Simple and More Efficient PRFs with Tight Security from LWE and Matrix-DDH[⋆]

Tibor Jager[1], Rafael Kurek[1], and Jiaxin Pan[2]

[1] Paderborn University, Paderborn, Germany
{tibor.jager, rafael.kurek}@upb.de
[2] Karlsruhe Institute of Technology, Karlsruhe, Germany
jiaxin.pan@kit.edu

Abstract. We construct efficient and tightly secure pseudorandom functions (PRFs) with only logarithmic security loss and short secret keys. This yields very simple and efficient variants of well-known constructions, including those of Naor-Reingold (FOCS 1997) and Lewko-Waters (ACM CCS 2009). Most importantly, in combination with the construction of Banerjee, Peikert and Rosen (EUROCRYPT 2012) we obtain the currently most efficient LWE-based PRF from a weak LWE-assumption with a much smaller modulus than the original construction. In comparison to the only previous construction with this property, which is due to Döttling and Schröder (CRYPTO 2015), we use a modulus of similar size, but only a *single* instance of the underlying PRF, instead of $\lambda \cdot \omega(\log \lambda)$ parallel instances, where λ is the security parameter. Like Döttling and Schröder, our security proof is only almost back-box, due to the fact that the number of queries made by the adversary and its advantage must be known *a-priori*.
Technically, we introduce *all-prefix* universal hash functions (APUHFs), which are hash functions that are (almost-)universal, even if *any* prefix of the output is considered. We give simple and very efficient constructions of APUHFs, and show how they can be combined with the augmented cascade of Boneh *et al.* (ACM CCS 2010) to obtain our results. Along the way, we develop a new and more direct way to prove security of PRFs based on the augmented cascade.

Keywords: Pseudorandom functions, LWE, MDDH, augmented cascade, tight security.

1 Introduction

A pseudorandom function (PRF) is a function $F : \mathcal{K} \times \mathcal{D} \to \mathcal{G}$ with the following security property. For random $k \xleftarrow{\$} \mathcal{K}$, the function $F(k, \cdot)$ is computationally indistinguishable from a random function $R(\cdot)$, given oracle access to either $F(k, \cdot)$ or $R(\cdot)$. PRFs are a foundational cryptographic primitive with countless applications, see [Gol01,Bel06,BG90,GGM84,Kra10] for example. While PRFs can be constructed generically from one-way functions (via pseudorandom generators) [GGM86], this generic construction is rather inefficient. Therefore we seek to construct efficient PRFs from as-weak-as-possible assumptions and with tight security proof.

⋆ The first and second authors were supported by DFG grant JA 2445/1-1. The third author was supported by DFG grant HO 4534/4-1.

Tight security. In a cryptographic security proof, we often consider an adversary $\mathcal{A}$ against a primitive like a PRF, and describe a reduction $\mathcal{B}$ that runs $\mathcal{A}$ as a subroutine to break some computational problem which is assumed to be hard. Let $(t_{\mathcal{A}}, \epsilon_{\mathcal{A}})$ and $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$ denote the running time and success probability of $\mathcal{A}$ and $\mathcal{B}$, respectively. Then we say that the reduction $\mathcal{B}$ *loses* a factor ℓ, if

$$\frac{t_{\mathcal{B}}}{\epsilon_{\mathcal{B}}} \geq \ell \cdot \frac{t_{\mathcal{A}}}{\epsilon_{\mathcal{A}}}$$

A reduction is usually considered "efficient", if ℓ is bounded by a polynomial in the security parameter. We say that a reduction is "tight", if ℓ is small. Our goal is to construct reductions $\mathcal{B}$ such that ℓ is as small as possible. Ideally we would like to have $\ell = O(1)$ constant, but there are many examples of cryptographic constructions and primitives where this is impossible to achieve [Cor02,KK12,HJK12,LW14,BJLS16].

State of the art. Many constructions of efficient number-theoretic PRFs, including the very general Matrix-DDH-based construction of [EHK$^+$17] (with the well-known algebraic constructions of Naor-Reingold [NR97] and Lewko-Waters [LW09] as special cases), as well as the LWE-based PRF of Banerjee, Peikert, and Rosen [BPR12], can in retrospect be seen as concrete instantiations of the *augmented cascade* framework of Boneh *et al.* [BMR10]. For these constructions, the size of the secret key and the loss in the security proof grow linearly[3] with the length n of the function input. Thus, efficiency and security both depend on the size of the input space. In order to extend the input space to $\{0,1\}^*$, one can generically apply a collision-resistant hash function $H : \{0,1\}^* \to \{0,1\}^n$, where $n = 2\lambda$ and λ denotes the security parameter, to the input before processing it in the PRF. This yields secret keys consisting of $n = O(\lambda)$ elements (where the concrete type of elements depends on the particular instantiation of the augmented cascade) and a security loss of $\ell = n = O(\lambda)$.

Contributions. We introduce *all-prefix universal hash functions* (APUHFs) as a special type of hash functions that are universal, even if the output of the hash function is truncated. We also describe a very simple and efficient construction, which is based on the hash function of Dietzfelbinger *et al.* [DHKP97], as well as a generic construction from pairwise independent hash functions with range $\{0,1\}^n$ for some $n \in \mathbb{N}$.

Then we show that by combining the augmented cascade with an APUHF, we are able to significantly improve both the asymptotic size of secret keys and the security loss of these constructions. Specifically, we achieve keys consisting of only a slightly super-logarithmic number of elements $m = \omega(\log \lambda)$ and an only logarithmic security loss $O(\log \lambda)$. Both the number of elements in the secret key and tightness are *independent* of the input size n, except for the key of the APUHF, which consists of n bits when instantiated with the APUHF of Dietzfelbinger *et al.* [DHKP97]. Based on this generic result, we then obtain simple variants of algebraic PRFs based on a large class of Matrix-DDH assumptions [EHK$^+$17], which include the PRFs of Naor and Reingold [NR97] and its generalization by Lewko and Waters [LW09] as special cases.

[3] As common in the literature, we count the number of elements here, not their bit size that increases with the security parameter.

Furthermore, we obtain a simple variant of the PRF of Banerjee, Peikert and Rosen [BPR12] (BPR). This PRF is based on the learning-with-errors (LWE) assumption [Reg05], and has the property that the required size of the LWE modulus depends on the length of the PRF input. More precisely, the lower bound on the LWE modulus p is exponential in the input length $n = \Theta(\lambda)$. We observe this in almost all the well-known LWE-based PRFs such as [BLMR13,BP14]. In order to improve efficiency and to base security on a weaker LWE assumption, it is thus desirable to make p as small as possible. We show that simply encoding the PRF input with an APUHF before processing it in the original BPR construction makes it possible to reduce the lower bound on the LWE modulus p from exponential to only slightly super-polynomial in the security parameter, which yields a weaker assumption and a significant efficiency improvement (see Section 5.2 for details). Furthermore, even for an arbitrary polynomially-bounded input size n, our construction requires to store only $m = \omega(\log \lambda)$ matrices, independent of the size n of the input space $\{0,1\}^n$, plus a single bitstring of length n when instantiated with the APUHF of Dietzfelbinger *et al.* [DHKP97]. In contrast, the original construction from [BPR12] requires $\Theta(n)$ matrices.

A similar improvement of the LWE modulus p was achieved by a different BPR variant due to Döttling and Schröder in [DS15], via a technique called *on-the-fly adaptation*. However, their construction requires to run $\lambda \cdot \omega(\log \lambda)$ copies of the BPR PRF in parallel, while ours requires only a single copy plus an APUHF. Thus, our approach is significantly more efficient, and also more direct, as it essentially corresponds to the original BPR function, except that an APUHF is applied to the input. This simplicity gives not only a useful conceptual perspective on the construction of tightly secure PRFs, but it also makes schemes easier to implement securely.

Another advantage of our approach is that the resulting PRF construction is extremely simple. It is essentially identical to the augmented cascade from [BMR10], except that an APUHF h is applied to the input before it is processed by the PRF. More precisely, let $\hat{F}^m$ be a PRF that is constructed from an m-fold application of an underlying function F via the augmented cascade construction from [BMR10]. Then our construction $\hat{F}(K, x)$ has the form

$$\hat{F}(K, x) := \hat{F}^m(s, h(x))$$

where the key of our new function is a tuple $K = (s, h)$ consisting of a random key s for the augmented cascade construction and a random function $h \overset{\$}{\leftarrow} \mathcal{H}$ from a family $\mathcal{H} = \{h : \{0,1\}^n \rightarrow \{0,1\}^m\}$ of APUHFs.

We remark that we require an additional property called *perfect one-time security*("1-uniformity") of the underlying function F of the augmented cascade, and thus technically our variant of [BMR10] is slightly less general. However, this is a minor restriction, as we show that this property is satisfied by all known instantiations of the augmented cascade. Furthermore, our security proof assumes that the reduction "knows" sufficiently close approximations of the number of queries Q and the advantage $\epsilon_{\mathcal{A}}$ of the adversary. Thus, the proof shows how such non-black-box knowledge can be used to achieve more efficient PRFs with short keys and very tight security from weaker assumptions.

Technical approach. Technically, our argument is inspired by the construction of adaptively-secure PRFs from non-adaptively secure ones by Berman and Haitner [BH12].

Essentially, an augmented cascade PRF with m-bit input is a function $\hat{F}^m : S^m \times K \times \{0,1\}^m \to K$ with key space $S^m \times K$. In the sequel, let $(s_1, \ldots, s_m, k) \in S^m \times K$ be a key for $\hat{F}^m$ and $h : \{0,1\}^n \to \{0,1\}^m$. For a string $a \in \{0,1\}^m$ we write $a_{v:w}$ to denote the substring $(a_v, \ldots, a_w) \in \{0,1\}^{w-v+1}$ of a. Let j be an integer with $j \leq m$ (we will explain later how to choose j in a suitable way).

We start from the observation that, for each $j \in \{1, \ldots, m\}$, we can implement an augmented cascade PRF $\hat{F}^m$ equivalently as a two-step algorithm, which proceeds as follows.

1. In the first step, the function $\hat{F}^m$ processes only the first j bits $h(x)_{1:j} \in \{0,1\}^j$ of $h(x)$, to compute an intermediate value k_x that depends only on the first j bits of $h(x)$:
$$k_x = \hat{F}^j((s_1, ..., s_j), k, h(x)_{1:j})$$
2. Then the remaining $m - j$ bits are processed, starting from k_x, by computing
$$y = \hat{F}^{m-j}((s_{j+1}, ..., s_m), k_x, h(x)_{j+1:m})$$

The resulting function is identical to the function $\hat{F}^m$, so this is merely a specific way to implement $\hat{F}^m$, which will be particularly useful to describe our approach.

To explain how we prove security, let $x^{(1)}, \ldots, x^{(Q)}$ denote the sequence of pairwise distinct oracle queries issued by the adversary in the PRF security experiment, and suppose for now that it holds $h(x^{(u)})_{1:j} \neq h(x^{(v)})_{1:j}$ for $u \neq v$. Our goal is to show that then the security of $\hat{F}^m$ is implied by the security of $\hat{F}^j$, which is a PRF with shorter input. Intuitively, this holds due to the following two-step argument.

1. We replace $\hat{F}^j$ with a random function R, which is computationally indistinguishable thanks to the security of $\hat{F}^j$. Note that now the intermediate value $k_x = R(h(x)_{1:j})$ is an independent random value for each oracle query made by the adversary, because we assume $h(x^{(u)})_{1:j} \neq h(x^{(v)})_{1:j}$ for $u \neq v$.
2. Next we argue that now also $\hat{F}^m$ is distributed exactly like a random function. We achieve this by identifying an additional property required from $\hat{F}^{m-j}$ that we call *perfect one-time security*. This property guarantees that
$$\Pr_{k_x \xleftarrow{\$} K}\left[\hat{F}^{m-j}((s_{j+1}, ..., s_m), k_x, h(x)_{j+1:m}) = y\right] = \frac{1}{|K|}$$
for all $(s_{j+1}, ..., s_m), h(x)_{j+1:m}, y) \in S^{m-j} \times \{0,1\}^{m-j} \times K$. This is sufficient to show that indeed now the function
$$\hat{F}^{m-j}((s_{j+1}, ..., s_m), R(h(x)_{1:j}), h(x)_{j+1:m})$$
is a random function, because we have $h(x^{(u)})_{1:j} \neq h(x^{(v)})_{1:j}$ for $u \neq v$.

It remains to ensure that $h(x^{(u)})_{1:j} \neq h(x^{(v)})_{1:j}$ holds for all $u \neq v$ with "sufficiently large" probability and for some "sufficiently small" value of j. Here we use the all-prefix universal hash function, in combination with an argument which on a high level follows similar proofs from [BH12] and [DS15]. The main difference is that we use the all-prefix universality to argue that setting $j := \lceil \log(2Q^2/\epsilon_\mathcal{A}) \rceil = O(\log \lambda)$, where Q is the number of oracle queries made by the adversary in the PRF security experiment and $\epsilon_\mathcal{A}$ is its advantage, is sufficient to guarantee that $h(x^{(u)})_{1:j} \neq h(x^{(v)})_{1:j}$ holds with sufficiently large probability for all $u \neq v$.

Note that we have $j = O(\log \lambda)$, so that we only have to require security of a "short-input" augmented cascade $\hat{F}^j$ with $j = O(\log \lambda)$. For our algebraic instantiations based on Matrix-DDH problems, this yields tightness with a security loss of only $O(\log \lambda)$. For our application to the LWE-based PRF of Banerjee, Peikert and Rosen [BPR12], this yields that we have to require only a weaker LWE assumption. Furthermore, since we need only that $m \geq j$ holds for all possible values of j, and we have $j = \lceil \log(2Q^2/\epsilon_\mathcal{A}) \rceil = O(\log \lambda)$, it is sufficient to set $m = \omega(\log \lambda)$ slightly super-logarithmic, which yields short secret keys and efficient evaluation for all instantiations.

Our proof technique, in particular the perfect one-time security property, can also be seen as an alternative and more direct way of proving the augmented cascade construction secure, while Boneh et al. used the somewhat more complex q-parallel security of the underlying PRF.

Why all-prefix *universal hash functions?* We stress that we need an *all-prefix* universal hash function, which works for any possible prefix length j. This is necessary to make the construction and the security proof independent of particular values Q and $\epsilon_\mathcal{A}$ of a particular adversary, because j depends on these values via the definition $j = \lceil \log(2Q^2/\epsilon_\mathcal{A}) \rceil$. *All-prefix* universality guarantees basically that a suitable value of j exists for any efficient adversary. This is also required to achieve tightness. See Section 4.7 for further discussion.

More related work. There were several other works about the domain extension of PRFs. The first one is due to Levin [Lev87]. It shows that larger inputs can be hashed with a universal hash function if the underlying PRF has a sufficiently large domain. Otherwise it is vulnerable to the so called "birthday attack". The framework of Jain, Pietrzak, and Tentes [JPT12] works for small domains, but has a rather lossy security proof and is not very efficient, as it needs $\mathcal{O}(\log q)$ invocations of the underlying pseudo-random generator (PRG), where q is the upper bound of queries to the PRF. Additionally, as the authors already mention, it seems not to work for number-theoretic PRFs like the Naor-Reingold PRF. It was revisited by Chandran and Garg [CG14]. Bernam *et al.* show how to circumvent the "birthday attack" using Cuckoo Hashing [BHKN13] via two invocations of the original PRF.

2 Preliminaries

Let $\lambda \in \mathbb{N}$ denote a security parameter. All our results are in the asymptotic setting, that is, we view all expressions involving λ as functions in λ. This includes the running time $t_\mathcal{A} = t_\mathcal{A}(\lambda)$ and success probability $\epsilon_\mathcal{A} = \epsilon_\mathcal{A}(\lambda)$ of adversaries, even though we occasionally omit λ in this case to simplify our notation. Similarly, all algorithms implicitly receive the security parameter 1^λ as their first input. We say that an algorithm is *efficient*, if it runs in (probabilistic) polynomial time in λ.

Notation. If A is a finite set, then we write $a \xleftarrow{\$} A$ to denote the action of sampling a uniformly random element a from A. If A is a probabilistic algorithm, then $a \xleftarrow{\$} A(x)$ denotes the action of running $A(x)$ on input x with uniform coins and output a. For $v, w \in \mathbb{N}$ and $v < w$, we write $[\![v, w]\!] := \{v, \ldots, w\} \subset \mathbb{N}$ to denote the interval of positive integers from v to w, and set $[\![w]\!] := \{1, \ldots, w\} \subset \mathbb{N}$. For a bit string $a = (a_1, \ldots, a_n) \in \{0,1\}^n$ and $v, w \in [\![n]\!]$ with $v \leq w$, we write $a_{v:w}$ to denote the substring $(a_v, \ldots, a_w)$ of a, and a_i to denote the i-th bit a_i.

2.1 Pseudorandom Functions

Let $\mathcal{K}, \mathcal{D}$ be sets such that there is an efficient algorithm that samples uniformly random elements $k \xleftarrow{\$} \mathcal{K}$. Let $F : \mathcal{K} \times \mathcal{D} \to \mathcal{G}$ be an efficiently computable function. For an adversary $\mathcal{A}$ define the following security experiment $\mathsf{Exp}^{\mathsf{prf}}_{\mathcal{A},F}(\lambda)$.

1. The experiment generates a random key $k \xleftarrow{\$} \mathcal{K}$ and tosses a coin $b \xleftarrow{\$} \{0,1\}$.
2. The experiment provides adversary $\mathcal{A}^{\mathcal{O}}(1^\lambda)$ with an oracle $\mathcal{O}$ which takes as input $x \in \mathcal{D}$ and responds as follows.

$$\mathcal{O}(x) = \begin{cases} F(k,x) & \text{if } b = 1 \\ R(x) & \text{if } b = 0 \end{cases}$$

   where $R : \mathcal{D} \to \mathcal{G}$ is a random function. When the adversary terminates and outputs a bit b', then the experiment outputs 1 if $b = b'$, and 0 otherwise.

Let $x_1, \ldots, x_Q \in \mathcal{D}$ be the sequence of queries issued by $\mathcal{A}$ throughout the security experiment. We assume that we always have $Q \geq 1$, as otherwise the output of $\mathcal{A}$ is independent of b. Furthermore, we assume that $\mathcal{A}$ never issues the same query twice. More precisely, we assume $x_u \neq x_v$ for $u \neq v$. This is without loss of generality, since both $F(k, \cdot)$ and $R(\cdot)$ are deterministic functions.

Definition 1. *We say that adversary $\mathcal{A}$ $(t_\mathcal{A}, \epsilon_\mathcal{A}, Q)$-breaks the pseudorandomness of F, if $\mathcal{A}$ runs in time $t_\mathcal{A}$, issues Q queries in the PRF security experiment, and*

$$\Pr\left[\mathsf{Exp}^{\mathsf{prf}}_{\mathcal{A},F}(\lambda) = 1\right] \geq 1/2 + \epsilon_\mathcal{A}$$

2.2 (Almost-)Universal Hash Functions

Let us first recall the standard definition of universal hash functions.

Definition 2 **([CW79]).** *A family $\mathcal{H}$ of hash functions mapping finite set $\{0,1\}^n$ to finite set $\{0,1\}^m$ is* universal, *if for all $x, x' \in \{0,1\}^n$ with $x \neq x'$ holds that*

$$\Pr_{h \xleftarrow{\$} \mathcal{H}}\left[h(x) = h(x')\right] \leq 2^{-m}.$$

We will also consider *almost-universal* hash functions, as defined below.

Definition 3. *A family $\mathcal{H}$ of hash functions mapping finite set $\{0,1\}^n$ to finite set $\{0,1\}^m$ is* almost-universal, *if for all $x, x' \in \{0,1\}^n$ with $x \neq x'$ holds that*

$$\Pr_{h \xleftarrow{\$} \mathcal{H}} [h(x) = h(x')] \leq 2^{-m+1}.$$

Universal and almost-universal hash functions can be constructed efficiently and without additional complexity assumptions, see e.g. [CW79,DHKP97,IKOS08].

3 All-Prefix Universal Hash Functions

In this section, we define all-prefix almost universal hash functions and describe two constructions. The first one is based on the almost-universal hash function of Dietzfelbinger *et al.* [DHKP97], and yields an all-prefix almost-universal hash function. The second one is based on pairwise independent hash functions with suitable range, and yields an all-prefix universal hash function.

3.1 Definitions

Recall that for a bit string $a = (a_1, \ldots, a_n) \in \{0,1\}^n$ and $v, w \in [\![n]\!]$ with $v \leq w$, we write $a_{v:w} := (a_v, \ldots, a_w)$.

Definition 4. *Let $\mathcal{H}$ be a family of hash functions mapping $\{0,1\}^n$ to $\{0,1\}^m$. We say that $\mathcal{H}$ is a family of* all-prefix universal hash functions, *if for all $x, x' \in \{0,1\}^n$ with $x \neq x'$ and all $w \in [\![m]\!]$ holds that*

$$\Pr_{h \xleftarrow{\$} \mathcal{H}} [h(x)_{1:w} = h(x')_{1:w}] \leq 2^{-w}.$$

Note that all-prefix universality essentially means that for all prefixes of length w the truncation of h to its first w bits $h(x)_{1:w}$ is a universal hash function. We also define the slightly weaker notion of all-prefix *almost*-universality.

Definition 5. *Let $\mathcal{H}$ be a family of hash functions mapping $\{0,1\}^n$ to $\{0,1\}^m$. We say that $\mathcal{H}$ is a family of* all-prefix almost-universal hash functions *(APUHFs), if for all $x, x' \in \{0,1\}^n$ with $x \neq x'$ and all $w \in [\![m]\!]$ holds that*

$$\Pr_{h \xleftarrow{\$} \mathcal{H}} [h(x)_{1:w} = h(x')_{1:w}] \leq 2^{-w+1}.$$

3.2 First Construction (Almost-Universal)

We construct a simple and efficient APUHF family based on the almost-universal hash function of Dietzfelbinger *et al.* [DHKP97], which is defined as follows. Let $m, n \in \mathbb{N}$ with $m \leq n$. Let

$$\mathcal{H}_{n,m} := \{h_a : a \in [\![2^n - 1]\!] \text{ and } a \text{ is odd}\} \tag{1}$$

be the family of hash functions, which for $x \in \mathbb{Z}_{2^n}$ is defined as

$$h_a(x) := (ax \bmod 2^n) \operatorname{div} 2^{n-m}, \tag{2}$$

Before we prove that this function is all-prefix almost-universal, we first state the following lemma of Dietzfelbinger *et al.* [DHKP97].

Lemma 1 **([DHKP97]).** *Let n and m be positive integers with $m \in [\![n]\!]$. If $x, y \in \mathbb{Z}_{2^n}$ are distinct and $h_a \in \mathcal{H}_{n,m}$ is chosen at random, then*

$$\Pr[h_a(x) = h_a(y)] \leq 2^{-m+1}$$

Thus, $\mathcal{H}_{n,m}$ is a family of almost-universal hash functions in the sense of Definition 3.

All-prefix almost-universality of $\mathcal{H}_{n,m}$. Now we prove that the hash function family $\mathcal{H}_{n,m}$ of Dietzfelbinger *et al.* [DHKP97] is not only almost-universal, but also satisfies the stronger property of *all-prefix* almost-universality.

Theorem 1. *$\mathcal{H}_{n,m}$ is a family of all-prefix almost-universal hash functions in the sense of Definition 5.*

PROOF. Let ω, m, n be any positive integers with $\omega \leq m \leq n$. Note that if $h_a(\cdot)$ is a function in $\mathcal{H}_{n,m}$ then $h_a(\cdot)_{1:\omega}$ is a function in $\mathcal{H}_{n,\omega}$. Further note that Lemma 1 holds for all $\omega \in [\![n]\!]$, which proves the claim. $\square$

In the sequel, we will sometimes write h instead of h_a, when it is clear from the context that h is be chosen uniformly random from $\mathcal{H}_{n,m}$.

3.3 Second Construction (Universal)

While the almost-universal construction from Section 3.2 is already sufficient for all our applications, it is natural to ask whether also all-prefix *universal* hash functions (not almost-universal) can be constructed. We will show that each pairwise-independent family of hash functions with range $\{0,1\}^n$ is also a family of all-prefix universal hash functions. To this end, let us first recall the notion of pairwise independent hash functions.

Definition 6. *Let $\mathcal{H}$ be a family of hash functions with domain $\{0,1\}^n$ and range $\{0,1\}^m$. We say that $\mathcal{H}$ is* pairwise independent*, if for all $x, x' \in \{0,1\}^n$ with $x \neq x'$ and all $y, z \in \{0,1\}^m$ holds that*

$$\Pr_{h \xleftarrow{\$} \mathcal{H}}[h(x) = y \wedge h(x') = z] = 2^{-2m}.$$

We first show that pairwise independence implies all-prefix pairwise independence, which is defined below. Then we show that this implies all-prefix universality.

Let us write x_i to denote the i-th bit of the bit string x.

Definition 7. *Let $\mathcal{H}$ be a family of hash functions mapping $\{0,1\}^n$ to $\{0,1\}^m$. We say that $\mathcal{H}$ is* all-prefix pairwise independent*, if for all $x, x' \in \{0,1\}^n$ with $x \neq x'$ and all $y, z' \in \{0,1\}^m$ holds that*

$$\Pr_{h \xleftarrow{\$} \mathcal{H}} [h(x)_{1:w} = y_{1:w} \wedge h(x')_{1:w} = z_{1:w}] = 2^{-2w}$$

for all $w \in [\![m]\!]$.

Lemma 2. *If $\mathcal{H}$ is pairwise independent, then it is also all-prefix pairwise independent.*

Proof. We have

$$\begin{aligned}
&\Pr_{h \xleftarrow{\$} \mathcal{H}} [h(x)_{1:j} = y_{1:j} \wedge h(x')_{1:j} = z_{1:j}] \\
&= \Pr_{h \xleftarrow{\$} \mathcal{H}} \left[\left(\bigcup_{y' \in \{0,1\}^{m-j}} h(x) = (y_{1:j} \parallel y') \right) \wedge \left(\bigcup_{z' \in \{0,1\}^{m-j}} h(x') = (z_{1:j} \parallel z') \right) \right] \\
&= \sum_{y' \in \{0,1\}^{m-j}} \sum_{z' \in \{0,1\}^{m-j}} \Pr_{h \xleftarrow{\$} \mathcal{H}} [h(x) = (y_{1:j} \parallel y') \wedge h(x') = (z_{1:j} \parallel z')] \\
&= \sum_{y' \in \{0,1\}^{m-j}} \sum_{z' \in \{0,1\}^{m-j}} \frac{1}{2^{2m}} = \frac{2^{m-j} \cdot 2^{m-j}}{2^{2m}} = \frac{1}{2^{2j}}.
\end{aligned}$$

□

Now it remains to show that all-prefix pairwise independence implies all-prefix universality.

Lemma 3. *If $\mathcal{H}$ is all-prefix pairwise independent, then it is also all-prefix universal.*

Proof. It holds that

$$\begin{aligned}
\Pr_{h \xleftarrow{\$} \mathcal{H}} [h(x)_{1:j} = h(x')_{1:j}] &= \sum_{y_{1:j} \in \{0,1\}^j} \Pr_{h \xleftarrow{\$} \mathcal{H}} [h(x)_{1:j} = y_{1:j} \wedge h(x')_{1:j} = y_{1:j}] \qquad (3) \\
&= \sum_{y_{1:j} \in \{0,1\}^j} \frac{1}{2^{2j}} = \frac{1}{2^j},
\end{aligned}$$

where (3) holds because of Lemma 2. □

Example instantiation. Let $n \in \mathbb{N}$ and let

$$\mathcal{H}_n := \{h_{a,b} : a, b \in \{0,1\}^n\}$$

be the family of hash functions

$$h_{a,b} : GF(2^n) \to GF(2^n); x \mapsto ax + b,$$

where the arithmetic operations are in $GF(2^n)$. Since it is well-known that $\mathcal{H}_n$ is pairwise independent we leave the following theorem without proof.

```
Input: Key $(s_1, ..., s_m, k_0) \in S^m \times K$ and $(x_1, ..., x_m) \in X^m$
For $i = 1, ..., m$ :
    $k_i \leftarrow F((s_i, k_{i-1}), x_i)$
Return $k_m$.
```

Fig. 1. Definition of function $\hat{F}^m$ of Boneh *et al.* [BMR10].

Theorem 2. *$\mathcal{H}_n$ is a family of all-prefix universal hash functions.*

Note that in the explicit construction of $GF(2^n)$ the choice of the irreducible polynomial has big impact on the efficiency of the arithmetic operations.

4 Augmented Cascade PRFs with Tighter Security

In this section, we show that APUHFs enable the instantiation of augmented cascade PRFs [BMR10] with shorter keys of sligtly super-logarithmic size $\omega(\log \lambda)$. The security proof loses only a factor $O(\log \lambda)$, independent of the input size of the PRF, assuming that (reasonably close bounds) on the number of queries Q and the success probability $1/2 + \epsilon_{\mathcal{A}}$ of the PRF adversary $\mathcal{A}$ are known *a priori*. In contrast, the loss of the previous security proof of [BMR10] is linear in the input size of the PRF (which is usually linear in λ), but does not assume any *a priori* knowledge about $\mathcal{A}$.

4.1 Augmented Cascade PRFs

Boneh *et al.* [BMR10] showed how to construct a PRF

$$\hat{F}^m : (S^m \times K) \times X^m \to K$$

with key space $(S^m \times K)$ and input space X from an augmented cascade of functions

$$F : (S \times K) \times X \to K$$

The augmented cascade construction is described in Figure 1. Boneh *et al.* [BMR10] prove that $\hat{F}^m$ is a secure PRF, if F is *parallel secure* in the following sense.

Definition 8 ([BMR10]). *For a function $F : (S \times K) \times X \to K$ define $F^{(Q)}$ as the function*

$$F^{(Q)} : (S \times K^Q) \times (X \times [\![Q]\!]) \to K \qquad ((s, k_1, ..., k_q), (x, i)) \mapsto F((s, k_i), x) .$$

*We say that $\mathcal{A}$ $(t_{\mathcal{A}}, \epsilon_{\mathcal{A}}, Q)$-breaks the Q-*parallel security *of $F : (S \times K) \times X \to K$, if it $(t_{\mathcal{A}}, \epsilon_{\mathcal{A}}, Q)$-breaks the pseudorandomness of $F^{(Q)}$ in the sense of Definition 1.*

Theorem 3 ([BMR10]). *From each adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, \epsilon_{\mathcal{A}}, Q)$-breaks the pseudorandomness of $\hat{F}^m$, one can construct an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}}, Q)$-breaks the Q-parallel security of $F^{(Q)}$ with*

$$t_{\mathcal{B}} = \Theta(t_{\mathcal{A}}) \qquad \text{and} \qquad \epsilon_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{A}}}{m}$$

Note that the security loss of this construction is linear in the length m of the input of function $\hat{F}^m$.

4.2 The Augmented Cascade with Encoded Input

We consider augmented cascade PRFs which are almost identical to the construction of Boneh *et al.* [BMR10], except that we apply an all-prefix almost-universal hash function to the input before processing it in the augmented cascade, and show that this enables a tighter security proof. We consider the special case with input space $X = \{0,1\}$, which encompasses the MDDH-based construction of Escala *et al.* [EHK$^+$17] and thus includes in particular both the instantiations of Naor-Reingold [NR97] and Lewko-Waters [LW09].

Let $\mathcal{H}_{n,m}$ be a family of all-prefix almost-universal hash functions according to Definition 5, and let $F : (S \times K) \times \{0,1\} \to K$ be a function. We define the corresponding augmented cascade PRF with $\mathcal{H}_{n,m}$-encoded input as the function

$$\begin{aligned}\hat{F}^{\mathcal{H}_{n,m}} &: S^m \times K \times \mathcal{H}_{n,m} \times \{0,1\}^n \to K \\ ((s_1, ..., s_m), k, h, x) &\mapsto \hat{F}^m((s_1, ..., s_m), k, h(x)) \end{aligned} \tag{4}$$

where $\hat{F}^m$ is the augmented cascade construction of Boneh *et al.* [BMR10], applied to F as described in Figure 1.

Remark 1. Note that evaluating the PRF requires only m recursions in the augmented cascade, and that, accordingly, the secret key consists of only m elements and the description of h, while the input size can be any polynomial number of n bits, with possibly $n \gg m$. We will later show that it suffices to set $m = \omega(\log \lambda)$ slightly super-logarithmic, thanks to the input encoding with an all-prefix almost-universal hash function. Also the security loss of this construction is only $O(\log \lambda)$ and independent of the size of the input n.

4.3 Preparation for the Security Proof

In this section we describe a few technical observations which will simplify the security proof. Furthermore, we define *perfect one-time security* as an additional property of a function $F(s, x, k)$, which will also be required for the proof. We will argue later that the Matrix-DDH-based instantiations of the augmented cascade of [EHK$^+$17], including the functions of Naor-Reingold [NR97] and Lewko-Waters [LW09], all satisfy this additional notion. Moreover, we will show that the LWE-based PRF of [BPR12] can be viewed as an augmented cascade and it is perfectly one-time secure.

An observation about the augmented cascade. The following observation will be useful to follow the security proof more easily. Suppose we want to compute

$$z = \hat{F}^m((s_1, ..., s_m), k, h(x))$$

then, due to the recursive definition of $\hat{F}^m$, we can equivalently proceed in the following two steps.

1. Let $i \in [\![m]\!]$. We first process the first i bits $h(x)_{1:i}$ of $h(x)$ with $(s_1, \ldots, s_i, k)$, and compute and "intermediate key" k_x as

$$k_x := \hat{F}^i((s_1, \ldots, s_i), k, h(x)_{1:i})$$

2. Then we process the remaining $m - i$ bits $h(x)_{i+1:m}$ of $h(x)$ with the remaining key elements $(s_{i+1}, \ldots, s_m, k_x)$ by computing

$$z = \hat{F}^{m-i}((s_{i+1}, ..., s_m), k_x, h(x)_{i+1:m})$$

We formulate this observation as a lemma.

Lemma 4. *For all $i \in [\![m]\!]$, we have*

$$\hat{F}^m((s_1, ..., s_m), k, h(x)) = \hat{F}^{m-i}((s_{i+1}, ..., s_m), k_x, h(x)_{i+1:m})$$

where $k_x := \hat{F}^i((s_1, \ldots, s_i), k, h(x)_{1:i})$.

Perfect One-Time Security. We will furthermore require an additional security property of F, which we call *perfect one-time security*, and show that this property is satisfied by all instantiations of function F considered in this section. We demand that $F(s, x, k)$ is identically distributed to a random function $R(x)$, if it is only evaluated once. This must hold over the uniformly random choice $k \xleftarrow{\$} K$, and for any $s \in S$ and $x \in \{0, 1\}$.

Definition 9. *We say that a function $F : S \times K \times \{0,1\}^m \to K$ is* perfectly one-time secure*, if*

$$\Pr_{k \xleftarrow{\$} K}[F(s, k, x) = k'] = \frac{1}{|K|}$$

for all $(s, x, k') \in S \times \{0,1\}^m \times K$.

Perfect one-time security basically guarantees uniformity of the hash function, if it is evaluated only once ("1-uniformity").

The following lemma follows directly from Definition 9. It will be useful to prove security of our variant of the augmented cascade.

Lemma 5. *Let $m \in \mathbb{N}$ and $F : S \times K \times \{0,1\} \to K$ be perfectly one-time secure. Then the augmented cascade $\hat{F}^m$ constructed from F is also perfectly one-time secure. That is*

$$\Pr_{k \xleftarrow{\$} K}\left[\hat{F}^m((s_1, ..., s_m), k, x) = k'\right] = \frac{1}{|K|}$$

for all $((s_1, ..., s_m), k', x) \in S^m \times K \times \{0,1\}^m$.

PROOF. For a uniformly random chosen k it holds that $\Pr[F(s_1, k, x_1) = k_1] = \frac{1}{|K|}$ for all $(s_1, k, x_1) \in S \times K \times \{0, 1\}$ because of the perfect one-time security of F. Thus the input for the second iteration stays uniformly random. Due to the recursive construction executing all the following iterations will keep this distribution, which gives us the perfect one-time security of $\hat{F}^m$. □

4.4 Security Proof

Now we are ready to prove the following theorem.

Theorem 4. *Let $m = \omega(\log \lambda)$ be (slightly) super-logarithmic, $\mathcal{H}_{n,m}$ be a family of all-prefix almost universal hash functions and F be perfectly one-time secure.*

From each adversary $\mathcal{A}$ that $(t_\mathcal{A}, \epsilon_\mathcal{A}, Q)$-breaks the pseudorandomness of $\hat{F}^{\mathcal{H}_{n,m}}$ with $Q/\epsilon_\mathcal{A} = \mathsf{poly}(\lambda)$ for some polynomial poly, we can construct an adversary $\mathcal{B}$ that $(t_\mathcal{B}, \epsilon_\mathcal{B}, Q)$-breaks the pseudorandomness of $\hat{F}^j$, where

$$j = O(\log \lambda) \qquad \textit{and} \qquad t_\mathcal{B} = \Theta(t_\mathcal{A}) \qquad \textit{and} \qquad \epsilon_\mathcal{B} \geq \epsilon_\mathcal{A}/2$$

Proof. In the sequel let $j = j(\lambda)$ be defined such that

$$j := \lceil \log(2Q^2/\epsilon_\mathcal{A}) \rceil \tag{5}$$

Observe that we have $j(\lambda) \leq m(\lambda)$ for sufficiently large λ, because the fact that we have $Q/\epsilon_\mathcal{A} = \mathsf{poly}(\lambda)$ for some polynomial poly and $j < \log(2Q^2/\epsilon_\mathcal{A}) + 1$ together yield that $j = O(\log \lambda)$, while we have $m = \omega(\log \lambda)$.

Remark 2. Note that although we have $j = O(\log(2Q^2/\epsilon_\mathcal{A})) = O(\log \lambda)$, the constant hidden in the big-O notation depends on the adversary.

We describe a sequence of games, where Game 0 is the original PRF security experiment, and in the last game the probability that the experiment outputs 1 is $1/2$, such that no adversary can have any advantage. Let X_i denote the event that the experiment outputs 1 in Game i, and let $\mathcal{O}_i$ denote the oracle provided by the experiment in Game i.

Game 0. This is the original security experiment. In particular, we have

$$\mathcal{O}_0(x) = \begin{cases} \hat{F}^{\mathcal{H}_{n,m}}((s_1, ..., s_m), k, h, x) & \text{if } b = 1 \\ R(x) & \text{if } b = 0 \end{cases}$$

where R is a random function. Therefore, by definition, it holds that

$$\Pr[X_0] = 1/2 + \epsilon_\mathcal{A}$$

Game 1. We change the way how the oracle implements function $\hat{F}^{\mathcal{H}_{n,m}}$. That is, we modify the behaviour of $\mathcal{O}_1$ in case $b = 1$, while in case $b = 0$ oracle $\mathcal{O}_1$ proceeds identical to $\mathcal{O}_0$. Recall that

$$\hat{F}^{\mathcal{H}_{n,m}}((s_1, ..., s_m), k, h, x) = \hat{F}^m((s_1, ..., s_m), k, h(x))$$

$\mathcal{O}_1$ implements this function in a specific way. Using the observation from Lemma 4, it computes $\hat{F}^m((s_1, ..., s_m), k, h(x))$ in two steps:

1. $k_x := \hat{F}^j((s_1, \ldots, s_j), k, h(x)_{1:j})$,
2. $z := \hat{F}^{m-j}((s_{j+1}, ..., s_m, k_x, h(x)_{j+1:m})$,

where j is as defined above, and we use that $j \leq m$. By Lemma 4, this is just a specific way to implement function $\hat{F}^m$, so the change is purely conceptual and we have

$$\Pr[X_1] = \Pr[X_0]$$

Game 2. This game is identical to Game 1, except that we replace the function $\hat{F}^m$ implemented by oracle $\mathcal{O}_1$ *partially* with a random function. More precisely, oracle $\mathcal{O}_2$ chooses a second random function $R_j : \{0,1\}^j \to K$. If $b = 1$, then it computes $z = \mathcal{O}_2(x)$ as

1. $k_x := R_j(h(x)_{1:j})$
2. $z := \hat{F}^{m-j}((s_{j+1}, ..., s_m), k_x, h(x)_{j+1:m})$

If $b = 0$, then it proceeds exactly like $\mathcal{O}_1$. The proof of the following lemma is postponed to Section 4.5.

Lemma 6. *From each $\mathcal{A}$ that runs in time $t_\mathcal{A}$ and issues Q oracle queries one can construct an adversary $\mathcal{B}$ that $(t_\mathcal{B}, \epsilon_\mathcal{B}, Q)$-breaks the pseudorandomness of $\hat{F}^j$ where*

$$t_\mathcal{B} = \Theta(t_\mathcal{A}) \quad \textit{and} \quad \epsilon_\mathcal{B} = |\Pr[X_1] - \Pr[X_2]| \tag{6}$$

Game 3. This game is identical to Game 2, but $\mathcal{O}_3$ performs an additional check. Whenever $\mathcal{A}$ makes an oracle query x, $\mathcal{O}_3$ checks whether there has been a previous oracle query x' such that

$$h(x)_{1:j} = h(x')_{1:j}$$

If this holds, then $\mathcal{O}_3$ raises event coll, and the experiment outputs a random bit and terminates. Note that the check is always performed, for both values $b \in \{0,1\}$. Since both games are identical until coll, we have

$$|\Pr[X_2] - \Pr[X_3]| \le \Pr[\mathsf{coll}]$$

Again, the proof of the following lemma is postponed, to Section 4.6.

Lemma 7. *If F is perfectly one-time secure, then* $\Pr[\mathsf{coll}] \le \epsilon_\mathcal{A}/2$ *and* $\Pr\left[X_3 \mid \overline{\mathsf{coll}}\right] = 1/2$.

We finish the proof of Theorem 4 before we prove Lemmas 6 and 7. We have

$$\Pr[X_3] = \Pr[X_3 \mid \mathsf{coll}] \cdot \Pr[\mathsf{coll}] + \Pr\left[X_3 \mid \overline{\mathsf{coll}}\right] \cdot (1 - \Pr[\mathsf{coll}]) \tag{7}$$

Recall that X_3 denotes the probability that the experiment outputs 1, which happens if and only if $\mathcal{A}$ outputs b' with $b = b'$. By construction of the experiment, we abort and output a random bit in Game 3, if coll occurs. In combination with Lemma 7 we thus get

$$\Pr[X_3 \mid \mathsf{coll}] = \Pr\left[X_3 \mid \overline{\mathsf{coll}}\right] = 1/2$$

Plugging this into (7) yields

$$\Pr[X_3] = 1/2 \cdot \Pr[\mathsf{coll}] + 1/2 \cdot (1 - \Pr[\mathsf{coll}]) = 1/2 \tag{8}$$

Lower bound on $\epsilon_\mathcal{B}$. Finally, using (8), the bounds from Lemmas 6 and 7, and the fact that $\Pr[X_0] = \Pr[X_1]$, we obtain a lower bound on $\epsilon_\mathcal{B}$:

$$\begin{aligned} 1/2 + \epsilon_\mathcal{A} = \Pr[X_0] = \Pr[X_1] &\le \Pr[X_2] + \epsilon_\mathcal{B} \le 1/2 + \epsilon_\mathcal{A}/2 + \epsilon_\mathcal{B} \\ \iff \epsilon_\mathcal{B} &\ge \epsilon_\mathcal{A}/2 \end{aligned}$$

Furthermore, by Lemma 6, algorithm $\mathcal{B}$ runs in time $t_\mathcal{B} = \Theta(t_\mathcal{A})$ and issues Q oracle queries. □

4.5 Proof of Lemma 6

Adversary $\mathcal{B}$ plays the pseudorandomness security experiment with function $\hat{F}^j$. Let $\mathcal{O}$ denote the PRF oracle provided to $\mathcal{B}$ in this game. $\mathcal{B}$ runs $\mathcal{A}$ as a subroutine by simulating the security experiment as follows.

Initialization. $\mathcal{B}$ samples a bit $b \overset{\$}{\leftarrow} \{0,1\}$, a hash function $h \leftarrow \mathcal{H}_{n,m}$, and picks $(s_{j+1}, ..., s_m)$, where $s_i \leftarrow S$ for all $i \in [\![j+1, m]\!]$

Handling of oracle queries. Whenever $\mathcal{A}$ queries $x \in \{0,1\}^n$, $\mathcal{B}$ proceeds as follows.

– If $b = 0$, then $\mathcal{B}$ proceeds exactly like the original experiment. That is, it responds with $R(x)$, where $R : \{0,1\}^n \to K$ is a random function.
– If $b = 1$, then $\mathcal{B}$ computes $h(x)$ and queries $\mathcal{O}$ to obtain $k_x := \mathcal{O}(h(x)_{1:j})$. Then it computes
$$z := \hat{F}^{m-j}((s_{j+1}, ..., s_m), k_x, h(x)_{j+1:m})$$
and returns z to $\mathcal{A}$.

Finalization. Finally, when $\mathcal{A}$ terminates, then $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs, and terminates.

Analysis of $\mathcal{B}$. Note that the running time of $\mathcal{B}$ is essentially identical to the running time of $\mathcal{A}$ plus a minor number of additional operations, thus we have $t_{\mathcal{B}} = \Theta(t_{\mathcal{A}})$. If $\mathcal{O}(x) = \hat{F}^j((s_1, ..., s_j, k), h(x)_{1:j})$, then by Lemma 4 it holds that $z = \hat{F}^m((s_1, ..., s_m, k), h(x))$. Thus, the view of $\mathcal{A}$ is identical to Game 1. If $\mathcal{O}(x)$ implements a random function, then its view is identical to Game 2. This yields the claim.

4.6 Proof of Lemma 7

In order to show that $\Pr[\mathsf{coll}] \leq \epsilon_{\mathcal{A}}/2$, we prove that all queries of $\mathcal{A}$ are independent of h, regardless of $b = 0$ or $b = 1$, until coll occurs. This allows us to derive an upper bound on coll. Consider the sequence of queries $x_1, \ldots, x_Q$ made by $\mathcal{A}$. Recall that we assume $x_u \neq x_v$ for $u \neq v$ without loss of generality.

The case $b = 0$. In this case, $\mathcal{O}_3(x_i)$ is a random function $R(x_i)$, and therefore all information observed by $\mathcal{A}$ is independent of h, until coll occurs. Thus, the view of $\mathcal{A}$ is equivalent to a world in which the experiment does not choose h at the beginning, but only after $\mathcal{A}$ has made all queries, and only then computes $h(x_i)_{1:j}$ for all $i \in [\![Q]\!]$ and outputs a random bit if a collision occurred. By the almost-universality, we thus obtain that
$$\Pr[\mathsf{coll} \mid b = 0] \leq \sum_{i=2}^{Q} \frac{i-1}{2^{j-1}} \leq \frac{Q^2}{2^j} \leq \frac{Q^2 \epsilon_{\mathcal{A}}}{2Q^2} = \frac{\epsilon_{\mathcal{A}}}{2}.$$

Note that we use here that $j \geq \log(2Q^2/\epsilon_{\mathcal{A}})$, which holds due to the definition of j in (5).

The case $b = 1$. We may assume without loss of generality that $Q > 0$, as otherwise $\mathcal{A}$ receives no information about b and thus we would have $\epsilon_{\mathcal{A}} = 0$. Consider the first query $\mathcal{O}_3(x_1)$ of $\mathcal{A}$. The oracle proceeds as follows. At first it computes $k_{x_1} := R_j(h(x_1)_{1:j})$. Since R_j is a random function, this value is independent of h. In the next step it computes $z_1 := \hat{F}^{m-j}((s_{j+1}, ..., s_m), k_{x_1}, h(x_1)_{j+1:m})$, which is still uniformly random. To see this, note that the perfect one-time security of F guarantees perfect one-time security of $\hat{F}^{m-j}$ as shown in Lemma 5. Thus $\mathcal{A}$ gains no information about h at this point and the next query cannot be adaptive with regard to h.

Now if $\mathcal{A}$ queries $\mathcal{O}_3(x_2)$, then the experiment will evaluate the random functions R_j on a different position than in the first query, unless

$$h(x_1)_{1:j} = h(x_2)_{1:j} \tag{9}$$

Due to the fact that the response to x_1 was independent of h and the almost-universality of h, (9) happens with probability at most $1/2^{j-1}$. Therefore, again by the perfect one-time security of F, $\mathcal{A}$ receives another uniformly random value z_2, which is independent of h, except with probability at most $1/2^{j-1}$. Continuing this argument inductively over all Q queries of $\mathcal{A}$, we see that on its i-th query $\mathcal{A}$ will receive a random response which is independent of h, except with probability $(i-1)/2^{j-1}$, provided that all previous responses were independent of h. A union bound now yields

$$\Pr\left[\mathsf{coll} \mid b = 1\right] \leq \sum_{i=2}^{Q} \frac{i-1}{2^{j-1}} \leq \frac{Q^2}{2^j} \leq \frac{Q^2 \epsilon_{\mathcal{A}}}{2Q^2} = \frac{\epsilon_{\mathcal{A}}}{2}.$$

It remains to show that $\Pr\left[X_3 \mid \overline{\mathsf{coll}}\right] = 1/2$. Let us consider the case $b = 1$. If $\overline{\mathsf{coll}}$ occurs, then there are no collisions, such that the oracle calls random function R_j on always different inputs, each time receiving an independent, uniformly random value. Applying the perfect one-time security of $\hat{F}^{m-j}$ again, the response of the oracle to each query is therefore uniformly distributed and independent of all other queries. Thus, provided that no collision occurs, the view in case $b = 1$ is perfectly indistinguishable from the case $b = 0$, which yields the claim.

4.7 On the necessity of the *"all-prefix"* property

One may ask at this point whether the *"all-prefix"* property is really necessary, or whether it is possible to use a standard universal hash function with fixed output space $\{0, 1\}^j$ instead.

Let us explain why the *"all-prefix"* property is not only sufficient, but also necessary. Recall that j depends on the particular values of Q and $\epsilon_{\mathcal{A}}$ of a particular given adversary, via the definition $j = \lceil \log(2Q^2/\epsilon_{\mathcal{A}}) \rceil$ in (5). One may wonder why we set j so precisely, depending on the given adversary, rather than simply choosing j sufficiently large such that it would work for *any* efficient adversary.

The purpose of this precise choice is because we have to find the right balance between two properties that we need to obtain *tight* security:

1. On the one hand, we need j to be sufficiently large, such that the probability of a collision of (the j-bit prefix of) the universal hash function is sufficiently unlikely.

2. On the other hand, we have to keep j short enough, in order to get a tight reduction.

This is why we make the value j dependent on the given adversary, specifically on the particular values of Q and $\epsilon_{\mathcal{A}}$.

We stress that we do this *only in the security proof*, but not in the PRF *construction* itself. That is, we do not simply fix j to be the largest value of j such that the collision probability is sufficiently small for any adversary, because then for certain adversaries j could be "too large" such that the reduction would not be tight. Similarly, if we used a standard universal hash function with output length j, then this would also fix j to some specific value in the *construction* of the PRF, and thus would again make the PRF construction only tightly secure for certain adversaries that match this particular choice of j, but not necessarily for all efficient adversaries.

For example using a standard UHF with $m = \omega(\log \lambda)$ is sufficient to bound the collision probaility, but this yields only super-logarithmic tightness, and thus would be worse than in the construction of Döttling and Schröder [DS15], while with an APUHF we achieve logarithmic tightness.

Hence, the important new feature that *all-prefix* universality provides is that it guarantees that a suitable choice of j exists for *any* efficient adversary. This makes the construction independent of a particular class of adversaries that match a certain fixed value of j, while at the same time it ensures that the security proof depends *tightly* on the particularly given adversary. Hence, using an APUHF instead of a standard universal hash function is not just sufficient, but also necessary in order to capture all efficient adversaries and to keep the security proof tight.

We note that Döttling and Schröder [DS15] also use multiple instances of the underlying pseudorandom function, with increasing security, in order to achieve tightness. Essentially, we replace these multiple instances with a single instance, in combination with an all-prefix universal hash function. From an abstract high-level perspective, in our approach each prefix implicitly corresponds to one PRF instance of [DS15]. This makes our construction significantly more efficient.

5 Applications

5.1 Efficient and Tightly-Secure PRF from Matrix Diffie-Hellman Assumptions

We recall the definition of the matrix Diffie-Hellman (MDDH) assumption and the pseudorandom function (PRF) from [EHK$^+$17]. We consider a variant where an all-prefix almost-universal hash function is applied to the input before it is processed by the PRF. We note that the MDDH assumption generalizes the Decisional Diffie-Hellman (DDH) and Decisional d-Linear (d-LIN) assumptions, and, moreover, it gives us a framework to analyze the algebraic structure behind the Diffie-Hellman-based cryptographic primitives. Thus, our results can be carried on to the Naor-Reingold PRF (based on the DDH assumption) [NR97] and the Lewko-Waters PRF (based on the d-LIN assumption) [LW09].

Notations and the MDDH Assumption. Let $\mathcal{G} := (\mathbb{G}, P, q)$ be a description of an additive group $\mathbb{G}$ with random generator P and prime order q. Following the "implicit notation" of [EHK+13], we write $[a]$ shorthand for aP. More generally, for a matrix $\mathbf{A} = (a_{ij}) \in \mathbb{Z}_q^{n\times m}$, we define $[\mathbf{A}]_s$ as the implicit representation of $\mathbf{A}$ in $\mathbb{G}$:

$$[\mathbf{A}] := \begin{pmatrix} a_{11}P \ \ldots \ a_{1m}P \\ \\ a_{n1}P \ \ldots \ a_{nm}P \end{pmatrix} \in \mathbb{G}^{n\times m}$$

Let us first recall the definition of the matrix Diffie-Hellman (MDDH) problem [EHK+13,EHK+17].

Definition 10 **(Matrix distribution).** *Let $\ell, d \in \mathbb{N}$ and $\ell > d$. We call $\mathcal{D}_{\ell,d}$ a matrix distribution if it outputs matrices in $\mathbb{Z}_q^{\ell\times d}$ of full rank d in polynomial time, namely, it is efficiently samplable. We define $\mathcal{D}_d := \mathcal{D}_{d+1,d}$.*

Without loss of generality, we assume the first d rows of $\mathbf{A} \xleftarrow{\$} \mathcal{D}_{\ell,d}$ form a full-rank and invertible matrix, and we denote it by $\overline{\mathbf{A}}$ and the rest $\ell - d$ rows by $\underline{\mathbf{A}}$.

Definition 11 **(Transformation matrix).** *Let $\mathcal{D}_{\ell,d}$ be a matrix distribution and $\mathbf{A}$ be a matrix from it. The* transformation matrix *of $\mathbf{A}$ is defined as $\mathbf{T} := \underline{\mathbf{A}} \cdot \overline{\mathbf{A}}^{-1} \in \mathbb{Z}_q^{(\ell-d)\times d}$.*

The $\mathcal{D}_{\ell,d}$-MDDH problem is to distinguish the two distributions $([\mathbf{A}], [\mathbf{Aw}])$ and $([\mathbf{A}], [\mathbf{u}])$ where $\mathbf{A} \xleftarrow{\$} \mathcal{D}_{\ell,d}$, $\mathbf{w} \xleftarrow{\$} \mathbb{Z}_q^d$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^\ell$.

Definition 12 **($\mathcal{D}_{\ell,d}$-Matrix Diffie-Hellman assumption, $\mathcal{D}_{\ell,d}$-MDDH).** *Let $\mathcal{D}_{\ell,d}$ be a matrix distribution. We say that adversary $\mathcal{A}$ $(t_\mathcal{A}, \epsilon_\mathcal{A})$-breaks the $\mathcal{D}_{\ell,d}$-Matrix Diffie-Hellman ($\mathcal{D}_{\ell,d}$-MDDH) assumption in group $\mathbb{G}$, if $\mathcal{A}$ runs in time $t_\mathcal{A}$ and*

$$|\Pr[\mathcal{A}(\mathcal{G}, [\mathbf{A}], [\mathbf{Aw}]) = 1] - \Pr[\mathcal{A}(\mathcal{G}, [\mathbf{A}], [\mathbf{u}]) = 1]| \geq \epsilon_\mathcal{A},$$

where the probability is taken over $\mathbf{A} \xleftarrow{\$} \mathcal{D}_{\ell,d}, \mathbf{w} \xleftarrow{\$} \mathbb{Z}_q^d, \mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^\ell$.

Examples of $\mathcal{D}_{\ell,d}$-MDDH. [EHK+13,EHK+17] define distributions $\mathcal{L}_d$, $\mathcal{C}_d$, $\mathcal{SC}_d$, $\mathcal{IL}_d$, and $\mathcal{U}_d$ which corresponds to the d-Linear, d-Cascade, d-Symmetric-Cascade, d-Incremental-Linear, and d-Uniform assumption, respectively. All these assumptions are proven secure in the generic group model [EHK+13,EHK+17] and form a hierarchy of increasingly weaker assumptions.

A simple example is the $\mathcal{L}_1$-MDDH assumption for $d = 1$, which is the DDH assumption: Choose $a, w, z \xleftarrow{\$} \mathbb{Z}_q$, and the DDH assumption states that the following two distributions are computationally indistinguishable:

$$([1, a, w, aw]) \approx_c ([1, a, w, z]).$$

This can be represented via the $\mathcal{L}_1$-MDDH assumption which states the following two distributions are computationally indistinguishable:

$$([\begin{smallmatrix} a \\ 1 \end{smallmatrix}], [\begin{smallmatrix} aw \\ w \end{smallmatrix}]) =: ([\mathbf{A}], [\mathbf{Aw}]) \approx_c ([\mathbf{A}], [\mathbf{u}]) := ([\begin{smallmatrix} a \\ 1 \end{smallmatrix}], [\begin{smallmatrix} z \\ w \end{smallmatrix}]).$$

For $d = 1$ the transformation matrix $\mathbf{T}$ contains only one element, and for $\mathcal{L}_1$-MDDH the corresponding transformation matrix is $\mathbf{T} = \frac{1}{a}$.

We give more examples of matrix distributions from [EHK+13,EHK+17] for $d = 2$ in Appendix A.

The PRF construction of [EHK$^+$17] and its security. Let $\mathcal{G} := (\mathbb{G}, P, q)$ be a description of an additive group $\mathbb{G}$ with random generator P and prime order q. Let $\mathcal{D}_{\ell,d}$ be a matrix distribution and we assume that $(\ell - d)$ divides d and define $t := d/(\ell - d)$.

Following the approach of Section 5.3 of [EHK$^+$17], we choose a random vector $\mathbf{h} \xleftarrow{\$} \mathbb{Z}_q^d$, and, for $i = 1, ..., m$ and $j = 1, ..., t$, we choose $\mathbf{A}_{i,j} \xleftarrow{\$} \mathcal{D}_{\ell,d}$ and compute transformation matrices $\hat{\mathbf{T}}_{i,j} := \underline{\mathbf{A}}_{i,j}\overline{\mathbf{A}}_{i,j}^{-1} \in \mathbb{Z}_q^{(\ell-d)\times d}$ and define the aggregated transformation matrices

$$\mathbf{T}_i := \begin{pmatrix} \hat{\mathbf{T}}_{i,1} \\ \vdots \\ \hat{\mathbf{T}}_{i,t} \end{pmatrix} \in \mathbb{Z}_q^{d\times d},$$

and $\mathbf{S} := (\mathbf{T}_1, ..., \mathbf{T}_m)$. Here, for $i \in \{1, ..., m\}$, we require that $\mathbf{T}_i$ has full rank. We note that this requirement can be satisfied by all the matrix distributions described in [EHK$^+$17] with overwhelming probability. This implies the distribution of our $\mathbf{T}_i$'s is statistically close to that in [EHK$^+$17], up to a negligibly small statistical distance of $1/(q - 1)$. Thus, their security results can be applied here.

Now let $S := \mathbb{Z}_q^{d\times d}$, $K := \mathbb{G}^d$, and $X := \{0, 1\}$. The basis of the PRF construction from [EHK$^+$17] is the function $F_{\text{MDDH}} : S \times K \times X \to K$ defined as

$$F_{\text{MDDH}}(\mathbf{T}, [\mathbf{h}], x) := \begin{cases} [\mathbf{h}] & \text{if } x = 0 \\ [\mathbf{T} \cdot \mathbf{h}] & \text{if } x = 1 \end{cases} \tag{10}$$

By applying the augmented cascade of Boneh *et al.* [BMR10] (Figure 1) to F_{MDDH}, Escala *et al.* [EHK$^+$17] obtain their PRF F^m_{MDDH} with key space $(\mathbb{Z}_q^{(d\times d)})^m \times \mathbb{G}^d$ and domain $\{0, 1\}^m$:

$$F^m_{\text{MDDH}} : (\mathbb{Z}_q^{(d\times d)})^m \times \mathbb{G}^d \times \{0, 1\}^m \to \mathbb{G}$$
$$F^m_{\text{MDDH}}(\mathbf{S}, [\mathbf{h}], x) := \left[\left(\prod_{i:x_i=1} \mathbf{T}_i \right) \cdot \mathbf{h} \right] \tag{11}$$

where $\mathbf{S} := (\mathbf{T}_1, ..., \mathbf{T}_m)$. The following theorem was proven in [EHK$^+$13,EHK$^+$17].

Theorem 5 ([EHK$^+$17, Theorem 12]). *From each adversary $\mathcal{A}$ that $(t_\mathcal{A}, \epsilon_\mathcal{A}, Q)$-breaks the security of F^m_{MDDH} with input space $\{0, 1\}^m$ we can construct an adversary $\mathcal{B}$ that $(t_\mathcal{B}, \epsilon_\mathcal{B})$-breaks the $\mathcal{D}_{\ell,d}$-MDDH assumption in $\mathbb{G}$ with*

$$t_\mathcal{B} = \Theta(t_\mathcal{A}) \qquad \textit{and} \qquad \epsilon_\mathcal{B} \geq \frac{\epsilon_\mathcal{A}}{dm}$$

Note that d is a constant, so that the security loss is linear in the size m of the input space.

Our construction. By additionally encoding the input with an APUHF as described in (4), we finally obtain the function $F^{\mathcal{H}_{n,m}}_{\text{MDDH}} : S^m \times K \times \mathcal{H}_{n,m} \times \{0, 1\}^n \to K$ as

$$F^{\mathcal{H}_{n,m}}_{\text{MDDH}}(\mathbf{S}, [\mathbf{h}], h, x) = F^m_{\text{MDDH}}(\mathbf{S}, [\mathbf{h}], h(x)) = \left[\left(\prod_{i:h(x)_i=1}^{m} \mathbf{T}_i \right) \cdot \mathbf{h} \right] \tag{12}$$

In order to apply Theorem 4 to show that this particular instance of the augmented cascade with encoded input is a secure PRF with key space $S^m \times K \times \mathcal{H}_{n,m}$ and domain $\{0,1\}^n$, we merely have to prove that function F_{MDDH} is perfectly one-time secure.

Lemma 8. *Function F_{MDDH} from (10) is perfectly one-time secure.*

Proof. We have to show that

$$\Pr_{[\mathbf{h}] \xleftarrow{\$} \mathbb{G}^d} [F_{\text{MDDH}}(\mathbf{T}, [\mathbf{h}], x) = [\mathbf{h}']] = \frac{1}{|\mathbb{G}|^d}.$$

for all $(\mathbf{T}, x, [\mathbf{h}']) \in S \times \{0,1\} \times \mathbb{G}^d$.

If $x = 0$ then $F_{\text{MDDH}}(\mathbf{T}, [\mathbf{h}], 0) = [\mathbf{h}]$, which is a random vector in $\mathbb{G}^d$ by definition. If $x = 1$ then $F_{\text{MDDH}}(\mathbf{T}, [\mathbf{h}], 1) = [\mathbf{Th}]$, which is again a random vector, due to the fact that $\mathbf{T}$ is a full-rank matrix. □

By combining Theorem 4 with Theorem 5 we now obtain the following result, which shows that setting $m = \omega(\log \lambda)$ is sufficient to achieve tight security.

Theorem 6. *Let $m = \omega(\log \lambda)$ be (slightly) super-logarithmic and $\mathcal{H}_{n,m}$ be a family of all-prefix almost universal hash functions. From each adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, \epsilon_{\mathcal{A}}, Q)$-breaks the security of $F_{MDDH}^{\mathcal{H}_{n,m}}$ with $Q/\epsilon_{\mathcal{A}} = \mathsf{poly}(\lambda)$ for some polynomial poly we can construct an adversary $\mathcal{B}$' that $(t'_{\mathcal{B}}, \epsilon'_{\mathcal{B}})$-breaks the $\mathcal{D}_{\ell,d}$-MDDH assumption in $\mathbb{G}$ with*

$$t'_{\mathcal{B}} = \Theta(t_{\mathcal{A}}) \qquad \textit{and} \qquad \epsilon'_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{A}}}{2dj}$$

where $j = O(\log \lambda)$.

Proof. Theorem 4 shows that from each adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, \epsilon_{\mathcal{A}}, q)$-breaks the pseudorandomness of $F_{\text{MDDH}}^{\mathcal{H}_{n,m}}$ with $Q/\epsilon_{\mathcal{A}} = \mathsf{poly}(\lambda)$ for some polynomial poly, we can construct an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}}, Q)$-breaks the pseudorandomness of the function F_{MDDH}^{j} with input space $\{0,1\}^j$, where

$$j = O(\log \lambda) \qquad \text{and} \qquad t_{\mathcal{B}} = \Theta(t_{\mathcal{A}}) \qquad \text{and} \qquad \epsilon_{\mathcal{B}} \geq \epsilon_{\mathcal{A}}/2$$

Theorem 5 in turn shows that from each adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}}, Q)$-breaks the security of F_{MDDH}^{j} we can construct an adversary $\mathcal{B}$' that $(t'_{\mathcal{B}}, \epsilon'_{\mathcal{B}})$-breaks the $\mathcal{D}_{\ell,d}$-MDDH assumption in $\mathbb{G}$ with

$$t'_{\mathcal{B}} = \Theta(t_{\mathcal{B}}) \qquad \text{and} \qquad \epsilon'_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{B}}}{dj} \geq \frac{\epsilon_{\mathcal{A}}}{2dj}$$

which yields the claim. □

Comparison to the DDH-based PRF of [NR97]. One particularly interesting instantiation of F^m_{MDDH} is based on the $\mathcal{L}_1$-MDDH assumption, which is an improvement over the famous Naor-Reingold construction based on the DDH (namely, $\mathcal{L}_1$-MDDH) assumption from [NR97]. In F^m_{MDDH}, we sample $\mathbf{A}_i$ from $\mathcal{D}_{\ell,d}$ and then compute the aggregated transformation matrices $\mathbf{T}_i$. For the $\mathcal{L}_1$ distribution, we can equivalently pick random elements T_i from $\mathbb{Z}_q$.

Let $\mathbb{G}$ be a group of prime order q, $S := \mathbb{Z}_q$, $K := \mathbb{G}$, $X := \{0,1\}^n$ and $m = \omega(\log \lambda)$ as above. Then we choose $T_1, ..., T_m, a \overset{\$}{\leftarrow} \mathbb{Z}_q$ and obtain a PRF with domain $\{0,1\}^n$ as

$$F^{\mathcal{H}_{n,m}}_{\mathsf{DDH}}(\mathbf{S}, [a], h, x) = \left[\left(\prod_{i:h(x)_i=1}^{m} T_i \right) \cdot a \right].$$

Note that the resulting PRF is identical to the original Naor-Reingold function [NR97], except that an APUHF h is applied to the input x before it is processed in the Naor-Reingold construction. For the original construction from [NR97] both the size of the secret key and the tightness loss of the security proof (based on the DDH assumption in $\mathbb{G}$) are *linear* in the bit-length of the function input. We show that merely by encoding the input with an APUHF one can obtain shorter secret keys of size $m = \omega(\log \lambda)$ and with security loss $O(\log \lambda)$ (based on the same assumption as [NR97]), even for input size $n \gg m$.

Comparison to the Matrix-DDH PRF of [DS15]. Döttling and Schröder [DS15] also described a variant of the Matrix-DDH-based PRF of [EHK+13]. Their PRF is the function

$$F^{\mathsf{DS15}}_{\mathsf{MDDH}}(\mathbf{S}, [\mathbf{h}], x) := \left[\left(\prod_{j=1}^{m} (\mathbf{T}_i + x^{2^j} \cdot \mathbf{I}) \right) \cdot \mathbf{h} \right] \qquad (13)$$

where $\mathbf{S}$, $[\mathbf{h}]$, and m are as in our construction, and $x \in \mathbb{Z}_q$. Thus, in comparison, our construction from (12) uses the same value of m, but is somewhat simpler that (13) and also slightly more efficient to evaluate. In particular, the computation of the terms of the form $(x^{2^j} \cdot \mathbf{I})$ is replaced with a single evaluation of the APUHF h. Another difference is that the domain of their function is restricted to $x \in \mathbb{Z}_q$, while in our case $x \in \{0,1\}^n$ can be any bit string of polynomially-bounded length $n = n(\lambda)$.

5.2 More Efficient LWE-based PRFs

We recall the learning with error (LWE) assumption. Then we apply our results to the LWE-based PRF from Banerjee, Peikert and Rosen [BPR12].

Definition 13 (Learning With Errors assumption, LWE). *Let p be a modulus, N be a positive integer, and $\chi_\alpha := D_{\mathbb{Z}_p,\alpha}$ be a Gaussian distribution with noise parameter*

α. Let $\mathbf{h} \overset{\$}{\leftarrow} \mathbb{Z}_p^N$ be a random vector. We say that adversary $\mathcal{A}$ $(t_\mathcal{A}, \epsilon_\mathcal{A})$-breaks the LWE$_{p,N,\alpha}$ assumption if it runs in time $t_\mathcal{A}$ and

$$|\Pr[\mathcal{A}(\mathbf{h}, \mathbf{h}^\top \mathbf{s} + e) = 1] - \Pr[\mathcal{A}(\mathbf{h}, u) = 1]| \geq \epsilon_\mathcal{A},$$

where $\mathbf{h} \overset{\$}{\leftarrow} \mathbb{Z}_p^N$, $\mathbf{s} \overset{\$}{\leftarrow} \mathbb{Z}_p^N$, $e \overset{\$}{\leftarrow} \chi_\alpha$ and $u \overset{\$}{\leftarrow} \mathbb{Z}_p$.

Let $\lfloor \cdot \rceil$ be the rounding function, which rounds a real number to the largest integer which does not exceed it. Let $p \geq q$. For an element $h \in \mathbb{Z}_p$, we define the rounding function $\lfloor \cdot \rceil_q : \mathbb{Z}_p \to \mathbb{Z}_q$ as $\lfloor h \rceil_q := \lfloor (q/p) h \rceil$, and for a vector $\mathbf{h} \in \mathbb{Z}_p^N$, the rounding function $\lfloor \mathbf{h} \rceil_q$ is defined component-wise.

The PRF construction of [BPR12] and its security. Let $S := \chi_\alpha^{N \times N}$ and $K := \mathbb{Z}_p^N$, and $X := \{0,1\}$. We assume that $\mathbf{S} \in S$ has full rank. The basis of the PRF of [BPR12] is the function $F_{\text{LWE}} : S \times K \times X \to K$,

$$F_{\text{LWE}}(\mathbf{S}, \mathbf{h}, x) := \begin{cases} \mathbf{h} & \text{if } x = 0 \\ \mathbf{S} \cdot \mathbf{h} & \text{if } x = 1 \end{cases} \tag{14}$$

We apply a slightly different augmented cascade transformation in Figure 1 to obtain the PRF of [BPR12] with key space $(\chi_\alpha^{(N \times N)})^m \times \mathbb{Z}_p^N$ and domain $\{0,1\}^m$:

$$\begin{aligned} F_{\text{LWE}}^m : (\chi_\alpha^{(N \times N)})^m \times \mathbb{Z}_p^N \times \{0,1\}^m &\to \mathbb{Z}_q \\ F_{\text{LWE}}^m(\mathbf{S}, \mathbf{h}, x) &:= \left\lfloor \left(\prod_{i : x_i = 1}^m \mathbf{S}_i \right) \cdot \mathbf{h} \right\rceil_q \end{aligned} \tag{15}$$

where $\mathbf{S} := (\mathbf{S}_1, ..., \mathbf{S}_m)$ and $\mathbf{h} \overset{\$}{\leftarrow} \mathbb{Z}_p^N$. Different to Figure 1, we apply the rounding function on the output of Figure 1.

Theorem 7 ([BPR12, Theorem 5.2]). *Let $\chi_\alpha = D_{\mathbb{Z},\alpha}$ be a Gaussian distribution with parameter $\alpha > 0$, let m be a positive integer that denotes the length of message inputs. Define $B := m(C\alpha\sqrt{N})^m$ for a suitable universal constant C. Let p, q be two moduli such that $p > q \cdot B \cdot N^{\omega(1)}$.*

From each adversary $\mathcal{A}$ that $(t_\mathcal{A}, \epsilon_\mathcal{A}, Q)$-breaks the security of F_{LWE}^m with input space $\{0,1\}^m$ (for an arbitrary positive integer m) we can construct an adversary $\mathcal{B}$ that $(t_\mathcal{B}, \epsilon_\mathcal{B})$-breaks the LWE$_{p,N,\alpha}$ assumption with

$$t_\mathcal{B} = \Theta(t_\mathcal{A}) \qquad \textit{and} \qquad \epsilon_\mathcal{B} \geq \frac{\epsilon_\mathcal{A}}{m \cdot N}$$

Note that B is an important parameter, since it determines the size of the LWE modulus p and contains the expensive term $N^{m/2}$, which is exponential in m. Thus, a smaller m can give us a smaller p, which in turn yields a weaker LWE assumption and a much more efficient PRF. In the following, we apply our results to F_{LWE}^m to reduce m from polynomial to logarithmic in security parameter λ.

Our construction. By additionally encoding the input with an APUHF as described in (4), we finally obtain $F_{\text{LWE}}^{\mathcal{H}_{n,m}} : (\chi_\alpha^{(N \times N)})^m \times \mathbb{Z}_p^N \times \mathcal{H}_{n,m} \times \{0,1\}^m \to \mathbb{Z}_q^N$ as

$$F_{\text{LWE}}^m(\mathbf{S}, \mathbf{h}, h(x)) := \left\lfloor \left(\prod_{i:h(x)_i=1}^{m} \mathbf{S}_i \right) \cdot \mathbf{h} \right\rfloor_q \tag{16}$$

In order to apply Theorem 4 to show that this particular instance of the augmented cascade with encoded input is a secure PRF with key space $S^m \times K \times \mathcal{H}_{n,m}$ and domain $\{0,1\}^n$, we have to prove that function F_{LWE} is perfectly one-time secure.

Lemma 9. *Function F_{LWE} from (14) is perfectly one-time secure.*

Proof. We have to show that

$$\Pr_{\mathbf{h} \xleftarrow{\$} \mathbb{Z}_p} [F_{\text{LWE}}(\mathbf{S}, \mathbf{h}, x) = \mathbf{h}'] = \frac{1}{p^N}.$$

for all $(\mathbf{S}, x, \mathbf{h}') \in S \times \{0,1\} \times \mathbb{Z}_p^N$.

If $x = 0$ then $F_{\text{LWE}}(\mathbf{S}, \mathbf{h}, 0) = \mathbf{h}$, which is a random vector in $\mathbb{Z}_p^N$ by definition. If $x = 1$ then $F_{\text{LWE}}(\mathbf{S}, \mathbf{h}, 1) = \mathbf{S} \cdot \mathbf{h}$, which is again a random vector, due to the fact that $\mathbf{S}$ is a full-rank matrix. $\square$

We recall the following useful notations and corollary for the proof of Theorem 8 given below. We define an error sampling function $E : \{0,1\}^j \to \mathbb{Z}^N$ and for $x \in \{0,1\}^j$ and $j \in [\![m]\!]$ we define the randomized version of F_{LWE}^j as $\tilde{F}_{\text{LWE}}^j(x) = \left(\prod_{i:x_i=1}^{j} \mathbf{S}_i \right) \cdot \mathbf{h} + E(x)$. The proof of Theorem 5.2 and Lemma 5.5 in [BPR12] show that $\tilde{F}_{\text{LWE}}^j$ is pseudorandom based on the decisional LWE assumption and it holds that $F_{\text{LWE}}^m(x) = \left\lfloor \left(\prod_{i>j \wedge x_i=1}^{m} \mathbf{S}_i \right) \cdot \tilde{F}_{\text{LWE}}^j(x) \right\rfloor_q$, except with negligible probability. We summarize this in the following corollary.

Corollary 1. *Let all the parameters be defined as in Theorem 7. There exists an efficiently randomized error sampling function $E : \{0,1\}^j \to \mathbb{Z}^N$, such that, from each adversary $\mathcal{A}$ that $(t_\mathcal{A}, \epsilon_\mathcal{A}, Q)$-breaks the security of $\tilde{F}_{LWE}^j(x) = \left(\prod_{i:x_i=1}^{j} \mathbf{S}_i \right) \cdot \mathbf{h} + E(x)$ with input $x \in \{0,1\}^j$ (for $j \in [\![m]\!]$) we can construct an adversary $\mathcal{B}$ that $(t_\mathcal{B}, \epsilon_\mathcal{B})$-breaks the LWE$_{p,N,\alpha}$ assumption with*

$$t_\mathcal{B} = \Theta(t_\mathcal{A}) \qquad \text{and} \qquad \epsilon_\mathcal{B} \geq \frac{\epsilon_\mathcal{A}}{m \cdot N}.$$

Moreover, except with probability $2^{-\Omega(N)}$, we have

$$F_{LWE}^m(x) = \left\lfloor \left(\prod_{i>j \wedge x_i=1}^{m} \mathbf{S}_i \right) \cdot \tilde{F}_{LWE}^j(x) \right\rfloor_q .$$

Theorem 8. *Let $m = \omega(\log \lambda)$ be (slightly) super-logarithmic and $\mathcal{H}_{n,m}$ be a family of all-prefix almost universal hash functions. Let $\chi_\alpha = D_{\mathbb{Z},\alpha}$ be a Gaussian distribution with parameter $\alpha > 0$, let m be a positive integer denotes the length of message inputs. Define $B := m(C\alpha\sqrt{N})^m$ for a suitable universal constant C. Let p, q be two moduli such that $p > q \cdot B \cdot N^{\omega(1)}$.*

From each adversary $\mathcal{A}$ that $(t_\mathcal{A}, \epsilon_\mathcal{A}, Q)$-breaks the security of $F_{\text{LWE}}^{\mathcal{H}_{n,m}}$ with $Q/\epsilon_\mathcal{A} = \mathsf{poly}(\lambda)$ for some polynomial poly we can construct an adversary $\mathcal{B}$' that $(t'_\mathcal{B}, \epsilon'_\mathcal{B})$-breaks the LWE$_{p,N,\alpha}$ assumption with

$$t'_\mathcal{B} = \Theta(t_\mathcal{A}) \qquad \text{and} \qquad \epsilon'_\mathcal{B} \geq \frac{\epsilon_\mathcal{A}}{2j \cdot N} - 2^{-\Omega(N)}$$

where $j = O(\log \lambda)$.

Proof. The proof is the same as the one for Theorem 4. The only difference is between Games 1 and 2. Here we do one intermediate game transition Game 1': We simulate $\mathcal{O}_1(x)$ by returning $F_{\text{LWE}}^m(x) = \left\lfloor \left(\prod_{i>j \wedge x_i=1}^m \mathbf{S}_i\right) \cdot \tilde{F}_{\text{LWE}}^j(x) \right\rceil_q$ and $\mathcal{O}_0$ by returning a random vector in $\mathbb{Z}_q^N$.

By the second statement of Corollary 1, the difference between Games 1 and 1' is bounded by the statistical difference $2^{-\Omega(N)}$. Moreover, the difference between Games 1' and 2 is bounded by the security of $\tilde{F}_{\text{LWE}}^j$. By the first statement of Corollary 1 we can conclude the proof. □

Comparison to the LWE PRF of [DS15]. Döttling and Schröder [DS15] describe a different variant of the BPR PRF. Their approach is to instantiate their Construction 1 with the BPR PRF and then obtain the following function

$$F_{\text{LWE}}^{\mathsf{DS15}}(K, \mathbf{h}, x) = \bigoplus_{i=1}^{L} \bigoplus_{j=1}^{\lambda} F_{\text{LWE}}^{2^i}(\mathbf{S}, \mathbf{h}, \mathsf{Bin}(j) || H_{2^i,j}(x))$$

where $L = \omega(\log \lambda)$, for each $j \in [\![\lambda]\!]$ the function $H_{2^i,j} : \{0,1\}^n \to \{0,1\}^{i+1}$ is chosen from a suitable universal hash function family with range $\{0,1\}^{i+1}$, and $\mathbf{S}$ is chosen the same as ours.

Compared with $F_{\text{LWE}}^{\mathsf{DS15}}$, our variant has shorter secret keys: instead of having $L \cdot \lambda$ many hash functions, we only have a single one. In terms of computation efficiency, instead of running H_i and F_{LWE}^i for $L \cdot \lambda$ times, we only run the hash function and F_{LWE}^m once.

6 Conclusion

We have introduced all-prefix (almost-)universal hash functions (APUHFs) as a tool to generically improve the augmented cascade construction of pseudorandom functions by Boneh, Montgomery, and Raghunathan [BMR10]. By generically applying an APUHF to the function input before processing it in the augmented cascade, we are able to reduce both the key size and the tightness of the security proof by one order of magnitude. We gave simple and very efficient constructions of such a function families, based on the almost-universal hash function family of Dietzfelbinger *et al.* [DHKP97], which can be evaluated by essentially a single modular multiplication, and generically on pairwise-independent hash functions.

For the instantiation based on Matrix-DDH assumptions of [EHK$^+$13], which includes the classical constructions of Naor-Reingold [NR97] and the Lewko-Waters [LW09] as special cases, this yields asymptotically short keys consisting of only $\omega(\log \lambda)$ elements, and tight security with loss only $O(\log \lambda)$. These parameters are similar to the respective constructions of Döttling and Schröder [DS15], but our instantiation is conceptually much simpler and slightly more efficient.

For the LWE-based instantiation based of Banerjee, Peikert and Rosen [BPR12] (BPR), we are able to reduce the required size of the LWE modulus p from exponential to super-polynomial in the security parameter, which significantly improves efficiency and allows to prove security under a weaker LWE assumption. Again, the latter is similar to a result from [DS15], but we replace their relatively expensive generic construction, which requires to run $\lambda \cdot \omega(\log \lambda)$ instances of the BPR function in parallel, with a *single* instance plus an all-prefix almost-universal hash function.

We believe that APUHFs may have many further applications in cryptography beyond pseudorandom functions. This may include, for example, constructions of more efficient cryptosystems with tight provable security, such as digital signatures or public-key encryption schemes. In particular constructions using arguments similar to pseudorandom functions based on the augmented cascade, such as [CW13,GHKW16], seem to be promising targets.

References

Bel06. Mihir Bellare. New proofs for NMAC and HMAC: Security without collision-resistance. In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 602–619. Springer, Heidelberg, August 2006.

BG90. Mihir Bellare and Shafi Goldwasser. New paradigms for digital signatures and message authentication based on non-interative zero knowledge proofs. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 194–211. Springer, Heidelberg, August 1990.

BH12. Itay Berman and Iftach Haitner. From non-adaptive to adaptive pseudorandom functions. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 357–368. Springer, Heidelberg, March 2012.

BHKN13. Itay Berman, Iftach Haitner, Ilan Komargodski, and Moni Naor. Hardness preserving reductions via Cuckoo hashing. In Amit Sahai, editor, *TCC 2013*, volume 7785 of *LNCS*, pages 40–59. Springer, Heidelberg, March 2013.

BJLS16. Christoph Bader, Tibor Jager, Yong Li, and Sven Schäge. On the impossibility of tight cryptographic reductions. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 273–304. Springer, Heidelberg, May 2016.

BLMR13. Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic PRFs and their applications. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 410–428. Springer, Heidelberg, August 2013.

BMR10. Dan Boneh, Hart William Montgomery, and Ananth Raghunathan. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 10*, pages 131–140. ACM Press, October 2010.

BP14. Abhishek Banerjee and Chris Peikert. New and improved key-homomorphic pseudorandom functions. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 353–370. Springer, Heidelberg, August 2014.

BPR12. Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 719–737. Springer, Heidelberg, April 2012.

CG14. Nishanth Chandran and Sanjam Garg. Balancing output length and query bound in hardness preserving constructions of pseudorandom functions. In Willi Meier and Debdeep Mukhopadhyay, editors, *INDOCRYPT 2014*, volume 8885 of *LNCS*, pages 89–103. Springer, Heidelberg, December 2014.

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, Heidelberg, April / May 2002.

CW79. Larry Carter and Mark N. Wegman. Universal classes of hash functions. *J. Comput. Syst. Sci.*, 18(2):143–154, 1979.

CW13. Jie Chen and Hoeteck Wee. Fully, (almost) tightly secure IBE and dual system groups. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 435–460. Springer, Heidelberg, August 2013.

DHKP97. Martin Dietzfelbinger, Torben Hagerup, Jyrki Katajainen, and Martti Penttonen. A reliable randomized algorithm for the closest-pair problem. *Journal of Algorithms*, 25(1):19–51, 1997.

DS15. Nico Döttling and Dominique Schröder. Efficient pseudorandom functions via on-the-fly adaptation. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 329–350. Springer, Heidelberg, August 2015.

EHK$^+$13. Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. An algebraic framework for Diffie-Hellman assumptions. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Heidelberg, August 2013.

EHK$^+$17. Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. An algebraic framework for Diffie–Hellman assumptions. *Journal of Cryptology*, 30(1):242–288, Jan 2017.

GGM84. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. On the cryptographic applications of random functions. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 276–288. Springer, Heidelberg, August 1984.

GGM86. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, October 1986.

GHKW16. Romain Gay, Dennis Hofheinz, Eike Kiltz, and Hoeteck Wee. Tightly CCA-secure encryption without pairings. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 1–27. Springer, Heidelberg, May 2016.

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, Heidelberg, May 2012.

IKOS08. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography with constant computational overhead. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 433–442. ACM Press, May 2008.

JPT12. Abhishek Jain, Krzysztof Pietrzak, and Aris Tentes. Hardness preserving constructions of pseudorandom functions. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 369–382. Springer, Heidelberg, March 2012.

KK12. Saqib A. Kakvi and Eike Kiltz. Optimal security proofs for full domain hash, revisited. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 537–553. Springer, Heidelberg, April 2012.

Kra10. Hugo Krawczyk. Cryptographic extraction and key derivation: The HKDF scheme. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 631–648. Springer, Heidelberg, August 2010.

Lev87. Leonid A. Levin. One way functions and pseudorandom generators. *Combinatorica*, 7(4):357–363, Dec 1987.

LW09. Allison B. Lewko and Brent Waters. Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *ACM CCS 09*, pages 112–120. ACM Press, November 2009.

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, Heidelberg, May 2014.

NR97. Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th FOCS*, pages 458–467. IEEE Computer Society Press, October 1997.

Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

A Further Examples of Matrix Distributions

Let us recall some further examples for matrix distributions from [EHK+13,EHK+17] for completeness and self-containedness.

$$\mathcal{L}_2 : \mathbf{A} = \begin{pmatrix} a_1 & 0 \\ 0 & a_2 \\ 1 & 1 \end{pmatrix}, \quad \mathcal{C}_2 : \mathbf{A} = \begin{pmatrix} a_1 & 0 \\ 1 & a_2 \\ 0 & 1 \end{pmatrix}, \quad \mathcal{IL}_2 : \mathbf{A} = \begin{pmatrix} a_1 & 0 \\ 0 & a_1{+}1 \\ 1 & 1 \end{pmatrix},$$

$$\mathcal{SC}_2 : \mathbf{A} = \begin{pmatrix} a_1 & 0 \\ 1 & a_1 \\ 0 & 1 \end{pmatrix}, \quad \mathcal{U}_2 : \mathbf{A} = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \\ a_5 & a_6 \end{pmatrix},$$

where $a_1, \ldots, a_6 \xleftarrow{\$} \mathbb{Z}_q$. The corresponding transformation matrices are as follow,

$$\mathcal{L}_2 : \mathbf{T} = (\frac{1}{a_1}, \frac{1}{a_2}), \quad \mathcal{C}_2 : \mathbf{T} = (\pm\frac{1}{a_1 a_2}, \mp\frac{1}{a_2}), \quad \mathcal{IL}_2 : \mathbf{T} = (\frac{1}{a_1}, \frac{1}{a_1+1})$$

$$\mathcal{SC}_2 : \mathbf{T} = (\pm\frac{1}{a_1^2}, \mp\frac{1}{a_1}), \quad \mathcal{U}_2 : \mathbf{T} = (\frac{a_4 a_5 - a_3 a_6}{a_1 a_4 - a_2 a_3}, \frac{a_1 a_6 - a_2 a_5}{a_1 a_4 - a_2 a_3}).$$

The advantage of $\mathcal{SC}_d$ and $\mathcal{IL}_d$ is that they can be represented by one group element and have the same security guarantee as the d-Linear assumption.