangle}=\{4\}$, and $4-8/2^3=3$, so we add $\{3\}$ to the guessing set losing another factor q:

$$\big|\Pr[H^{\{2,3,4\}}_{\langle 210\rangle}=1]-\Pr[H^{\{2,3,4\}}_{\langle 220\rangle}=1]\big| \geq \epsilon/(3^2q^3)\ , \tag{11}$$

and can now consider the games $H^{\{2,3,4\}}_{\langle 210\rangle}, H^{\{2,3,4\}}_{\langle 211\rangle}, H^{\{2,3,4\}}_{\langle 212\rangle}, H^{\{2,3,4\}}_{\langle 220\rangle}$ as shown inside the two boxes in Figure 2. Two consecutive hybrids in this sequence must be distinguishable with advantage at least 1/3 of the advantage we had for the first and last hybrid in this sequence; let's assume this is the case for the last two, then:

$$\big|\Pr[H^{\{2,3,4\}}_{\langle 212\rangle}=1]-\Pr[H^{\{2,3,4\}}_{\langle 220\rangle}=1]\big| \geq \epsilon/(3^3q^3)\ . \tag{12}$$

We have thus shown the existence of two games $H^{\mathcal{P}}_I$ and $H^{\mathcal{P}}_{I+\langle 1\rangle}$ (what $\mathcal{P}$ and I are exactly is not important for the rest of the argument) that can be distinguished with advantage $\epsilon/(3q)^n$. Any two consecutive (i.e., 1-neighboring) hybrids have the following properties (cf. eq. (5)). They only differ in one node on the path to x^* and its parent node is random. Moreover, by construction, the position of the differing node is in the guessing set $\mathcal{P}$, meaning we know its position in the tree. Together, this means we can use a distinguisher between $H^{\mathcal{P}}_I$ and $H^{\mathcal{P}}_{I+\langle 1\rangle}$ to break G as follows. Given a challenge for G we embed it as the value of the differing node and, depending on whether it was real or random, simulate one hybrid or the other. We formalize this in the following lemma, which is proven in Appendix B.4.

Lemma 2. *For any $I\in\{\langle 0\rangle,\ldots,\langle 2_n\rangle\}, \mathcal{P}\subset\{1,\ldots,N-1\}$ where $\mathcal{S}_I\cup\mathcal{S}_{I+\langle 1\rangle}\subseteq\mathcal{P}\cup\{0,N\}$ (so the games $H^{\mathcal{P}}_{I+\langle 1\rangle}, H^{\mathcal{P}}_I$ are defined) the following holds. If*

$$\big|\Pr[H^{\mathcal{P}}_I=1]-\Pr[H^{\mathcal{P}}_{I+\langle 1\rangle}=1]\big| = \delta$$

then G is not a (δ,s)-secure PRG for $s=|\mathsf{A_f}|-O(q\cdot N\cdot|\mathsf{G}|)$.

Lemma 3. *For $\ell\in\{0,\ldots,n-1\}$, any consecutive level-ℓ sets $\mathcal{S}_I,\mathcal{S}_{I'}$ (i.e., I,I' are of the form $\langle ?0_{n-\ell}\rangle$ and $I'=I+\langle 10_{n-\ell}\rangle$) and any $\mathcal{P}$ for which the hybrids $H^{\mathcal{P}}_I, H^{\mathcal{P}}_{I'}$ are defined (that is, $\mathcal{S}_I\cup\mathcal{S}_{I'}\subseteq\mathcal{P}\cup\{0,N\}$), the following holds. If*

$$\big|\Pr[H^{\mathcal{P}}_I=1]-\Pr[H^{\mathcal{P}}_{I'}=1]\big| = \delta \tag{13}$$

then for some consecutive level-$(\ell+1)$ sets $J\in\{I, I+\langle 10_{n-(\ell+1)}\rangle, I+\langle 20_{n-(\ell+1)}\rangle\}$ and $J'=J+\langle 10_{n-(\ell+1)}\rangle$ and some $\mathcal{P}'$:

$$\big|\Pr[H^{\mathcal{P}'}_J=1]-\Pr[H^{\mathcal{P}'}_{J'}=1]\big| = \delta/(3q)\ .$$

The proof of Lemma 3 is in Appendix B.5. The theorem now follows from Lemmata 2 and 3 as follows. Assume a q-query adversary $\mathsf{A_f}$ breaks the full security of $\mathsf{GGM}^{\mathsf{G}}$ for domain $\mathcal{X}=\{0,1\}^{2^n}$ with advantage ϵ, which, as explained in the paragraph before eq. (6), means that we can distinguish the two level-0 hybrids $H^{\emptyset}_{\langle 0\rangle}$ and $H^{\emptyset}_{\langle 10_n\rangle}$ with advantage ϵ. Applying Lemma 3 n times, we get that there exist consecutive level-n hybrids $H^{\mathcal{P}}_I, H^{\mathcal{P}}_{I+\langle 1\rangle}$ that can be distinguished with advantage $\epsilon/(3q)^n$, which by Lemma 2 implies that we can break the security of G with the same advantage $\epsilon/(3q)^n$. This concludes the proof of Theorem 1.

To reduce the loss to $2q\log q\cdot(3q)^{n-\log\log q}$ as stated below Theorem 1, we use the same proof as above, but stop after $n-\log\log q$ (instead of n) iterations. At this point, we have lost a factor $(3q)^{n-\log\log q}$, and have constructed games that are ($\log q$)-neighboring. We can now use a proof along the lines of the proof of Proposition 3, and guess the entire remaining path of length $\log q$ at once. This step loses a factor $2q \log q$ (a factor $q = 2^{\log q}$ to guess the path, and another $2 \log q$ as we have a number of hybrids which is twice the length of the path).

5 Impossibility Result for Prefix-Fixing Boneh-Waters PRF

In this section we show that we cannot hope to prove full security using known techniques without an exponential loss for another constrained PRF, namely the one due to Boneh and Waters [BW13].

5.1 The Boneh-Waters Constrained PRF

Leveled κ-linear maps. The Boneh-Waters constrained PRF [BW13] is based on leveled multilinear maps [GGH13], of which they use the following abstraction.

We assume a *group generator* $\mathcal{G}$ that takes as input a security parameter 1^λ and the number of levels $\kappa \in \mathbb{N}$ and outputs a sequence of groups $(\mathbb{G}_1, \ldots, \mathbb{G}_\kappa)$, each of prime order $p > 2^\lambda$, generated by g_i, respectively, such that there exists a set of bilinear maps $\{e_{i,j} \colon \mathbb{G}_i \times \mathbb{G}_j \to \mathbb{G}_{i+j} \mid i, j \geq 1;\ i + j \leq \kappa\}$ with

$$\forall\, a, b \in \mathbb{Z}_p : \ e_{i,j}(g_i^a, g_j^b) = (g_{i+j})^{\ ab} \ .$$

(For simplicity we will omit the indices of e.) Security of the PRF is based on the following assumption.

The *κ-multilinear decisional Diffie-Hellman assumption* states that given the output of $\mathcal{G}(1^\lambda, \kappa)$ and $(g_1, g_1^{c_1}, \ldots, g_1^{c_{\kappa+1}})$ for random $(c_1, \ldots, c_{\kappa+1}) \xleftarrow{*} \mathbb{Z}_p^{\kappa+1}$, no polynomial-time adversary can distinguish $(g_\kappa)^{\prod_{j \in [\kappa+1]} c_j}$ from a random element in $\mathbb{G}_\kappa$ with better than negligible advantage in λ.

The Boneh-Waters bit-fixing PRF. Based on a multilinear-group generator $\mathcal{G}$, Boneh and Waters [BW13] define a PRF with domain $\mathcal{X} = \{0,1\}^N$ and range $\mathcal{Y} = \mathbb{G}_\kappa$, where $\kappa = N + 1$. The sets $S \subseteq \mathcal{X}$ for which constrained keys can be derived are subsets of $\mathcal{X}$ where certain bits are fixed; a set S is described by a vector $v \in \{0, 1, ?\}^N$ (where '?' acts as a wildcard) as $S_v := \{x \in \{0,1\}^N \mid \forall i \in [N] : v_i = ? \vee x_i = v_i\}$.

The PRF is set up for domain $\mathcal{X} = \{0,1\}^N$ by running $\mathcal{G}(1^\lambda, N+1)$ to generate a sequence of groups $(\mathbb{G}_1, \ldots, \mathbb{G}_{N+1})$. We let g denote the generator of $\mathbb{G}_1$. Secret keys are random elements from $\mathcal{K} := \mathbb{Z}_p^{2N+1}$:

$$k = (\alpha, d_{1,0}, d_{1,1}, \ldots, d_{N,0}, d_{N,1}) \ . \tag{14}$$

and the PRF is defined as

$$\mathsf{F} \colon \mathcal{K} \times \mathcal{X} \to \mathcal{Y} \ , \qquad (k, x) \mapsto (g_{N+1})^{\alpha \prod_{i \in [N]} d_{i,x_i}} \ .$$

$\mathsf{F.constrain}(k, v)$: On input a key k as in (14) and $v \in \{0, 1, ?\}^N$ describing the constrained set, output the key $k_v := \big(v, K, \{D_{i,b}\}_{i \in [N] \setminus V,\, b \in \{0,1\}}\big)$, where $V := \{i \in [N] \mid v_i \neq ?\}$ is the set of fixed indices,

$$K := (g_{|V|+1})^{\alpha \prod_{i \in V} d_{i,v_i}} \quad \text{ and } \quad D_{i,b} := g^{d_{i,b}} \ , \quad \text{ for } i \in [N] \setminus V,\ b \in \{0,1\} \ .$$

$\mathsf{F.eval}(k_v, x)$: On input $k_v = (v, K, \{D_{i,b}\}_{i \in [N] \setminus V,\, b \in \{0,1\}})$ and $x \in \mathcal{X}$:
- if for some $i \in V$: $x_i \neq v_i$, return $\perp$ (as x is not in S_v);
- if $|V| = N$, output K (as $S_v = \{v\}$ and $K = \mathsf{F}(k, v)$);
- else, compute $T := (g_{N-|V|})^{\prod_{i \in [N] \setminus V} d_{i,x_i}}$ via repeated application of the bilinear maps to the elements $D_{i,x_i} = g^{d_{i,x_i}}$ for $i \in [N] \setminus V$ and output $e(T, K) = (g_{N+1})^{\alpha \prod_{i \in [N]} d_{i,x_i}} = \mathsf{F}(k, x)$.

In [BW13] it is shown how to use an adversary breaking the constrained PRF for N-bit inputs with advantage $\epsilon(\lambda)$ to break the $(N+1)$-multilinear decisional Diffie-Hellman assumption with advantage $\frac{1}{2^N} \cdot \epsilon(\lambda)$. (The exponential factor comes from security leveraging.) In the next section we show that this is optimal in the sense that every simple reduction (Definition 9) from a decisional problem must lose a factor that is exponential in the input length N.

We actually prove a stronger statement. First, this security loss is necessary even when the CPRF is only used as a *prefix-fixing* PRF, that is, constrained keys are only issued for sets $S_{(z,?\ldots?)}$ with $z \in \{0,1\}^{\leq N}$. Second, the loss is necessary even for proofs of *unpredictability* of the CPRF, a weaker notion where the adversary must *compute* $\mathsf{F}(k, x^*)$ instead of distinguishing it from random.

Definition 7 (Unpredictability). *For a constrained PRF* $(\mathsf{F}, \mathsf{F.constrain}, \mathsf{F.eval})$ *consider the following experiment.*

- *The challenger chooses* $k \stackrel{*}{\leftarrow} \mathcal{K}$*;*
- A *can query* $\mathsf{F.constrain}$ *for sets* S_i*;*
- A *wins if it outputs* $(x, \mathsf{F}(k,x))$ *with* $x \in \mathcal{X}$ *and* $x \notin S_i$ *for all queried* S_i*.*

The CPRF is (ϵ, t, q)-**unpredictable** *if no* A *running in time at most* t *making at most* q *queries wins the above game with probability greater than* ϵ*.*

Since unpredictability follows from pseudorandomness without any security loss (we assume that the domain $\mathcal{X}$ is of superpolynomial size), our impossibility result holds a forteriori for pseudorandomness. In particular, this precludes security proofs for the Boneh-Waters CPRF using the technique from Section 4.

5.2 Adaptive Security of the Boneh-Waters CPRF

Hierarchical identity-based encryption (HIBE) [HL02] is a generalization of identity-based encryption where the identities are arranged in a hierarchy and from a key for an identity *id* one can derive keys for any identities that are below *id*. In the security game for HIBE the adversary receives the system parameters and can query keys for any identity. He then outputs (id, m_0, m_1) and, provided that *id* is not below any identity for which he queried a key, receives the encryption for *id* of one of the two messages, and wins if he guesses which one it was.

Lewko and Waters [LW14], following earlier work [Cor02,HJK12], show that it is hard to prove full security of HIBE schemes if one can check whether secret keys and ciphertexts are correctly formed w.r.t. the public parameters. In particular, they show that a simple black-box reduction (that is, one that runs the attacker once without rewinding; see below) from a decisional assumption must lose a factor that is exponential in the depth of the hierarchy. We adapt their proof technique and show that a proof of full security of the Boneh-Waters PRF with constrained keys for prefix-fixing must lose a factor that is exponential in the length of the PRF inputs.

The proof idea in [LW14] is the following. Assume that there exists a reduction which breaks a challenge with some probability δ after interacting with an adversary that breaks the security of the HIBE with some probability ϵ. We define a concrete adversary A, which, after receiving the public parameters, guesses a random identity *id* at the lowest level of the hierarchy and then queries keys for all identities except *id*, checking whether they are consistent with the parameters. Finally it outputs a challenge query for *id*.

Given a challenge, we run the reduction and simulate this adversary until we have keys for all identities except *id*. We then rewind the reduction to the point right after it sent the parameters to A and simulate A again (choosing a fresh random identity id'; thus $id' \neq id$ with high probability). A now asks for a challenge for id' and can break it by using the key for id' it received in the first run. It is crucial that keys can be verified w.r.t. the parameters, as this guarantees that the reduction cannot detect that a key from the first run was used to win in the second run (the parameters being the same in both runs).

The reduction can thus be used to break the challenge without any adversary, as we can simulate the adversary ourselves. (The actual proof, as well as that of Theorem 2 is more complex, as we need to rewind more than once.) We formally define decisional problems and simple reductions, following [LW14].

Definition 8. *A non-interactive* **decisional problem** $\Pi = (C, \mathcal{D})$ *is described by a set of challenges* C *and a distribution* $\mathcal{D}$ *on* C*. Each* $c \in C$ *is associated with a bit* $b(c)$*, the solution for challenge* c*. An algorithm* A (ϵ, t)*-solves* Π *if* A *runs in time at most* t *and*

$$\Pr_{c \xleftarrow{\mathcal{D}} C}\left[b(c) \leftarrow \mathsf{A}(c)\right] \geq \tfrac{1}{2} + \epsilon \ .$$

Definition 9. *An algorithm* $\mathcal{R}$ *is a* **simple** $(t, \epsilon, q, \delta, t')$**-reduction** *from a decisional problem* Π *to breaking unpredictability of a CPRF if, when given black-box access to any adversary* A *that* (t, ϵ, q)*-breaks unpredictability then* $\mathcal{R}$ (δ, t')*-solves* Π *after simulating the unpredictability game once for* A*.*

We show that every simple reduction from a decisional problem to unpredictability of the Boneh-Waters CPRF must lose at least a factor exponential in N. Instead of checking validity of keys computed by the reduction w.r.t. the public parameters, as in [LW14], we show that after two concrete key queries, the secret key k used by the reduction is basically fixed; the two received constrained keys are thus a "fingerprint" of the secret key. Moreover, we show that, by using the multilinear map, correctness of any key can be verified w.r.t. to this fingerprint; which gives us the required checkability property. We define an adversary A that we can simulate by rewinding the reduction: After making the fingerprint queries, A chooses a random value $x^* \in \mathcal{X}$ and queries keys which allow it to evaluate all other domain points, checking every key is consistent with the fingerprint. (Note that keys for $(1 - x_1^*, ?, \ldots)$, $(x_1^*, 1 - x_2^*, ?, \ldots)$, $\ldots$,$(x_1^*, \ldots, x_{N-1}^*, 1 - x_N^*)$ allow evaluation of the PRF on $\mathcal{X} \backslash \{x^*\}$.)

By rewinding the reduction to the point after receiving the fingerprint and choosing a different $x' \neq x$, we can break security by using one of the keys obtained in the first run to evaluate the function at x'. The proof of the following can be found in Appendix C.

Theorem 2. *Let* $\Pi(\lambda)$ *be a decisional problem such that no algorithm running in time* $t = poly(\lambda)$ *has an advantage non-negligible in* λ*. Let* $\mathcal{R}$ *be a simple* $(t, \epsilon, q, \delta, t')$ *reduction from* Π *to unpredictability of the Boneh-Waters prefix-constrained PRF with domain* $\{0,1\}^N$*, with both* $t, t' = poly(\lambda)$*, and* $q \geq N - 1$*. Then* δ *vanishes exponentially as a function of* N *(up to terms that are negligible in* λ*).*

The reason why the same argument does not apply to the GGM construction is that its constrained keys are not "checkable". This is why in the intermediate hybrids we can embed random nodes on the path to x^*, which lead to constrained keys that are not correctly computed.

References

[AFL12] Michel Abdalla, Dario Fiore, and Vadim Lyubashevsky. From selective to full security: Semi-generic transformations in the standard model. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 316–333. Springer, 2012.

[BB04] Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, 2004.

[BCPR13] Nir Bitansky, Ran Canetti, Omer Paneth, and Alon Rosen. Indistinguishability obfuscation vs. auxiliary-input extractable functions: One must fall. Cryptology ePrint Archive, Report 2013/641, 2013. `http://eprint.iacr.org/`.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, 2001.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 501–519. Springer, 2014.

[BV98] Dan Boneh and Ramarathnam Venkatesan. Breaking RSA may not be equivalent to factoring. In Kaisa Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 59–71. Springer, 1998.

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 280–300. Springer, 2013.

[Cor02] Jean-Sébastien Coron. Optimal security proofs for PSS and other signature schemes. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 272–287. Springer, 2002.

[FS10] Marc Fischlin and Dominique Schröder. On the impossibility of three-move blind signature schemes. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 197–215. Springer, 2010.

[GGH13] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 1–17. Springer, 2013.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[Gol01] Oded Goldreich. *Foundations of Cryptography: Basic Tools*, volume 1. Cambridge University Press, Cambridge, UK, 2001.

[HJK12] Dennis Hofheinz, Tibor Jager, and Edward Knapp. Waters signatures with optimal security reduction. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 66–83. Springer, 2012.

[HL02] Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 466–481. Springer, 2002.

[Hof14] Dennis Hofheinz. Fully secure constrained pseudorandom functions using random oracles. IACR Cryptology ePrint Archive, Report 2014/372, 2014.

[HSW14] Susan Hohenberger, Amit Sahai, and Brent Waters. Replacing a random oracle: Full domain hash from indistinguishability obfuscation. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 201–220. Springer, 2014.

[HW09] Susan Hohenberger and Brent Waters. Short and stateless signatures from the RSA assumption. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 654–670. Springer, 2009.

[KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 13*, pages 669–684. ACM Press, 2013.

[LW14] Allison B. Lewko and Brent Waters. Why proving HIBE systems secure is difficult. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 58–76. Springer, 2014.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *46th ACM STOC*, pages 475–484. ACM Press, 2014.

A Hybrid Proofs

In this section we show a simple application of the hybrid technique to prove security (i.e., pseudorandomness of the output) of the "stream-cipher" we get when iterating a pseudorandom generator. The

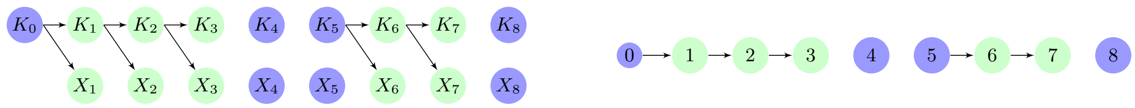

Fig. 4. The left picture shows the evaluation of $\mathsf{SC}^{\mathsf{G}}_{\{0,4,5,8\}}(8)$. The output is $X_1, \ldots, X_8$. The arrows indicate the evaluation of G, e.g., $(K_1, X_1) \leftarrow \mathsf{G}(K_0)$. The blue values are sampled uniformly at random. The right picture illustrates the corresponding compact representation we will use.

simple proofs in this section already exemplify some of the techniques that we will use in the proof of Proposition 3 and Theorem 1.

Given a function $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$, we define the function $\mathsf{SC}^{\mathsf{G}}\colon \mathbb{N} \times \{0,1\}^\lambda \to \{0,1\}^*$ as

$$\mathsf{SC}^{\mathsf{G}}(N, K_0) = (X_1, \ldots, X_N) \ , \quad \text{where for } i \geq 1 \ : \ (K_i, X_i) \leftarrow \mathsf{G}(K_{i-1}) \ . \tag{15}$$

For $\mathcal{S} \subset \mathbb{N}^0 = \{0, 1, \ldots\}$, we denote by $\mathsf{SC}^{\mathsf{G}}_{\mathcal{S}}(N)$ the random variable that has the output distribution of $\mathsf{SC}^{\mathsf{G}}(N, K_0)$ instantiated with a random key K_0, but where for every $i \in \mathcal{S}$, the output in the i-th round is replaced with a uniformly random value:

$$\begin{aligned} &\text{For } N \in \mathbb{N}, \mathcal{S} \subset \mathbb{N}^0: \quad \mathsf{SC}^{\mathsf{G}}_{\mathcal{S}}(N) \to (X_1, \ldots, X_N) \ . \\ &\text{where } K_0 \xleftarrow{*} \{0,1\}^\lambda \text{ and for } i \geq 1: \ \begin{cases} (K_i, X_i) \leftarrow \mathsf{G}(K_{i-1}) & \text{if } i \notin \mathcal{S} \\ (K_i, X_i) \xleftarrow{*} \{0,1\}^{2\lambda} & \text{otherwise} \end{cases} \end{aligned} \tag{16}$$

It will be convenient to require that 0 is always contained in $\mathcal{S}$ (which makes sense as K_0 is always random). In Figure 4 we illustrate the evaluation of $\mathsf{SC}^{\mathsf{G}}_{\{0,4,5,8\}}(8)$. Note that

$$\mathsf{SC}^{\mathsf{G}}_{\{0\}}(N) \sim \mathsf{SC}^{\mathsf{G}}(N, U_\lambda) \quad \text{and} \quad \mathsf{SC}^{\mathsf{G}}_{\{0,\ldots,N\}}(N) \sim U_{N\lambda} \ .$$

Definition 10. $\mathsf{SC}^{\mathsf{G}}\colon \mathbb{N} \times \{0,1\}^\lambda \to \{0,1\}^*$ *is* (N, ϵ', s')-**pseudorandom** *if no circuit of size s' can distinguish with advantage greater than ϵ' the first N blocks of output of SC^{G} from random when instantiated with a random key, i.e.,*

$$\mathsf{SC}^{\mathsf{G}}(N, U_\lambda) \sim_{(\epsilon', s')} U_{N\lambda} \ .$$

We say that SC^{G} is (N, ϵ', s') **next-block pseudorandom** *if, for any $N' \leq N$, no circuit of size s' can distinguish the N'-th output block from random given the first $N' - 1$ blocks, i.e.,*

$$\mathsf{SC}^{\mathsf{G}}(N', U_\lambda) \sim_{(\epsilon', s')} \mathsf{SC}^{\mathsf{G}}_{\{0,N'\}}(N') \sim \mathsf{SC}^{\mathsf{G}}(N'-1, U_\lambda) \| U_\lambda \ .$$

To prove (next-block) pseudorandomness of SC^{G}, we will use a hybrid argument. We call two hybrids $\mathsf{SC}^{\mathsf{G}}_{\mathcal{S}}(N)$ and $\mathsf{SC}^{\mathsf{G}}_{\mathcal{S}'}(N)$ *neighboring* if $\mathcal{S}$ and $\mathcal{S}'$ are 1-neighboring, as in Definition 6. The following lemma shows that two neighboring hybrids are indistinguishable if G is pseudorandom.

Lemma 4. *For any $N \in \mathbb{N}^+$ and two neighboring sets $\mathcal{S} \subset \mathcal{S}' \subseteq [N]$*

$$\mathsf{G}(U_\lambda) \sim_{(\epsilon, s)} U_{2\lambda} \quad \Rightarrow \quad \mathsf{SC}^{\mathsf{G}}_{\mathcal{S}}(N) \sim_{(\epsilon, s')} \mathsf{SC}^{\mathsf{G}}_{\mathcal{S}'}(N) \ ,$$

where $s' \approx s - N|\mathsf{G}|$.

Proof. We assume w.l.o.g. that $|\mathcal{S}'| > |\mathcal{S}|$. Given a PRG challenge $C \in \{0,1\}^{2\lambda}$, we can sample a variable X s.t.

$$X \sim \mathsf{SC}^{\mathsf{G}}_{\mathcal{S}}(N) \quad \text{if} \quad C \sim \mathsf{G}(U_\lambda) \quad \text{and} \quad X \sim \mathsf{SC}^{\mathsf{G}}_{\mathcal{S}'}(N) \quad \text{if} \quad C \sim U_{2\lambda}$$

as follows. Let $d \in \mathcal{S}'$ be the (unique) element not in $\mathcal{S}$. Now sample $\mathsf{SC}^{\mathsf{G}}_{\mathcal{S}}(N)$ as in (16), except that in the d-th step we use $(K_d, X_d) := C$. (Note that (K_{d-1}, X_{d-1}) is random as per 2. in Definition 6.) Thus, from any distinguisher for $\mathsf{SC}^{\mathsf{G}}_{\mathcal{S}}(N)$ and $\mathsf{SC}^{\mathsf{G}}_{\mathcal{S}'}(N)$, we get a distinguisher for the PRG G with the same advantage. □

We will also use the triangle inequality for indistinguishability.

Proposition 4. *Consider any random variables $H_0, H_1, \ldots, H_N$ then*

$$H_0 \not\sim_{(\epsilon,s)} H_N \;\Rightarrow\; \exists\, i \in [N]:\; H_{i-1} \not\sim_{(\frac{\epsilon}{N},s)} H_i \;.$$

Together with Lemma 4, this yields that the construction of the stream-cipher SC^{G} in eq. (15) is secure.

Proposition 5. *If $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ is an (ϵ, s)-secure PRG, then SC^{G} is (N, ϵ', s') pseudorandom with*

$$\epsilon' = \epsilon \cdot N \quad \text{and} \quad s' \approx s - N\,|\mathsf{G}| \;.$$

Proof. Consider the hybrids $H_0, \ldots, H_N$ where $H_i = \mathsf{SC}^{\mathsf{G}}_{[i]_0}(N)$ (as illustrated in Figure 5 for $N = 8$; the hybrids $H_9, \ldots, H_{15}$ in the figure are not needed in this proof). Assume for contradiction that $\mathsf{SC}^{\mathsf{G}}_{\{0\}}(N)$ is not $(\epsilon N, s')$ indistinguishable from $\mathsf{SC}^{\mathsf{G}}_{[N]_0}(N)$, i.e.,

$$H_0 \not\sim_{(\epsilon N, s')} H_N \;.$$

Then by Proposition 4, for some $i \in [N]$

$$H_{i-1} \not\sim_{(\epsilon, s')} H_i \;.$$

Since $[i-1]_0$ and $[i]_0$ are neighboring sets, applying Lemma 4, we get,

$$\mathsf{G}(U_\lambda) \not\sim_{(\epsilon, s' + N|\mathsf{G}|)} U_{2\lambda} \;,$$

contradicting (ϵ, s)-security of G. □

Proposition 6. *If $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ is an (ϵ, s)-secure PRG, then SC^{G} is (N, ϵ', s')* **next-block** *pseudorandom with*

$$\epsilon' = \epsilon \cdot \tfrac{1}{2N-1} \quad \text{and} \quad s' \approx s - (2N-1)|\mathsf{G}| \;.$$

We omit the proof as it is almost identical to the proof of Proposition 5, except that we use hybrids H_0 to H_{2N-1} (not just H_N) as illustrated in Figure 5.

Here each hybrid H_i corresponds to the variable $\mathsf{SC}^{\mathsf{G}}_{\mathcal{S}_i}(N)$, for $\mathcal{S}_0, \ldots, \mathcal{S}_{2N-1} \subseteq [N]$ where $\mathcal{S}_0 = \{0\}, \mathcal{S}_{2N-1} = \{0, N\}$ and for any i, the sets $\mathcal{S}_{i-1}$ and $\mathcal{S}_i$ are neighboring. Concretely,

$$\begin{aligned}
\mathcal{S}_0 &= \{0\} \\
\mathcal{S}_i &= \{0, 1, \ldots, i\} && \text{for } i \in \{1, \ldots, N\} \\
\mathcal{S}_i &= \{0, 1, \ldots, 2N - i - 1, N\} && \text{for } i \in \{N+1, \ldots, 2N-2\} \\
\mathcal{S}_{2N-1} &= \{0, N\}
\end{aligned} \tag{17}$$

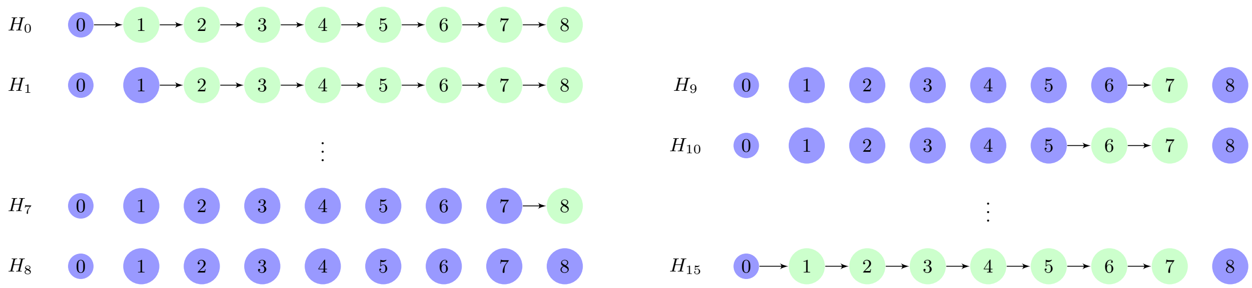

Fig. 5. The hybrids $H_0, \ldots, H_N$ (for $N = 8$) are as defined in the proof of Proposition 5. The proof of Proposition 6 additionally uses the hybrids $H_{N+1}, \ldots, H_{2N-1}$.

B Omitted Proofs

B.1 Proof of Lemma 1

From any adversary $\mathsf{A_f}$ against the full security of F we can construct an adversary $\mathsf{A_s}$ (of basically the same size) against the selective security of F losing a factor of $|\mathcal{X}|$ in the advantage, i.e.

$$\mathsf{Adv}_{\mathsf{F}}^{\mathsf{sel}}(\mathsf{A_s}, q) = \tfrac{1}{|\mathcal{X}|} \cdot \mathsf{Adv}_{\mathsf{F}}^{\mathsf{full}}(\mathsf{A_f}, q) \ , \tag{18}$$

as follows. $\mathsf{A_s}$ initially simply outputs a random $x' \overset{*}{\leftarrow} \mathcal{X}$ in the selective security game. Using its own oracle to answer $\mathsf{A_f}$'s queries, $\mathsf{A_s}$ then runs $\mathsf{A_f}^{\mathcal{O}(\cdot)}$, which outputs some x^*. If $x^* = x'$ then $\mathsf{A_s}$ uses $\mathsf{A_f}$ for the rest of the experiment, i.e., it forwards the challenge C_b to $\mathsf{A_f}$ and then returns the bit $\tilde{b}$ that $\mathsf{A_f}$ outputs. If $x^* \neq x'$ then $\mathsf{A_s}$ answers with $\tilde{b} = 0$. Thus, $\mathsf{A_s}$ outputs 0 with probability $1 - \frac{1}{|\mathcal{X}|}$, and whatever $\mathsf{A_f}$ outputs otherwise. Let ϵ (possibly negative) be such that

$$\Pr_{b \overset{*}{\leftarrow} \{0,1\}} \left[\mathbf{Exp}_{\mathsf{CPRF}}^{\mathsf{full}}(\mathsf{A_f}, \mathsf{F}, b, q) = b\right] = \tfrac{1}{2} + \epsilon \ . \tag{19}$$

Then

$$\begin{aligned}
&\Pr_{b \overset{*}{\leftarrow} \{0,1\}} \left[\mathbf{Exp}_{\mathsf{CPRF}}^{\mathsf{sel}}(\mathsf{A_s}, \mathsf{F}, b, q) = b\right] \\
&= \Pr_{b \overset{*}{\leftarrow} \{0,1\}} \left[\mathbf{Exp}_{\mathsf{CPRF}}^{\mathsf{sel}}(\mathsf{A_s}, \mathsf{F}, b, q) = b \;\middle|\; x^* = x'\right] \Pr[x^* = x'] + \\
&\qquad\qquad \Pr_{b \overset{*}{\leftarrow} \{0,1\}} \left[\mathbf{Exp}_{\mathsf{CPRF}}^{\mathsf{sel}}(\mathsf{A_s}, \mathsf{F}, b, q) = b \;\middle|\; x^* \neq x'\right] \Pr[x^* \neq x'] \\
&= \Pr_{b \overset{*}{\leftarrow} \{0,1\}}[\mathbf{Exp}_{\mathsf{CPRF}}^{\mathsf{full}}(\mathsf{A_f}, \mathsf{F}, b, q) = b] \cdot \tfrac{1}{|\mathcal{X}|} + \tfrac{1}{2} \cdot \left(1 - \tfrac{1}{|\mathcal{X}|}\right) \\
&= \left(\tfrac{1}{2} + \epsilon\right) \tfrac{1}{|\mathcal{X}|} + \tfrac{1}{2} \cdot \left(1 - \tfrac{1}{|\mathcal{X}|}\right) \;=\; \tfrac{1}{2} + \tfrac{\epsilon}{|\mathcal{X}|} \ .
\end{aligned}$$

By (1) this means $2|\epsilon|\frac{1}{|\mathcal{X}|} = \mathsf{Adv}_{\mathsf{F}}^{\mathsf{sel}}(\mathsf{A_s}, q)$; on the other hand (1) and (19) give $2|\epsilon| = \mathsf{Adv}_{\mathsf{F}}^{\mathsf{full}}(\mathsf{A_f}, q)$, which proves (18).

B.2 Proof of Proposition 2

We consider two hybrid games H_0 and H_m, which will correspond to the experiments

$$H_0 \sim \mathsf{A}^{\mathsf{F}(K,\cdot)} \quad \text{and} \quad H_m \sim \mathsf{A}^{f(\cdot)} \ , \quad \text{where} \quad K \overset{*}{\leftarrow} \mathcal{K},\ f \overset{*}{\leftarrow} \mathcal{F}[\{0,1\}^*, \{0,1\}^\lambda] \ .$$

Experiment H_i

$\forall x \in \{0,1\}^{\leq N}: \ K_x := \bot$
$K_\emptyset \xleftarrow{*} \{0,1\}^\lambda$
$x^* \leftarrow \mathsf{A_s}$
// Below, $\mathsf{A_s}$ can make max. q queries
// and last query must be x^*:
$\mathsf{A_s}^{\mathcal{O}(\cdot)}$
$\tilde{b} \leftarrow \mathsf{A_s}$
Return $\tilde{b}$

$\mathcal{O}(x = x[1\dots\ell])$
// Return K_x if it is already defined:
If $K_x \neq \bot$ then return K_x
// Get parent of K_x recursively
$K_{x[1\dots\ell-1]} := \mathcal{O}(x[1\dots\ell-1])$
// Compute K_x and its sibling using G,
// unless $x[1\dots\ell-1]$ is a prefix of x^* and $|x| \in \mathcal{S}_i$,
// in which case use random values:
If $x[1\dots\ell-1] = x^*[1\dots\ell-1]$ and $\ell \in \mathcal{S}_i$ then
$K_{x[1\dots\ell-1]\|0} \| K_{x[1\dots\ell-1]\|1} \xleftarrow{*} U_{2\lambda}$
else
$K_{x[1\dots\ell-1]\|0} \| K_{x[1\dots\ell-1]\|1} := \mathsf{G}(K_{x[1\dots\ell-1]})$
Return K_x

Fig. 6. Definition of the hybrid games $H_0, \dots, H_{2N-1}$ from the proof of Proposition 3, where $\mathcal{S}_i$ are as in eq. (17).

For ease of describing the subsequent hybrids, we describe H_0 as follows. In H_0 we begin by initially defining $K_x = \bot$ for all $x \in \{0,1\}^*$ and then sampling $K_\emptyset \xleftarrow{*} \mathcal{K}$. We then invoke A, who makes queries $x_1, x_2, \dots$ (of total length at most m), where we answer each query $x = x[1\dots\ell]$ with K_x, defined as follows: determine the largest ℓ' s.t. $K_{x[1\dots\ell']} \neq \bot$, and then for $j = \ell', \dots, \ell-1$ recursively define $K_{x[1\dots j\|0]} \| K_{x[1\dots j\|1]} := \mathsf{G}(K_{x[1\dots j]})$. The output of H_0 is whatever A finally outputs. We just emulated $\mathsf{F}(K,\cdot)$ for A, and thus $H_0 \sim \mathsf{A}^{\mathsf{F}(K,\cdot)}$.

Now for any $i \geq 0$, we define the experiment H_i to be the same as the experiment H_0, except that we replace the outputs of the first i invocations of G with uniformly random values. We have $H_m \sim \mathsf{A}^{f(\cdot)}$, as all the outputs A gets in H_m are uniformly random, exactly like the outputs of $f(\cdot)$.

It follows that if A can distinguish $\mathsf{F}(K,\cdot)$ from a random function (i.e., distinguish H_0 from H_m) with advantage ϵ then there are two hybrids H_i, H_{i+1} s.t.

$$\left|\Pr[1 \leftarrow H_i] - \Pr[1 \leftarrow H_{i+1}]\right| \geq \tfrac{\epsilon}{m} \ .$$

Using this, we can distinguish the output $\mathsf{G}(U_\lambda)$ from a random $U_{2\lambda}$ with the same advantage: given a challenge C, simulate the experiment H_i up to the $(i+1)$-th invocation of G, and replace its output with C. If $C = \mathsf{G}(U_\lambda)$, this emulates experiment H_i, and if $C = U_{2\lambda}$, this emulates experiment H_{i+1}.

B.3 Proof of Proposition 3

Full security as stated in Item *2.* of the proposition follows from selective security (Item *1.*) by complexity leveraging as explained in Lemma 1.

To prove selective security, we will use hybrid games $H_0, \dots, H_{2N-1}$ as defined in Figure 6. By inspection we see that the first and last hybrids defined in Figure 6 are exactly the real and random selective security game, i.e.,

$$H_0 \sim \mathbf{Exp}^{\mathsf{sel}}_{\mathsf{CPRF}}(\mathsf{A_s}, \mathsf{GGM}^{\mathsf{G}}, 0, q) \quad \text{and} \quad H_{2N-1} \sim \mathbf{Exp}^{\mathsf{sel}}_{\mathsf{CPRF}}(\mathsf{A_s}, \mathsf{GGM}^{\mathsf{G}}, 1, q) \ .$$

The other hybrids correspond to games where we sometimes use uniformly random values instead of the output of G on the path from the root $K_\emptyset$ to the challenge output K_{x^*}. More precisely, in H_i we use random values in the j-th step along the path computing the output for x^* for all $j \in \mathcal{S}_i$, with $\mathcal{S}_i$ as in eq. (17) and illustrated in Figure 5.

From any distinguisher for two neighboring hybrids H_i, H_{i+1} we get a distinguisher for G with the same advantage as follows. $\mathcal{S}_i$ and $\mathcal{S}_{i+1}$ differ by exactly one element d (and $d-1 \in \mathcal{S}_i, d-1 \in \mathcal{S}_{i+1}$). Given a PRG challenge C, simulate the experiment H_i up to the point where $\mathcal{O}(\cdot)$ (as in Figure 6) is queried for the first time on a query x where $x[1\dots d-1] = x^*[1\dots d-1]$. At this point, we embed the challenge $K_{x[1\dots d-1]\|0} \| K_{x[1\dots d-1]\|1} := C$. Depending on whether $C \overset{*}{\leftarrow} \mathsf{G}(U_\lambda)$ or $C \overset{*}{\leftarrow} U_{2\lambda}$, this will simulate either the experiment H_i or H_{i+1} (when $i < N$) and H_{i+1} or H_i (when $i \geq N$), respectively. Thus, we can break the security of G with the same advantage we have in distinguishing H_i from H_{i+1}. As there are $2N$ hybrids, we lose a factor $2N-1$; the reason we stated a loss of $2N$ in the proposition is explained in Remark 4.

B.4 Proof of Lemma 2

By eq. (5), $\mathcal{S}_I$ and $\mathcal{S}_{I+\langle 1\rangle}$ differ by exactly one element $d \in \{1,\dots,N-1\}$. Assume that $d \in \mathcal{S}_{I+\langle 1\rangle}$ (the case where $d \in \mathcal{S}_I$ is symmetric).

Given a PRG challenge $C \in \{0,1\}^{2\lambda}$, we simulate the game $H^{\mathcal{P}}_{I+\langle 1\rangle}$ (defined in Figure 3), but at the point where in $\mathcal{O}(\cdot)$ the if clause "if $\ell - 1 \in \mathcal{P}$ and $c_{\ell-1} = q_{\ell-1}$" evaluates to true for $\ell = d$, we set $K_{x[1\dots\ell-1]\|0} \| K_{x[1\dots\ell-1]\|1} := C$ (instead of assigning it a random value). If the challenge was sampled as $C \overset{*}{\leftarrow} U_{2\lambda}$ then we still simulated the game $H^{\mathcal{P}}_{I+\langle 1\rangle}$. But if $C \overset{*}{\leftarrow} \mathsf{G}(U_\lambda)$, we simulated $H^{\mathcal{P}}_I$. Thus, from any distinguisher for $H^{\mathcal{P}}_I$ and $H^{\mathcal{P}}_{I+\langle 1\rangle}$, we get a distinguisher for the PRG G with the same advantage.

B.5 Proof of Lemma 3

Let $\mathcal{P}' = \mathcal{P} \cup \{a\}$, where a is the element additionally contained in the level-$(\ell+1)$ sets $\mathcal{S}_{I+\langle 10_{n-\ell-1}\rangle}$, $\mathcal{S}_{I+\langle 20_{n-\ell-1}\rangle}$ between $\mathcal{S}_I$ and $\mathcal{S}_{I'}$ (i.e., the element $d - N/2^{\ell+1}$ in eq. (4)). Adding an element to the guessing set $\mathcal{P}$ simply decreases the probability of outputting 1 by a factor of q, thus with eq. (13):

$$\left|\Pr[H^{\mathcal{P}'}_I = 1] - \Pr[H^{\mathcal{P}'}_{I'} = 1]\right| = \left|\Pr[H^{\mathcal{P}}_I = 1]\cdot\tfrac{1}{q} - \Pr[H^{\mathcal{P}}_{I'} = 1]\cdot\tfrac{1}{q}\right| = \delta/q \ . \tag{20}$$

Now we can consider the sequence of consecutive level-$(\ell+1)$ sets $H^{\mathcal{P}'}_I, H^{\mathcal{P}'}_{I+\langle 10_{n-(\ell+1)}\rangle}, H^{\mathcal{P}'}_{I+\langle 20_{n-(\ell+1)}\rangle}, H^{\mathcal{P}'}_{I'}$. By eq. (20) and a standard hybrid argument, two of these can be distinguished with advantage $\delta/(3q)$ as required.

C Proof of Theorem 2

Our proof follows the one from [LW14], the main difference being that we check consistency of the oracle replies w.r.t. the answers of the fingerprint queries, whereas Lewko and Waters check consistency w.r.t. the public parameters.

Without loss of generality, we assume that the adversary makes exactly $q = N-1$ queries. We first construct an attacker A that $(t, \epsilon, N-1)$-breaks unpredictability of the prefix-constrained PRF for any given ϵ in some time t, which is not necessarily polynomial in λ. We then show how this attacker can be simulated in polynomial time.

An inefficient attacker. We start with constructing the hypothetical attacker A, which wins the unpredictability game with probability ϵ for any given ϵ. A first makes two queries for constrained keys which will serve as a "fingerprint" of the secret key used by the challenger. A then picks a random value x which cannot be evaluated with the obtained keys. Next, it queries keys with which it can evaluate the PRF at all points in the domain except x and its sibling (i.e., $(x_1, \dots, x_{N-1}, \overline{x}_N)$, where we let $\overline{x}_i := 1 - x_i$). It checks whether the received keys are consistent with the fingerprints and if so, it computes the PRF value at x under the secret key defined by the fingerprint (a step which may not be efficient); otherwise it aborts.

Phase 1 (Fingerprinting): A starts by making two constrained-key queries for $v_1 := (0, ?, \dots, ?)$ and $v_2 := (1, 0, ?, \dots, ?)$. Upon receiving the respective keys

$$(K_1, \{D_{i,b}\}_{i\in[2,N],\, b\in\{0,1\}}) \in \mathbb{G}_2 \times \mathbb{G}^{2(N-1)} \quad \text{and} \quad (K_2, \{D'_{i,b}\}_{i\in[3,N],\, b\in\{0,1\}}) \in \mathbb{G}_3 \times \mathbb{G}^{2(N-2)} \ ,$$

A aborts if $D_{i,b} \neq D'_{i,b}$ for any $i \in [3, N]$, $b \in \{0, 1\}$. A also aborts if $D_{2,0} = 1_{\mathbb{G}}$ (as in this case, K_2 does not uniquely fix the value $\alpha \cdot d_{1,1}$ of the challenger's secret key; see below). Otherwise, A stores the values $(K_1, K_2, \{D_{i,b}\}_{i,b})$.

Phase 2 (All-but-x): Next, A picks a random value $x' \in \{0, 1\}^{N-2}$, defines $x = 11\|x'$ and makes the following queries:

$$v_i = (1, 1, x_3, \dots, x_{i-1}, \overline{x}_i, ?, \dots, ?) \qquad \text{for } 3 \le i \le N-1 \ . \tag{21}$$

Note that the keys for v_1, v_2 and $\{v_i\}_{i\in[3,N-1]}$ let A evaluate the PRF at any point different from x and its sibling.

Let the i-th answer be $k_i = (K_i, \{D^{(i)}_{j,b}\}_{j\in[i+1,N],\, b\in\{0,1\}}) \in \mathbb{G}_{i+1} \times \mathbb{G}^{2(N-i)}$. A makes the following checks; if any of them fail, A aborts.

Check 1: For all $i \in [3, N-1]$, $j \in [i+1, N]$, $b \in \{0, 1\}$: $D^{(i)}_{j,b} \stackrel{?}{=} D_{j,b}$ (with $D_{j,b}$ obtained in Phase 1).

Check 2: For all $i \in [3, N-1]$: $e(K_i, D_{2,0}) \stackrel{?}{=} e(K_2, D_{2,1}, \dots, D_{i-1,x_{i-1}}, D_{i,\overline{x}_i})$.

These checks ensure that the keys are consistent with the fingerprint keys, in particular:

Lemma 5. *For all $i \in [2, N]$, $b \in \{0, 1\}$, let $d_{i,b} \in \mathbb{Z}_p$ be such that $D_{i,b} = g^{d_{i,b}}$ and let $d'_{1,0}$ and $d'_{1,1}$ be such that $K_1 = g_2^{d'_{1,0}}$ and $K_2 = g_3^{d'_{1,1}\cdot d_{2,0}}$. Since $d_{2,0} \neq 0$ (otherwise A aborted in Phase 1), these values are uniquely defined. Moreover, for all $i \in [3, N-1]$ for which Check 2 holds, we have*

$$K_i = g_{i+1}^{d'_{1,1}\cdot d_{2,1}\cdot d_{3,x_3}\cdots d_{i-1,x_{i-1}}\cdot d_{i,\overline{x}_i}} \ .$$

Proof. Let γ_i be such that $K_i = g_{i+1}^{\gamma_i}$ for $i \in [3, N-1]$. The check ensures that

$$g_{i+2}^{d'_{1,1}\cdot d_{2,0}\cdot d_{2,1}\cdots d_{i-1,x_{i-1}}\cdot d_{i,\overline{x}_i}} = e(K_2, D_{2,1}, \dots, D_{i-1,x_{i-1}}, D_{i,\overline{x}_i}) = e(K_i, D_{2,0}) = g_{i+2}^{\gamma_i\cdot d_{2,0}} \ ,$$

which, since $d_{2,0} \neq 0$, yields $\gamma_i = d'_{1,1} \cdot d_{2,1} \cdots d_{i-1,x_{i-1}}, d_{i,\overline{x}_i}$ and proves the claim. □

Phase 3 (Solve challenge): If all received keys passed the checks, A uses the values $d'_{1,1}, d_{2,1}, d_{3,x_3}, \dots,$ d_{N,x_N}, defined by the received keys as in Lemma 5 to compute the following (which we show is the PRF value at x):

$$y = g_{N+1}^{d'_{1,1}\cdot d_{2,1}\cdot \prod_{i=3}^{N} d_{i,x_i}}$$

(this step may not be efficient). A then flips a biased coin which yields $\beta = 1$ with probability $\xi := \epsilon \cdot (1 - \frac{1}{p})^{-1}$. If $\beta = 1$ then A outputs y; otherwise it aborts.

We show that A $(t, \epsilon, q = N-1)$-breaks unpredictability: A makes $N-1$ key queries. The value y is the PRF value of x: To see this, let $k^* = (\alpha^*, \{d^*_{i,b}\}_{i\in[N],\, b\in\{0,1\}})$ be the secret key chosen by A's challenger. If k^* is used to answer the fingerprint queries then with $d'_{1,0}, d'_{1,1}, d_{2,0}, d_{2,1}, \dots$ defined as in Lemma 5 we have

– For all $i \in [2, N]$, $b \in \{0,1\} : d_{i,b} = d^*_{i,b}$;
– and $d'_{1,0} = \alpha^* \cdot d^*_{1,0}$, $d'_{1,1} = \alpha^* \cdot d^*_{1,1}$.

This implies that $y = g_{N+1}^{\alpha^* d^*_{1,1} \cdot d^*_{2,1} \cdot \prod_{i=3,N} d^*_{i,x_i}} = F(k^*, x)$. Note also that constructing the function value this way (i.e., using the values defined by the first two replies) for any $11\|z \neq x$, except for $z = (x_1, \dots x_{N-1}, \overline{x}_N)$, leads to the same value as evaluating under the key k_j with $j = \min\{i | z_i \neq x_i\}$.

Finally, (with β, ξ as above)

$$\begin{aligned}\Pr\left[F(k^*, x) \leftarrow \mathsf{A}\right] &= \Pr\left[F(k^*, x) \leftarrow \mathsf{A} \,\middle|\, \beta = 1 \wedge d_{2,0} \neq 0\right] \xi\,(1 - \tfrac{1}{p}) \\ &\qquad + \Pr\left[F(k^*, x) \leftarrow \mathsf{A} \,\middle|\, \beta = 0 \vee d_{2,0} = 0\right] (1 - \xi(1 - \tfrac{1}{p})) \\ &= 1 \cdot \xi\big(1 - \tfrac{1}{p}\big) + 0 \cdot \big(1 - \xi\big(1 - \tfrac{1}{p}\big)\big) = \epsilon \ .\end{aligned}$$

Breaking the assumption using $\mathcal{R}$. Since the only random choices A makes are choosing x in Phase 2 and flipping the biased coin β at the end, we can assume that A draws its coins from a set $Z \times F$, where $Z = \{0,1\}^{N-2}$ is the set of possible strings x', and F are the coins used to choose the value β.

As we consider simple reductions, $\mathcal{R}$ runs A once in a straight-line fashion. We use $\mathcal{R}$ to create an algorithm B which solves Π. B does so by running the reduction $\mathcal{R}$ on a challenge and simulating the adversary A constructed above, but itself running in polynomial time. B starts by passing the received challenge instance $c \in C$ to $\mathcal{R}$, simulates Phase 1 of attacker A and stores the received values $(K_1, K_2, \{D_{i,b}\})$ (if A did not abort).

Next, B runs A's interaction with the reduction in the second phase τ times. (We will fix τ later so that it is polynomial in λ.) In each run B chooses fresh random coins for $\mathcal{R}$ and A. Thus, in the i-th interaction, B picks an independent random value $x^{(i)}$, makes the queries $v_j^{(i)}$ defined by $x^{(i)}$, as per (21), and performs the consistency checks. If all checks pass, B stores the received values $(K_3^{(i)}, \dots, K_{N-1}^{(i)})$; otherwise Run i is labeled an "aborting run". If all τ runs were aborting runs then B terminates and outputs a random guess.

If there was at least one non-aborting run, B chooses a random $z' \in \{0,1\}^{N-2}$, defines $z = 11\|z'$ and if for any $x^{(i)}$, we have $(z_1, \dots, z_{N-1}) = (x_1, \dots, x_{N-1})$ then B stops and outputs a random guess. Otherwise, B makes key queries for the values $v_3, \dots, v_{N-1}$ derived from z, as per (21). It checks consistency of the received keys and outputs a random guess if a check fails.

B picks a run i which was not aborting and lets $j \in [3, N-1]$ be the lowest index such that $x_j^{(i)} \neq z_j$. (This must exist, as otherwise, B would have aborted.) Since $v_j^{(i)}$ is a prefix of z, B can use the key received when querying $v_j^{(i)}$ to compute the PRF value y at z.

As we have argued above, the value computed this way is perfectly consistent with the information about the secret key fixed by the replies in Phase 1, meaning that B computes the same value as A would. B flips a biased coin β and with probability $\xi := \epsilon \cdot (1 - \frac{1}{p})^{-1}$ outputs y.

Analyzing B's success probability. Recall that C is the set of possible challenges for Π and that A's coins are drawn from $Z \times F$. Let $R = R_1 \times R_2$ be the set of possible random coins chosen by $\mathcal{R}$, where R_1 are the coins used up to the answering of A's fingerprint queries. Thus (c, r_1) determines the values $(K_1, K_2, \{D_{i,b}\})$, and (r_2, z, f) are the coins that are freshly chosen from $R_2 \times Z \times F$ in every rewind run.

We define W as the set of all tuples (c, r_1, r_2, z, f) such that when $\mathcal{R}$ is run with (r_1, r_2) on the challenge c and A is run with (z, f) then A does not abort and $\mathcal{R}$ solves the challenge c. We partition W into two sets according to a probability threshold ρ (which we will fix later) of a run not aborting when fixing c and r_1 and choosing the other coins freshly. Let T be the set of all (c, r_1, r_2, z, f) that lead to a run aborted by A; define

$$U := \left\{(c, r_1, r_2, z, f) \in W \,\middle|\, \Pr_{r_2', z', f'}\left[(c, r_1, r_2', z', f') \notin T\right] \geq \rho\right\} \text{ and } V := W \setminus U \ .$$

We first show the following lemma:

Lemma 6. $\Pr[V] \leq \rho$

Proof. First note that $W \subseteq \overline{T}$ (since for coins to be in W, A must not abort). This implies that

$$\begin{aligned} V &= \left\{(c, r_1, r_2, z, f) \in W \,\middle|\, \Pr_{r_2', z', f'}\left[(c, r_1, r_2', z', f') \notin T\right] < \rho\right\} \\ &\subseteq \left\{(c, r_1, r_2, z, f) \in W \,\middle|\, \Pr_{r_2', z', f'}\left[(c, r_1, r_2', z', f') \in W\right] < \rho\right\} \ . \end{aligned}$$

The lemma now follows because the probability of the latter set is strictly lower than ρ, since we have the following:

For any sets X, Y and $W \subseteq X \times Y$, the set $Z := \{(x, y) \in W \mid \Pr_{y' \leftarrow Y}[(x, y') \in W] < \rho\}$ has $\Pr[Z] < \rho$: Let $X_1 = \{x \in X \mid \Pr_{y' \leftarrow Y}[(x, y') \in W] < \rho\}$ and $X_2 = X \setminus X_1$. Then

$$\begin{aligned} \Pr[(x, y) \in Z] &= \sum_{x \in X} \Pr_{x' \leftarrow X}[x' = x] \Pr_{y' \leftarrow Y}[(x, y') \in Z] \\ &= \sum_{x \in X_1} \Pr_{x' \leftarrow X}[x' = x] \underbrace{\Pr_{y' \leftarrow Y}[(x, y') \in Z]}_{< \rho \ \text{(since } x \in X_1)} + \sum_{x \in X_2} \Pr_{x' \leftarrow X}[x' = x] \underbrace{\Pr_{y' \leftarrow Y}[(x, y') \in Z]}_{= 0 \ \text{(since } x \in X_2)} \end{aligned}$$

□

Next we define S as the set of all (c, r_1, r_2, z, f) so that $\mathcal{R}$ solves the challenge.

Lemma 7. *If Π is computationally hard then* $\Pr[T] \cdot \left|\Pr[S \mid T] - \frac{1}{2}\right|$ *is negligible in* λ.

Proof. Consider an adversary B′ which runs $\mathcal{R}$ and simulates A up to Phase 3. If A has not aborted until then, B′ outputs a random guess. If A aborted, it outputs whatever $\mathcal{R}$ outputs. B′ runs in polynomial time and solves the challenge with probability $\frac{1}{2}\big(1 - \Pr[T]\big) + \Pr[S \mid T] \Pr[T] = \frac{1}{2} + \Pr[T]\big(\Pr[S \mid T] - \frac{1}{2}\big)$.

Consider B″, which behaves like B′ except that it outputs the complement, i.e., in case A aborts, B″ outputs 0 if $\mathcal{R}$ outputs 1 and vice versa. B success probability is $\frac{1}{2}(1 - \Pr[T]) + (1 - \Pr[S \mid T]) \Pr[T] = \frac{1}{2} + \Pr[T](\frac{1}{2} - \Pr[S \mid T])$. Together this yields that $\Pr[T] \cdot |\Pr[S \mid T] - \frac{1}{2}|$ must be negligible. □

The probability that $\mathcal{R}$ solves the challenge when running A once is the probability of solving it when A does not abort plus the probability of solving it when A aborts:

$$\Pr[W] + \Pr[T] \Pr[S \mid T] = \tfrac{1}{2} + \delta \ .$$

We have $\Pr[W] = \Pr[U] + \Pr[V] < \Pr[U] + \rho$ (by Lemma 6), which together with Lemma 7 yields

$$\tfrac{1}{2} + \delta < \Pr[U] + \rho + \tfrac{1}{2} \Pr[T] + \mathrm{negl}(\lambda) \ . \tag{22}$$

Let $X^{(i)} \times F^{(i)}$ denote the set of coins for A and $R_2^{(i)}$ denote the set of coins used by $\mathcal{R}$ during the i-th run of $\mathcal{R}$ after answering the first two queries. Define T_i to be the set of those coins $(c, r_1, r_2^{(i)}, x^{(i)}, f^{(i)})$ that lead to an aborting run. Let E_i be the event that $z_j = x_j^{(i)}$ for all $j \in [3, N-1]$.

Consider a set of coins $\big(c, r_1, \{r_2^{(i)}, x^{(i)}, f^{(i)}\}_{i=1}^{\tau}, r_2, z, f\big)$ used by B during the overall computation, including the rewinds. Note that B aborts the computation if and only if

$$\forall\, i \in [1,\tau] : \big(c, r_1, r_2^{(i)}, x^{(i)}, f^{(i)}\big) \in T_i \quad \text{or} \quad \exists\, i \in [1,\tau]\ \forall\, j \in [3, N-1] : x_j^{(i)} = z_j \ ,$$

which corresponds to the coins being in the set $\bigcap_{i=1}^{\tau} T_i \cup \bigcup_{i=1}^{\tau} E_i$. On the other hand, if $(c, r_1, r_2, z, f) \in U \subseteq W$ and if B does not abort then it solves the challenge. Thus B wins with probability at least

$$\frac{1}{2}\Pr[T] + \sum_{(c,r_1,r_2,z,f)\in U} \Pr[(c, r_1, r_2, z, f)] \cdot \Big(1 - \Pr\Big[\bigcap_{i=1}^{\tau} T_i \cup \bigcup_{i=1}^{\tau} E_i \,\Big|\, (c, r_1, r_2, z, f)\Big]\Big) \ . \tag{23}$$

By the union bound, we have

$$\Pr\big[\textstyle\bigcup_{i=1}^{\tau} E_i \,\big|\, (c, r_1, r_2, z, f)\big] \le \tau 2^{-(N-3)} \ . \tag{24}$$

Since for fixed (c, r_1), the events T_i are independent, we have

$$\Pr\big[\textstyle\bigcap_{i=1}^{\tau} T_i \,\big|\, (c, r_1, r_2, z, f)\big] = \prod_{i=1}^{\tau} \Pr[T_i \,|\, (c, r_1, r_2, z, f)] \le (1-\rho)^{\tau} \ , \tag{25}$$

where the last inequality follows from (c, r_1, r_2, z, f) being in U. By the union bound and from equations (24) and (25), we have that (23) is greater than

$$\begin{aligned} \tfrac{1}{2}\Pr[T] + \Pr[U]\big(1 - \tau 2^{-(N-3)} - (1-\rho)^{\tau}\big) &\ge \tfrac{1}{2}\Pr[T] + \Pr[U] - 8\tau 2^{-N} - (1-\rho)^{\tau} \\ &\ge \tfrac{1}{2} + \delta - \rho - 8\tau 2^{-N} - (1-\rho)^{\tau} - \mathrm{negl}(\lambda) \ , \end{aligned}$$

where the last inequality follows from (22).

Setting $\rho = \frac{\delta}{4}$, and $\tau = \frac{\lambda}{\delta}$ (which is polynomial in λ), the last term equals

$$\tfrac{1}{2} + \tfrac{3}{4}\delta - 8\tfrac{\lambda}{\delta}2^{-N} - [(1-\tfrac{\delta}{4})^{\frac{1}{\delta}}]^{\lambda} - \mathrm{negl}(\lambda) \ = \ \tfrac{1}{2} + \tfrac{3}{4}\delta - 8\tfrac{\lambda}{\delta}2^{-N} - \mathrm{negl}(\lambda) \ , \tag{26}$$

since $(1-\frac{\delta}{4})^{\frac{1}{\delta}} < 1$ for all $\delta \in [0,1]$. We have showed that B's probability of solving $\mathit{\Pi}$ is at least (26), which by the assumption that $\mathit{\Pi}$ is computationally hard means that $\frac{3}{4}\delta - 8\frac{\lambda}{\delta}2^{-N}$ must be negligible in λ, and therefore δ must be exponentially small as a function of N.