CIRCULAR SECURITY RECONSIDERED

F. Betül Durak[1] and Serge Vaudenay[2]

[1] Rutgers University
Department of Computer Science
fbdurak@cs.rutgers.edu
[2] Ecole Polytechnique Fédérale de Lausanne(EPFL)
Lausanne, Switzerland
serge.vaudenay@epfl.ch

Abstract. The notion of circular security of pseudorandom functions (PRF) was introduced in Distance Bounding Protocols. So far, only a construction based on a random oracle model was proposed. Circular security stands between two new notions which we call Key Dependent Feedback (KDF) security and Leak security. So far, only a construction based on a random oracle was proposed. We give an algebraic construction based on a q-DDH assumpsion. We first prove that a small-domain Verifiable Random Functions (VRF) from Dodis-Yampolskiy is a circular secure PRF. We then use the extension to large-domain VRF by augmented cascading by Boneh et. al. This gives the first construction in the standard model.

1 Introduction

Pseudorandom functions (PRFs) were first introduced by Goldreich, Goldwasser, and Micali [10]. They play a fundamental role in cryptography with many applications. They are used for encryption, authentication, signatures, and many more cryptographic tools.

Briefly, a secure PRF is a deterministic function using a random secret key which is not distinguishable from a truly random function when used as a black box. They can be realized by random oracles. However, it is important to build cryptosystems in the standard model, i.e. without using random oracle heuristics since secure systems in the random oracle model can sometimes be trivially insecure under the instantiation of the oracle [8].

Moreover, as shown in [4], we cannot solely rely on the normal secure PRF assumption for Distance Bounding (DB) protocols, since the secret is often used as a key of PRF and is also externally used outside the PRF. In DB protocols, the circular secure PRF guarantees the normal security of PRF, even when we encrypt some functions of the key. So far, only one construction based on random oracle has been given and constructing a circular secure PRF without random oracle was left as an open problem. We present an algebraic construction of circular secure PRF in Section 4 without using random oracles. The security is based on a stronger variant of the q-DDH assumption using a fixed generator

g. The construction demonstrates that a circular secure PRF can exist without random oracles. However, making instances for DB protocols is still open.

2 Preliminaries

2.1 Pseudorandom Functions

Definition 1. *Consider a security parameter* k *and a parameter* n. *Let* f_s *be a function from* $\{0,1\}^* \to \{0,1\}^n$, *where* $s \leftarrow \{0,1\}^k$ *is chosen uniformly at random. Consider a function family* $\mathcal{F}$ *of all functions from* $\{0,1\}^*$ *to* $\{0,1\}^n$ *and a function* F *chosen from that family uniformly at random. For an adversary* $\mathcal{A}$ *limited to complexity* T, *we define the following Game:*

PRF Security Game with bit b***:***
- The challenger picks a secret s *and* $F \in \mathcal{F}$ *at random.*
- $\mathcal{A}$ *queries its oracle and gets either* $f_s(x)$ *(if* $b = 1$*) or* $F(x)$ *(if* $b = 0$*).*
- $\mathcal{A}$ *returns a bit* b'.

The advantage is $\mathbf{Adv}^{\mathsf{PRF}}_{f_s}(\mathcal{A}) = \left|\Pr[\mathcal{A}^{O_{f_s}} = 1] - \Pr[\mathcal{A}^{O_F} = 1]\right|$. *We say that the function* f_s *is a* (ϵ, T)*-secure PRF if for any distinguisher* $\mathcal{A}$ *limited to a complexity* T, *the advantage of* $\mathcal{A}$ *in the PRF Game is bounded by* ϵ.

The PRF Game is depicted on Fig. 1. We have $\mathbf{Adv}^{\mathsf{PRF}}_{f_s}(\mathcal{A}) = \Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]|$.

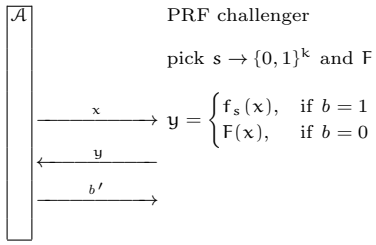

Fig. 1. PRF Game

2.2 Circular Secure Pseudorandom Functions

Definition 2. *Given a security parameter* k, *and some parameters* m, n, *consider* $s \in \{0,1\}^k$, *a family* $\mathcal{L}$ *of functions* $L : \{0,1\}^k \to \mathbb{G}^m$, *the set* $\mathcal{F}$ *of all functions* $F : \{0,1\}^* \to \mathbb{G}^n$, *where* $\mathbb{G}$ *is an additive group, and a function* F *chosen from that family. We define an oracle* $O_{s,F}(x, L, A, B) = A \cdot L(s) + B \cdot F(x)$ *using the dot product over* $\mathbb{G}$. *We assume that* L *is taken from* $\mathcal{L}$ *and* $x \in \{0,1\}^*$,

$A \in \mathbb{G}^m$, $B \in \mathbb{G}^n$. *Let* $(f_s)_{s\in\{0,1\}^k}$ *be a family of functions in* $\mathcal{F}$*. For an adversary* $\mathcal{A}$ *limited to complexity* T*, we define the following Game:*

Circular-PRF Security Game with bit b***:***
- The challenger picks a secret s *and* $F \in \mathcal{F}$ *at random.*
- $\mathcal{A}$ *queries its oracle and gets either* $A\cdot L(s)+B\cdot f_s(x)$ *(if* $b = 1$*) or* $A\cdot L(s)+B\cdot F(x)$ *(if* $b = 0$*).*
- $\mathcal{A}$ *returns a bit* b'*.*

The advantage is $\mathsf{Adv}^{\mathrm{circular}}_{f_s}(\mathcal{A}) = \left|\Pr[\mathcal{A}^{O_{s,f_s}} = 1] - \Pr[\mathcal{A}^{O_{s,F}} = 1]\right|$.

We say that the family f_s *is an* (ϵ, T)*-circular-PRF with respect to* $\mathcal{L}$ *if for any distinguisher limited to a complexity T, the advantage of distinguishing* O_{s,f_s} *from* $O_{s,F}$ *is bounded by* ϵ*.*

We require 2 conditions:
- for any pair of queries (x, L, A, B) *and* (x', L', A', B')*, if* $x = x'$*, then* $L = L'$*;*
- for any $x \in \{0,1\}^*$*, if* (x, L, A_i, B_i)*,* $i = 1, ..., \ell$ *is a list of queries using this value* x*, then*

$$\forall \lambda_1, ..., \lambda_\ell \in \mathbb{G}, \quad \sum_{i=1}^{\ell} \lambda_i B_i = 0 \quad \Rightarrow \quad \sum_{i=1}^{\ell} \lambda_i A_i = 0$$

We depict the circular-PRF Game in Fig. 2.

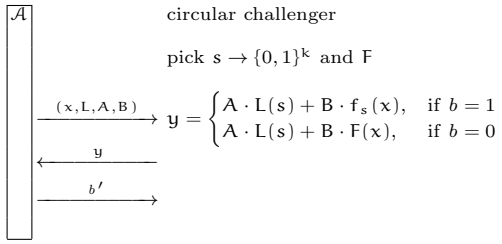

Fig. 2. circular-PRF Game

Note that the last condition implies that $B = 0 \Rightarrow A = 0$ for each query.

Def. 2 is equivalent to the circular security definition in [6] and [5], if we take for $\mathcal{L}$ the set of all linear functions. On the other hand, if $\mathcal{L}$ is a set of all functions with "polynomially bounded representation", the definition is equivalent to the circular security defined in [7]. In [7], the function L could indeed be some non-linear function. We define that $L_\mu(s) = \mathtt{map}(\mu \cdot s)$ using the dot product over $\mathbb{Z}_2^k$, where μ is a chosen vector and $\mathtt{map}$ is a given mapping from $\mathbb{Z}_2$ to $\mathbb{G}$. In the construction from [7], however, we only need the set $\mathcal{L}$ of the L_μ functions for all μ vectors and $\mathtt{map}$ is fixed.

For simplicity, we later on assume that $\mathcal{L}$ has a single element L.

For $n = 1$, we can always reduce to $B = 1$ and no x repetition, and obtain $O_{s,F}(x, L, A) = A \cdot L(s) + F(x)$.

We note that there exists no circular security if the adversary can set L to f_s(without knowing the secret s). Indeed, we let $(f_s)_{s\in\{0,1\}^k}$ be a pseudorandom family. We define an adversary $\mathcal{A}$ who queries the oracle with a tuple of $(x, L(s), A, B)$, where $x = 1$, $L(s) = f_s(1)$, and $B = -A$. The O_{s,f_s} oracle returns $A\cdot f_s(1) - A\cdot f_s(1) = 0$ if it is real oracle. Therefore, $\mathcal{A}$ outputs 1 in circular security Game, if the oracle responds with zero, and it outputs 0 otherwise. Clearly, the oracle replies the query with zero if it is the real oracle, then $\mathcal{A}$ outputs 1 with probability 1. On the other hand, if it is the ideal oracle, the response from the oracle is non-zero and $\mathcal{A}$ outputs 1 with probability bounded by $\frac{1}{p}$. Therefore, $Adv_{f_s}^{circular}(\mathcal{A}) \geqslant 1 - \frac{1}{p}$ where p is the cardinality of $\mathbb{G}$.

3 Derived PRF Notions

3.1 Secure Key-dependent Feedback PRF

Consider a security parameter k, and the parameters n and m. Let $\mathbb{G}$ be a group. Given a secret $s \leftarrow_{\$} \{0,1\}^k$, and an arbitrary function $L : \{0,1\}^k \to \mathbb{G}^m$ producing column vectors with elements in $\mathbb{G}$, we let F be a function chosen from the function family $\mathcal{F} : \{0,1\}^* \to \mathbb{G}^n$ uniformly at random. Let $(f_s)_{s\in\{0,1\}^k}$ be a family of functions from $\{0,1\}^* \to \mathbb{G}^n$. We define an oracle $O_{s,\cdot}$ such that for a matrix $M \in \mathbb{Z}^{n\times m}$ and an input $x \in \{0,1\}^*$, $O_{s,F}(x, M) = ML(s) + F(x)$ and $O_{s,f_s}(x, M) = ML(s) + f_s(x)$ using the matrix product defined from $\mathbb{Z}^{n\times m} \times \mathbb{G}^m$ to a column vector in $\mathbb{G}^n$, where each element in $\mathbb{G}^n$ is output of matrix product multiplication of each row of $M \in \mathbb{Z}^m$ with $\mathbb{G}^m$. The above is when $\mathbb{G}$ has additive notations. With multiplicative ones, we write $O_{s,f_s} = L(s)^M f_s(x)$

The condition for using O_{s,f_s} or $O_{s,F}$ is that for any pair of queries (x, M) and (x', M'), if $x = x'$, then $M = M'$. Equivalently, since f_s and F are deterministic functions, we can require that x never repeats in queries. Then, we can define an oracle $O_F(x, M) = F(x)$ which does not use M. Clearly, if x does not repeat in queries, $O_{s,F}$ is indistinguishable from O_F. This motivates the definition below.

Definition 3. *Given a security parameter k, let f_s be a function from $\{0,1\}^* \to \mathbb{G}$. Let $L : \{0,1\}^k \to \mathbb{G}^m$ be a function. For an adversary $\mathcal{A}$ limited to complexity T, we define the following Game:*

***KDF-PRF Security Game with bit** b:*
- The challenger picks a secret s and $F \in \mathcal{F}$ at random.
- $\mathcal{A}$ queries its oracle and gets either $ML(s) + f_s(x)$ (if $b = 1$) or $ML(s) + F(x)$ (if $b = 0$).
- $\mathcal{A}$ returns a bit b'.

The advantage is $Adv_{f_s}^{KDF}(\mathcal{A}) = |Pr(b' = 1|b = 0) - Pr(b' = 1|b = 1)|$.

We say that the family $(f_s)_{s\in\{0,1\}^k}$ is a (ϵ, T) Key-dependent Feedback secure (KDF-secure) PRF with respect to L if for any distinguisher limited to a complexity T, the advantage of $\mathcal{A}$ in the KDF-PRF Game is bounded by ϵ.

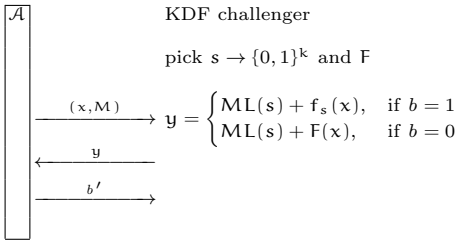

Fig. 3. KDF-PRF Game

The corresponding KDF-PRF Game is depicted in Fig. 3.

Lemma 1. *(Circular security implies KDF security) Let f_s be any PRF to $\mathbb{G}^m$ where $\mathbb{G}$ is a group. For any KDF adversary $\mathcal{A}$ for f_s of complexity T, there exists a circular adversary $\mathcal{B}$ for f_s of complexity $T + \mathcal{O}(nmQ)$, where Q is the number of queries made by $\mathcal{A}$ such that:*

$$Adv_{f_s}^{KDF}(\mathcal{A}) = Adv_{f_s}^{circular}(\mathcal{B})$$

Proof. Given an adversary $\mathcal{A}$ playing against KDF-secure oracle, we build another adversary $\mathcal{B}$ that plays against circular-secure oracle. Let (x_i, M_i) be a query made by an adversary $\mathcal{A}$ against its KDF-secure oracle. We define the adversary $\mathcal{B}$ simulating $\mathcal{A}$ by taking its queries, and transforming each (x_i, M_i) into $(x_i, L, A_{i,j}, B_{i,j})$ queries. For each (x_i, M_i), the adversary $\mathcal{B}$ sets $A_{i,j}$ as the j^{th} row of M_i, where $1 \leqslant j \leqslant n$, and set $B_{i,j}$ to the j^{th} row of the $n \times n$ identity matrix. Notice that, since the linear combinations of $B_{i,j}$s do not vanish (they are the rows of identity matrix), we do not have any problem with the condition that for the queries $(x_i, L, A_{i,j}, B_{i,j})$, the linear combinations of $A_{i,j}$ vanishes with same x_i whenever the linear combination of $B_{i,j}$s vanishes in $\mathcal{B}$'s queries. $\mathcal{B}$ uses these queries to query its circular secure oracle and responds them with the replies it gets from its oracle. When $\mathcal{A}$ is done with its queries, it returns its output. Then, $\mathcal{B}$ uses the same output to return its oracle as its output. Hence, the advantage of $\mathcal{A}$ is equal to the advantage of $\mathcal{B}$. If the simulation of $\mathcal{A}$ wins, so is $\mathcal{B}$. Therefore, any PRF which is (ϵ, Q)-circular secure is also KDF-secure. □

Lemma 2. *(KDF security implies non-adaptive circular security) Let f_s be any PRF. Let $\mathbb{G}$ be a group in KDF-security Game. For any circular adversary $\mathcal{B}$ of complexity T making non-adaptive queries on the same x, there exists a KDF adversary $\mathcal{A}$ of complexity $T + \mathcal{O}((n^2 + m^2 + n^3)Q)$ such that:*

$$Adv_{f_s}^{circular}(\mathcal{B}) = Adv_{f_s}^{KDF-secure}(\mathcal{A})$$

Proof. Given a non-adaptive adversary $\mathcal{B}$ playing with a circular-secure oracle, we build another adversary $\mathcal{A}$ that plays with the KDF-secure oracle. We take all Q non-adaptive queries as (A_i, B_i) for each x, where $1 \leqslant i \leqslant Q$, $A_i \in \mathbb{Z}^m$ and $B_i \in \mathbb{Z}^n$ made by circular adversary $\mathcal{B}$, we transform the queries (A_i, B_i) made

by circular adversary $\mathcal{B}$ into a pair of matrix (A, B) of size $Q \times m$ and $Q \times n$ respectively. We define the matrices $A = (A_1 \cdots A_Q)^T$ and $B = (B_1 \cdots B_Q)^T$ formed by rows of A_i and B_i respectively. We know that for any row λ, $\lambda \cdot B = 0$ implies $\lambda \cdot A = 0$. So, if we take a vector X of n undeterminates, any combination $\lambda \cdot BX$ vanishing implies $\lambda \cdot A = 0$. So, the equation $BM = A$ has a solution M in $\mathbb{Z}^{n \times m}$. We make the KDF query (x, M) to get $y = M \times L(s) + f(x)$. Then, by $BM \cdot L(s) + B \times f(x) = A \times L(s) + B \times f(x)$ so we obtain the answer of the circular oracle.

Hence, if $\mathcal{B}$ wins against its circular security oracle, $\mathcal{A}$ wins with the same advantage and with complexity $T + \mathcal{O}((n^2 + m^2 + n^3)Q)$.

□

Let f_s be any PRF. When we define the adversaries as non-adaptive adversaries, the previous two lemmas imply that f_s is non-adaptive circular-secure if and only if it is non-adaptive KDF-secure.

For $n = 1$, since x never repeats, we can see that the circular security and KDF security are equivalent.

We start our attempt to construct a KDF-secure PRF with 2 negative examples. In the first example, we define $f_s(x) = x^s$, which is shown to be not secure PRF based on Def. 1. Similarly, in the second negative example, we define $f_s(x) = g^x h^s$, and show that it is an insecure PRF under Def. 1.

Example 1. Let $f_s(x)$ be a function from $\mathbb{Z} \to Z_p^*$ for a prime number p defined as $f_s(x) = x^s$. $f_s(x)$ is not a secure PRF.

Let us make a single query with $x = 1$ to normal-secure PRF oracle. If we interact with the real oracle, the oracle returns $O_{s,f_s}(x) = x^s$. Clearly, the result we will get is 1, if the oracle is real, and we get a random integer if the oracle is random. It allows us to distinguish between O_{s,f_s} and $O_{s,F}$

Example 2. Let $f_s(x)$ be a function from $\mathbb{Z}$ to G for a group G, where $g, h \in G$ are arbitrary, defined as $f_s(x) = g^x h^s$. $f_s(x)$ is not a secure PRF.

Let us make two queries as $2x, x$ to normal-secure PRF oracle. If we interact with the real oracle, the oracle returns $O_{s,f_s}(2x) = g^{2x} h^s$ and $O_{s,f_s}(x) = g^x h^s$ respectively. Clearly, when we divide the results, we get g^x, which does not depend on the secret s, if the oracle is real, and we get a random string if the oracle is random. It allows us to distinguish between O_{s,f_s} and $O_{s,F}$.

3.2 Leak-PRF security

Definition 4. *Given a security parameter k, let f_s be a function from $\{0,1\}^* \to \mathbb{G}$. Let $L : \{0,1\}^k \to \mathbb{G}^m$ be a function respectively let $L_g : \{0,1\}^k \to \mathbb{G}^m$ be a function for all g in a given set. For an adversary $\mathcal{A}$ limited to complexity T, we define the Leak-PRF game (respectively the rnd-Leak-PRF Game) as follows:*

Leak-PRF (respectively rnd-Leak-PRF) Security Game with bit b***:***
- The challenger picks a secret s*,* $\mathsf{F} \in \mathcal{F}$ *(and* g *in a given set) at random.*
- The challenger computes $\mathsf{L}(\mathsf{s})$ *(respectively* $\mathsf{L}_{\mathsf{g}}(\mathsf{s})$ *corresponding to random* g*) and gives it (and* g*) to* $\mathcal{A}$*.*
- $\mathcal{A}$ *queries its oracle and gets either* $\mathsf{y}_1 = \mathsf{f}_{\mathsf{s}}(\mathsf{x})$ *(if* $b = 1$*) or* $\mathsf{y}_0 = \mathsf{F}(\mathsf{x})$ *(if* $b = 0$*).*
- If $\mathcal{A}$ *repeats a query* x*, the game aborts.*
- $\mathcal{A}$ *returns a bit* b'*.*

The advantage is $\mathbf{Adv}^{\mathsf{Leak}}_{\mathsf{f}_{\mathsf{s}}}(\mathcal{A})(= \mathbf{Adv}^{\mathsf{rnd-Leak}}_{\mathsf{f}_{\mathsf{s}}}(\mathcal{A})) = |\mathbf{Pr}(b' = 1|b = 0) - \mathbf{Pr}(b' = 1|b = 1)|$.

The function f_{s} *is a* (ϵ, T)*-secure* Leak*-PRF (respectively rnd-Leak-PRF) with respect to* L *if for any adversary* $\mathcal{A}$ *limited to the complexity* T*, the advantage of* $\mathcal{A}$ *in the* Leak*-PRF Game is bounded by* ϵ*.*

The Leak-PRF (respectively rnd-Leak-PRF) Game is depicted in Fig. 4 (respectively in Fig. 5).

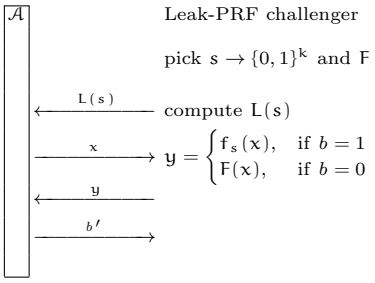

Fig. 4. Leak-PRF Game

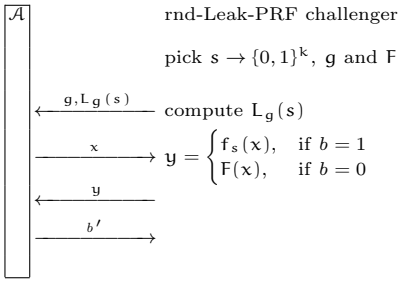

Fig. 5. rnd-Leak-PRF Game

Theorem 1. *(*Leak*-PRF implies KDF-security) Let* f_{s} *from* $\{0,1\}^* \to \mathbb{G}$ *be any PRF. We define* $\mathsf{Leak}(\mathsf{s}) = \mathsf{L}(\mathsf{s})$ *in* Leak*-PRF Game. For any* (ϵ, T)*-secure KDF adversary for* L*, there exists a* Leak *adversary* $\mathcal{B}$ *complexity* $\mathsf{T} + \mathcal{O}(\mathsf{Q})$*, where* Q *is the number of queries made by* $\mathcal{A}$ *s.t.*

$$\mathbf{Adv}^{\mathsf{KDF}}_{\mathsf{f}_{\mathsf{s}}}(\mathcal{A}) = \mathbf{Adv}^{\mathsf{Leak}}_{\mathsf{f}_{\mathsf{s}}}(\mathcal{B})$$

Proof. Given an adversary $\mathcal{A}$ playing against KDF-secure oracle with $\mathsf{L}(\mathsf{s})$, we build another adversary $\mathcal{B}$ that plays against Leak-PRF Game where $\mathsf{Leak}(\mathsf{s}) = \mathsf{L}(\mathsf{s})$. In this Game $\mathcal{B}$ obtains $\mathsf{L}(\mathsf{s})$ from its challenger as an output to its Leak function. $\mathcal{B}$ simulates $\mathcal{A}$'s queries $(\mathsf{M}_{\mathsf{i}}, \mathsf{x}_{\mathsf{i}})$ for $\mathsf{i} = 1..\mathsf{Q}$ as following: $\mathcal{B}$ queries its oracle with x_{i} and receives either $\mathsf{y} = \mathsf{f}_{\mathsf{s}}(\mathsf{x}_{\mathsf{i}})$ or $\mathsf{y} \leftarrow_{\$} \mathbb{G}$. $\mathcal{B}$ adds y with $\mathsf{ML}(\mathsf{s})$ using the leak of the secret to send $\mathsf{ML}(\mathsf{s}) + \mathsf{y}$ to $\mathcal{A}$. $\mathcal{A}$ outputs a bit and $\mathcal{B}$ outputs its Leak-challenger with the same bit as $\mathcal{A}$. Hence if $\mathcal{A}$ wins against its oracle, $\mathcal{B}$ wins with the same advantage and with the complexity $\mathsf{T} + \mathsf{Q}$.

□

4 Algebraic Construction

4.1 The Dodis-Yampolskiy Construction

The **q- decisional Diffie-Hellman** problem is defined in [3] as follows:
Let $\mathbb{G}$ be a group of prime order p. For $a \leftarrow_{\$} \mathbb{Z}_p$ and $g \in \mathbb{G}$ picked uniformly at random, given a q-tuple $(g, g^a, g^{a^2}, \cdots g^{a^{q-1}})$, the q-DDH assumption states that $g^{\frac{1}{a}}$ is indistinguishable from a random element in $\mathbb{G}$. More precisely, for any adversary $\mathcal{A}$, the advantage of distinguishing $g^{\frac{1}{a}}$ from a random element in $\mathbb{G}$ is bounded by ϵ.

Definition 5. *For $q > 1$, given a group $\mathbb{G}$ of prime order p, we define $Adv_q^{DDH}[\mathcal{A}, \mathbb{G}]$ $= Pr[\mathcal{A}(g, g^a, \cdots, g^{a^{q-1}}, g^{\frac{1}{a}}) = 1] - Pr[\mathcal{A}(g, g^a, \cdots, g^{a^{q-1}}, h) = 1] \leqslant \epsilon$ where the probability is over random choice of g, h, and a. We say that the (T, q, ϵ)-DDH assumption holds in $\mathbb{G}$, if for all poly-time T adversary $\mathcal{A}$, the $Adv_q^{DDH}[\mathcal{A}, \mathbb{G}]$ advantage is at least ϵ.*

When we let g be a generator of the group $\mathbb{G}$ and fix it, we define the (g, q)-DDH assumption as follows:

Definition 6. *For $q > 1$, we define $Adv_{g,q}^{DDH}$ similarly for g fixed and a probability over the random choice of h and a. We say that the (t, g, q, ϵ)-DDH assumption holds in $\mathbb{G}$, if for all poly-time T adversary $\mathcal{A}$, the $Adv_{g,q}^{DDH}[\mathcal{A}, \mathbb{G}]$ advantage is at least ϵ.*

The q-DDH assumption is defined with a random generator while we fix the generator g in the (g, q)-DDH assumption. Clearly, any poly-time q-DDH adversary $\mathcal{A}$ has the same advantage of some poly time (g, q)-DDH adversary by using some randomization tricks. We state that (g, q)-DDH assumption implies q-DDH assumption. However, the other direction does not seem to hold.

Surprisingly, we have the implication for both directions for the computational-DH (CDH) problem.

Theorem 2. *(Leak-PRFness of the Dodis-Yampolskiy function [9]) Let k be a security parameter and $\mathbb{G}$ be a group of prime order p generated by some g. Assume that $(T + Qq.poly(k), g, q, \epsilon)$-DDH assumption holds in $\mathbb{G}$. Then, $f_{s,h}(x) = h^{\frac{1}{x+s}}$ where $h \in \mathbb{G}$, $s \in \mathbb{Z}_p$ and x is in a domain D defined as a subset of $\mathbb{Z}_p$ of size Q where $Q \leqslant q$, is an $(\epsilon Q + \frac{Q^2}{p}, T)$-secure Leak-PRF for $L_g(s, h) = (g, g^s, ..., g^{s^{q-1}}, h, h^s, ..., h^{s^{q-Q}})$ over D. More precisely,*

$$Adv_{f_{s,h}}^{Leak}(\mathcal{A}) \leqslant \sum_{i=0}^{Q-1} Adv_{g,q}^{DDH}(\mathcal{B}_i, \mathbb{G}) + \frac{Q^2}{p}$$

for some distinguisher $\mathcal{B}_i$, where $i = 0, ..., Q-1$.

We have the same statements with q-DDH and rnd-Leak-PRF security but L_g defined on a random g. And, the proof follows as same.

Proof. Suppose there exists an adversary $\mathcal{A}$ that plays Leak-PRF security Game to distinguish between $f_{s,h}(x) = h^{\frac{1}{x+s}}$ and a random element in $\mathbb{G}$. Let $D = \{x_1, ..., x_Q\}$. We design a sequence of games Game_i for $i = 0, ..., Q$ between a challenger and the Leak-PRF adversary $\mathcal{A}$. We define the probability p_i to output 1 of $\mathcal{A}$ in Game_i, where Game_i is defined as:
- The challenger picks a secret (s, h) at random and reveals $\mathsf{Leak}(s, h) = (g, g^s, ..., g^{s^{q-1}}, h, h^s, ..., h^{s^{q-Q}})$ to $\mathcal{A}$.
- The challenger also picks a random function F to answer the queries x_j from $\mathcal{A}$ with:
  - if $j \leqslant i$, the challenger answers by $F(x_j)$.
  - if $j > i$, the challenger answers by $f_{s,h}(x_j)$.

Note that the way to answer depends on the value x_j of the query and not on the sequence number of the query in time.

It is clear that Game_0 is the Leak-PRF Game with real function $f_{s,h}$ and Game_Q is the Leak-PRF Game with random function F. Hence, the advantage of $\mathcal{A}$ to distinguish between $f_{s,h}(x) = h^{\frac{1}{x+s}}$ and a random element in $\mathbb{G}$ is $|p_0 - p_Q|$. We like to show that $|p_0 - p_Q|$ is negligible. Given the sequence of games, we build an adversary called $\mathcal{B}_i$ such that $|p_i - p_{i+1}| = \mathsf{Adv}^{\mathrm{DDH}}_{g,q}(\mathcal{B}_i, \mathbb{G}) + \frac{Q}{p}$ for $0 \leqslant i \leqslant Q - 1$. Then, we achieve that $|p_0 - p_Q| = \sum_i \mathsf{Adv}^{\mathrm{DDH}}_{g,q}(\mathcal{B}_i, \mathbb{G}) + \frac{Q^2}{p}$. Thus, we only need to prove that Game_i is indistinguishable from Game_{i+1}.

We build our adversary $\mathcal{B}_i$ that uses $\mathcal{A}$ to break the (t, q, ϵ)-DDH assumption in group $\mathbb{G}$. In other words, when an adversary $\mathcal{B}_i$ is given a challenge tuple $(g, g^a, ..., g^{a^{q-1}}, \Gamma) \in \mathbb{G}^{q+1}$, where Γ is either $g^{\frac{1}{a}}$ or a random element in $\mathbb{G}$, $\mathcal{B}$ can distinguish Γ by using $\mathcal{A}$.

We start with $\mathcal{B}_i$ given its challenge tuple to simulate the queries made by $\mathcal{A}$ to its oracle. The adversary $\mathcal{B}_i$ simulates $\mathcal{A}$ by taking its challenge query and responding it using its own challenge tuple $(g, g^a, ..., g^{a^{q-1}}, \Gamma)$ as follows:

$\mathcal{B}_i$ sets $s = a - x_i$ to generate a private key for adversary $\mathcal{A}$ and selects a random $r \in \mathbb{Z}_p^*$. It does not know what s is because a is not known. Using Binomial Theorem, $\mathcal{B}_i$ computes $(g, g^s, g^{s^2}, ..., g^{s^{q-1}})$ from $(g, g^a, ..., g^{a^{q-1}})$. Define the function $f(z) = r \times \Pi_{y \in D - \{x_i\}}(z + y) = \sum_{j=0}^{Q-1} c_j z^j$, where $y \neq x_i$. Since $\mathcal{B}$ knows g^{s^j}, where $1 \leqslant j \leqslant q - 1$ and $Q \leqslant q$, it computes $h = g^{f(s)}$ as follows:

$$g^{f(s)} = g^{\sum_{j=0}^{q-1}(c_j s^j)} = \Pi_{j=0}^{q-1}(g^{s^j})^{c_j}$$

$\mathcal{B}_i$ can further compute $h^s, ...h^{s^{q-Q}}$ similarly.

In the (g, q)-DDH challenge, we pick $a \in \mathbb{Z}_p$ uniformly at random. We know that g is a generator and that $r \neq 0$ is random. If $f(s) \neq 0$, or equivalently, $a \neq x_i - x_j$ for all $j \neq i$, we have that (s, h) is uniformly distributed among pairs such that $h \neq 1$ and $s \neq -x_j$ for all $j \neq i$. So, (s, h) follows a distribution which is indistinguishable from the one in Game_i to Game_{i+1}. More precisely, the failure probability that a is some $x_j - x_i$ is $\frac{Q-1}{p}$. The failure probability that

$h = 1$ is $\frac{1}{p}$. So, the cumulated failure probability between the (g, q)-DDH game, Game_i and Game_{i+1} is bounded by $\frac{Q}{p}$.

Then, $\mathcal{B}_i$ gives the tuple $\mathsf{Leak}(s, h) = (g, g^s, ..., g^{s^{q-1}}, h, h^s, ..., h^{s^{q-Q}})$ to $\mathcal{A}$. Let (x_j) be a query made by $\mathcal{A}$ to its Leak-secure PRF oracle, where $1 \leqslant j \leqslant Q$. Wherever $\mathcal{A}$ queries the challenger $\mathcal{B}_i$ with x_j
- if $j < i$, $\mathcal{B}_i$ simulates the answer to $\mathcal{A}$ with $F(x_j)$ by lazy sampling.
- if $j > i$, $\mathcal{B}_i$ simulates the answer to $\mathcal{A}$ with $f_{s,h}(x_j)$ as follows:

Let $f_j(s)$ be a function defined as:

$$f_j(s) = \frac{f(s)}{s+x_j} = \sum_{j=0}^{q-2} d_j s^j$$

where it is polynomial of degree $q - 2$. Notice that $f_{s,h}(x_j) = h^{\frac{1}{s+x_j}} = g^{f_j(s)}$ is computable by $\mathcal{B}_i$ from the tuple $(g, g^s, g^{s^2}, ..., g^{s^{q-1}})$.
- if $j = i$, $\mathcal{B}_i$ answers as following:

Let $f_i(s)$ be another function defined as:

$$f_i(s) = \frac{f(s)}{s+x_i} = \sum_{i=0}^{q-2} \gamma_i s^i + \frac{\gamma}{a}$$

Notice that $f(s)$ is not divisible by $(s+x_i)$, so $\gamma \neq 0$. $\mathcal{B}_i$ replies the challenge query (x_i) by computing $y = (\Gamma)^\gamma g^{\sum_{i=0}^{q-2} \gamma_i s^i}$.

If $\Gamma = g^{\frac{1}{a}} = g^{\frac{1}{s+x_i}}$, then y is $g^{f_i(s)} = f_{s,h}(x_i)$. If Γ is random, since $\gamma \neq 0$, y is random as well.

Clearly, if Γ in $\mathcal{B}_i$'s challenge tuple is $g^{\frac{1}{a}}$, then we are in Game_{i+1}. Otherwise, we are in Game_i. Hence, $|p_i - p_{i+1}| \leqslant \mathsf{Adv}^{DDH}_{g,q}(\mathcal{B}_i, \mathbb{G}) + \frac{Q}{p}$.

Therefore, we have $|p_0 - p_Q| \leqslant Q\varepsilon + \frac{Q^2}{p}$.

The running time of the reduction is upper bounded by simulating oracle queries by $\mathcal{B}_i$. Per query, $\mathcal{B}_i$ performs $3q-2$ multiplications and exponentiations which take $(3q - 2).\mathsf{poly}(k)$. Since $\mathcal{A}$ can make at most Q queries, the running time of $\mathcal{A}$ is bounded by $Qq.\mathsf{poly}(k) = t$. Hence, $f_{s,g}(x)$ is a $(\epsilon q, Qq.\mathsf{poly}(k))$-secure Leak-PRF.

□

4.2 Extension to KDF-Security and Circular Security

We have just shown that a function $f_{s,h}(x) = h^{\frac{1}{s+x}}$ defined from $[\mathbb{Z} \times \mathbb{G}] \times D$ to $\mathbb{G}$, where D is a subset of $\mathbb{Z}_p$ of size q, is a Leak-secure pseudorandom function for a small domain size q under (g, q)-DDH assumption.

Theorem 3. *(KDF security of the Dodis-Yampolskiy function) Let k be a security parameter and $\mathbb{G}$ be a group of prime order p generated by some g. Assume that $(t + q^2.\mathsf{poly}(k), g, q, \epsilon)$-DDH assumption holds in $\mathbb{G}$. We define*

$L(s,h) = (g^s, h)$. *Then,* $f_{s,h}(x) = h^{\frac{1}{x+s}}$ *where* $h \in \mathbb{G}$, $s \in \mathbb{Z}_p$ *and* x *is in a domain* D *defined as a subset of* $\mathbb{Z}_p$ *of size* q, *is a* $(q\epsilon + \frac{q^2}{p}, T)$*-secure KDF-secure PRF for* $L(s,h)$ *when the real oracle defined as* $O_{s,h,f}(x, M) = L(s,h)^M f(x) = g^{\alpha s} h^{\beta} f_{s,h}(x)$ *for* $M = (\alpha, \beta)$.

The proof follows from Thm. 1 and 2.

For the parameter $n = 1$, KDF-security is equivalent to circular security. So, $f_{s,h}$ is both KDF-secure and circular-secure for L under the (g, q)-DDH assumption.

4.3 Parallel Leak security

Definition 7. *Consider a security parameter* k, *a set* K, *an integer* t, *a group* $\mathbb{G}$ *and a secure PRF* $f_{s,h} : [\mathbb{Z} \times \mathbb{G}] \times D \to \mathbb{G}$, *where the domain* $D \subset \mathbb{Z}_p$ *is of size* q *and the secret consists of* $s \in \mathbb{Z}$ *and* $h \in K$. *We let* $L(s, h_i)$ *be a leak function for* $1 \leqslant i \leqslant t$. *We define* t *related keys as* $(s, h_1), ..., (s, h_t)$, *where* $h_i \in K$. *We define* $\mathsf{Leak}(s, h_1, ..., h_t) = (L(s, h_1), ..., L(s, h_t))$ *and* $f^t_{s,h_1,...,h_t}(x, i) = f_{s,h_i}(x)$.

We say that the function $f_{s,h}$ *is a* t*-parallel* Leak *secure for* L *if the function* $f^t_{s,h_1,...,h_t}$ *is* Leak*-secure for* Leak.

We state that if the function $f_{s,h}$ defined in Thm. 3 is a Leak-secure PRF and (g, q)-DDH assumption holds in $\mathbb{G}$, then $f^t_{s,h_1,...,h_t}$ is a t-parallel Leak secure PRF for all q polynomial with the following Lemma.

Lemma 3. *(Parallel* Leak *security of the Dodis-Yampolskiy function) We let* $f_{s,h}(x) = h^{\frac{1}{x+s}}$ *be a function in* $\mathbb{G}$ *generated by some* g, *in which the* (g, q)*-DDH assumption holds. The input* x *is defined as an element of a domain* D *of size* Q, *where* $Q \leqslant q$. *For every* t*-parallel* Leak *secure adversary* $\mathcal{A}$ *for* $L_g(s, h_i) = (g, g^s, ..., g^{s^{q-1}}, h_i, h_i^s, ..., h_i^{s^{q-Q}})$, *there exists a* Leak *adversary* $\mathcal{B}_0$ *for* L_g *and* (g, q)*-DDH adversary* $\mathcal{B}_1$ *such that*

$$\mathsf{Adv}^{\mathsf{Leak}}_{f^t_{s,h_1,...,h_t}}(\mathcal{A}) \leqslant \mathsf{Adv}^{\mathsf{Leak}}_{f_{s,h}}(\mathcal{B}_0) + t.\mathsf{Adv}^{\mathsf{DDH}}_{g,q}(\mathcal{B}_1, \mathbb{G})$$

We can state a same Lemma with q*-DDH assumption and rnd-Leak-PRF security but* L_g *depends on a random* g. *The proof follows as same.*

Proof. The proof uses a sequence of three Games between a challenger and a parallel Leak secure PRF adversary $\mathcal{A}$ that attacks $f^t_{s,h_1,...,h_t}$. For $i = 0, 1, 2, 3$, we define the probability to win for $\mathcal{A}$ as p_i at the end of Game i.

Game 0. (Fig. 6) The challenger picks a random key as $(s, h_1, ..., h_t)$. The t-parallel Leak adversary $\mathcal{A}$ receives $L_g(s, h_i)$ for $1 \leqslant i \leqslant t$ and queries its challenger with (x, i). The challenger behaves as a real oracle for $f^t_{s,h_1,...,h_t}$, meaning that it replies the query with $h_i^{\frac{1}{x+s}}$.

Game 1. (Fig. 7) The challenger picks a random function $u : D \to \mathbb{G}$, random exponents $r_1, ..., r_t$ in $\mathbb{Z}_p$, and s, h. It sets $h_i = h^{r_i}$. An adversary $\mathcal{A}$ receives

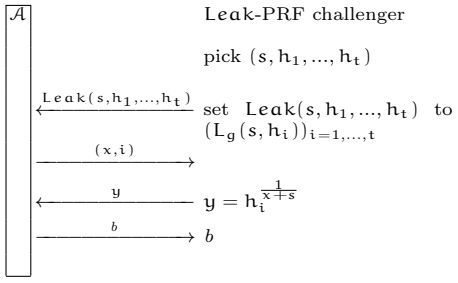

Fig. 6. Game 0.

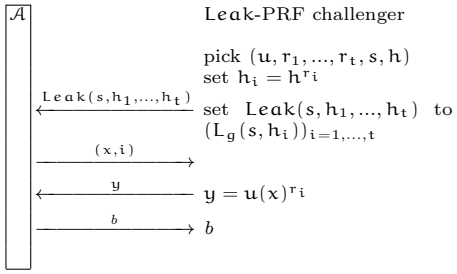

Fig. 7. Game 1.

$L_g(s, h_i)$ for $1 \leqslant i \leqslant t$ and queries its challenger with (x, i). The challenger replies the query with $u(x)^{r_i}$.

We show that Game 0 and Game 1 are indistinguishable if $f_{s,h}$ is a Leak secure PRF. We construct a Leak-secure adversary B_0 whose running time is same as $\mathcal{A}$ and such that

$$|p_1 - p_0| = Adv^{Leak}_{f_{s,h}}(\mathcal{B}_0) \tag{1}$$

The Leak adversary $\mathcal{B}_0$ interacts with its Leak oracle and simulates the $f^t_{s,h_1,...,h_t}$ challenger for $\mathcal{A}$. More precisely, $\mathcal{B}_0$ receives its $L_g(s, h) = (g, g^s, ..., g^{s^{q-1}}, h, h^s, ..., h^{s^{q-Q}})$ from its challenger and chooses random $r_1, ..., r_t \in \mathbb{Z}_p$. Then, $\mathcal{B}_0$ computes $Leak(s, h_i) = (g, g^s, ..., g^{s^{q-1}}, h_i, h_i^s, ..., h_i^{s^{q-Q}})$, where $h_i = h^{r_i}$ for $1 \leqslant i \leqslant t$. Whenever $\mathcal{A}$ issues a query with (x, i), $\mathcal{B}_0$ queries its Leak oracle with (x) to obtain its response y and $\mathcal{B}_0$ responds $\mathcal{A}$ with y^{r_i}. Finally, $\mathcal{B}_0$ outputs same as $\mathcal{A}$'s output.

When Leak oracle responds $\mathcal{B}_0$'s query, $y = h^{\frac{1}{x+s}}$ with random key (s, h), then $\mathcal{B}_0$ response to $\mathcal{A}$ is $y^{r_i} = h_i^{\frac{1}{x+s}}$, where we define $h_i = h^{r_i}$. Hence, in this case, $\mathcal{B}_0$ simulates Game 0. See Fig. 8.

When Leak oracle responds $\mathcal{B}_0$'s query with a random function $y = u(x)$, then $\mathcal{B}_0$ response to $\mathcal{A}$ is $y^{r_i} = u(x)^{r_i}$. Hence, in this case, $\mathcal{B}_0$ simulates Game 1. See Fig. 9.

Thus, we prove the equation (1).

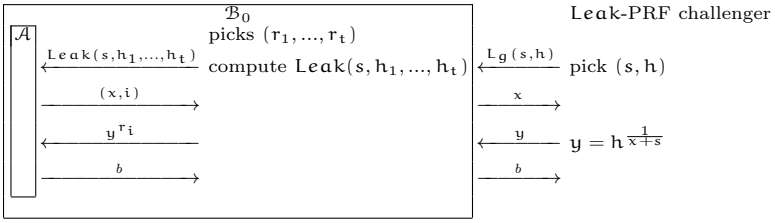

Fig. 8. Leak-PRF Game (real)

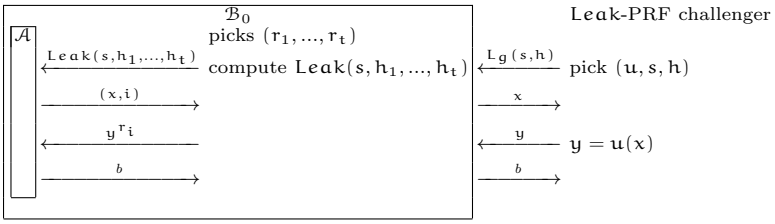

Fig. 9. Leak-PRF Game (ideal)

Game 2. The challenger picks a random function $\omega : D \times [t] \to \mathbb{G}$ and some $h_1, ..., h_t$. The adversary $\mathcal{A}$ receives $L_g(s, h_i)$ for $1 \leqslant i \leqslant t$ and queries its challenger with (x, i). The challenger replies the query with $\omega(x, i)$.

The proof for indistinguishability of Game 1 and Game 2 follows from [2, Lemma 1], where we have $|p_1 - p_2| \leqslant t.Adv^{DDH}_{g,q}(\mathcal{B}_1, \mathbb{G})$ with a (g, q)-DDH adversary $\mathcal{B}_1$.

The advantage of $Adv^{KDF}_{f^t_{s,h_1,...,h_t}}(\mathcal{A})$ which is equal to $|p_0 - p_2|$ is bounded by $Adv^{KDF}_{f_{s,h}}(\mathcal{B}_0) + t.Adv^{DDH}_{g,q}(\mathcal{B}_1, \mathbb{G})$ as it is claimed. This completes the proof. □

4.4 The Boneh-Montgomery-Raghunathan Augmentation

In [1], a classical cascade function constructs a PRF with a large domain from a PRF with a small domain by cascading. Given that, in [3], an algebraic PRF structure is constructed based on the extended results of this classical cascade function. However, as stated in [3], the classical cascade construction requires the output of the underlying PRF to be at least as long as its secret key. Boneh et al. eliminates the requirement by injecting a supplemental secret. Therefore, we will use Boneh-Montgomery-Raghunathan's augmented cascade result.

The augmented cascade pseudorandom function, defined in [3], gives a secure PRF with domain D^n from a secure PRF with domain D, where $D \subset \mathbb{Z}_p$ of size q. More precisely, let $f_{s,h} : [\mathbb{Z} \times \mathbb{G}] \times D \to \mathbb{G}$ be a secure PRF. The augmented cascade PRF of $f_{s,h}$, denoted as $f^{*n}_{s_1,...,s_n,h} : [\mathbb{Z}^n \times \mathbb{G}] \times D^n \to \mathbb{G}$ is defined on input key $(s_1, ..., s_n, h) \in [\mathbb{Z}^n \times \mathbb{G}]$ and value $(x_1, ..., x_n) \in D^n$ as:

$h_0 = h$
for $i = 1, ..., n$ do
  $h_i \leftarrow f_{s_i,h_{i-1}}(x_i)$
output h_n.

If we plug $f_{s,h}(x) = h^{\frac{1}{s+x}}$ in an augmented cascade, we obtain a secure pseudorandom function $f^{*n}_{s_1,...,s_n,h}(x_1, ..., x_n) = h^{\frac{1}{(s_1+x_1)...(s_n+x_n)}}$ in exponential domain size q^n.

Theorem 4. *Let $\mathbb{G}$ be a group of prime order p generated by some g. Assume that (t, g, q, ϵ)-DDH assumption holds in $\mathbb{G}$. Let $L_g(s_1, ..., s_n, h) = (g^{s_1}, ..., g^{s_n}, h)$. We define $f^{*n}_{s_1,...,s_n,h}$ as in Boneh-Montgomery-Raghunathan augmentation over D^n where D is size of q. The augmented cascade $f^{*n}_{s_1,...,s_n,h} = h^{\frac{1}{(s_1+x_1)...(s_n+x_n)}}$ is a Leak-secure PRF. More precisely,*

$$Adv^{Leak}_{f^{*n}_{s_1,...,s_n,h}}(\mathcal{A}) = \sum_{i=1}^{n} Adv^{Leak}_{f^{t}_{s,h_1,...,h_t}}(\mathcal{B}_i)$$

for some t-parallel Leak adversary $\mathcal{B}_i$.

Proof. The proof uses a hybrid argument where we define the hybrids as following: Let $\mathcal{A}$ be a Leak-PRF adversary playing against augmented cascade function. We construct hybrid game H_i for $0 \leqslant i \leqslant n$ (shown in Fig. 10). The challenger picks a random function $F : D^i \mapsto \mathbb{G}$ and random keys $(s_1, ..., s_n, h) \in \mathbb{Z}^n \times \mathbb{G}$. $\mathcal{A}$ gets its $L_g(s_1, s_2, ..., s_n, h)$ function and plays the regular PRF Game: he submits a query $(x_1, ..., x_n)$. The challenger applies the function F to obtain h_i and then iteratively computes h_n:

$h_i = F(x_1, ..., x_i)$
for $j = i + 1, ..., n$ do
    $h_j \leftarrow f_{s_j, h_{j-1}}(x_j)$
output h_n.

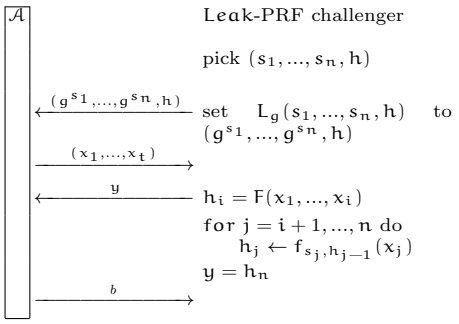

Fig. 10. H_i Game against cascade function.

The challenger returns h_n to $\mathcal{A}$. Let p_i be the probability that $\mathcal{A}$ returns 1 in H_i. It is clear that in H_0, the adversary $\mathcal{A}$ interacts with f^{*n} while in H_n, it interacts with a random function $F : D^n \mapsto \mathbb{G}$. Therefore, the Leak-PRF advantage of $\mathcal{A}$ is $Adv^{Leak}_{f^{*n}}(\mathcal{A}) = |p_n - p_0| = \sum_i (p_i - p_{i-1})$.

We construct a t-parallel Leak adversary $\mathcal{B}_i$ such that $Adv^{Leak}_{f^{t}_{s,h_1,...,h_t}}(\mathcal{B}_i) = |p_{i+1} - p_i|$ (in Fig. 11, we show the construction where the Leak-PRF challenger replied with real function). The adversary $\mathcal{B}_i$ simulates the challengers in H_i or

H_{i+1}. To do that, $\mathcal{B}_i$ needs to simulate a random function $F : D^i \mapsto \mathbb{G}$. For this purpose, $\mathcal{B}_i$ defines an injection $\mathsf{Index} : D^{i-1} \mapsto \{1, ..., t\}$.

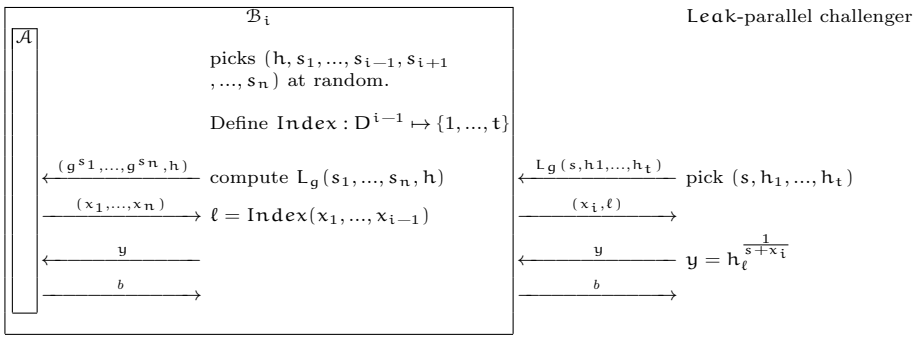

Fig. 11. Leak-PRF Game (real)

Now, $\mathcal{B}_i$ receives $\mathsf{Leak}(s, h_1, ..., h_t) = (g, g^s, ..., g^{s^{q-1}}, h_k^s, ..., h_k^{s^{q-Q}})$ for each $1 \leqslant k \leqslant t$ from its t-parallel Leak secure challenger. Then, $\mathcal{B}_i$ picks $(h, s_1, ..., s_{i-1}, s_{i+1}, ..., s_n)$ at random and sets $s_i = s$ ($\mathcal{B}_i$ does not know what s is). Given the $\mathsf{Leak}(s, h_1, ..., h_t) = (g, g^s, ..., g^{s^{q-1}}, h_k, ..., h_k^{s^{q-Q}})$ for each $1 \leqslant k \leqslant t$, $\mathcal{B}$ can compute $L_g(s_1, ..., s_n, h)$ from his selection. $\mathcal{B}_i$ simulates $\mathcal{A}$ by sending him $L_g(s_1, ..., s_n, h)$.

When $\mathcal{A}$ queries $(x_1, ..., x_n)$, $\mathcal{B}_i$ computes $\ell = \mathsf{Index}(x_1, ..., x_{i-1})$. If ℓ is not defined, it takes the next available index in $\{1, ..., t\}$ to define it. $\mathcal{B}_i$ queries its t-parallel Leak challenger with (x_i, ℓ) and obtains a $\overline{h}_i \in \mathbb{G}$. Note that $\overline{h}_i$ is either random or is equal to $y = f_{s,h_\ell} = h_\ell^{\frac{1}{s+x_i}}$ for some random key (s, h_ℓ). $\mathcal{B}_i$ finishes the cascade as:

$\overline{h}_i = y$
for $j = i + 1, ..., n$ do
  $\overline{h}_i \leftarrow f_{s_i, \overline{h}_{i-1}}(x_i)$
output $\overline{h}_n$.

Finally $\mathcal{B}_i$ returns $\overline{h}_n$ $\mathcal{A}$. Eventually $\mathcal{A}$ outputs a bit $b \in \{0, 1\}$. $\mathcal{B}_i$ outputs the same b to its challenger.

The Index function together with the random selection of the h_ℓ simulates well a random function on $(x_1, ..., x_{i-1})$. So, p_{i-1} is the probability that $\mathcal{B}_i$ returns $b = 1$ in the game with the real function.

When $\mathcal{B}_i$'s challenger responds with an ideal function, the random selection of the function Index together with the random selection of the h_ℓ makes $(x_1, ..., x_i) \to \overline{h}_i$ simulates well a random function. So, p_{i-1} is the probability that $\mathcal{B}_i$ returns $b = 1$ in the game with the ideal function.

Hence, $|p_n - p_0| = \sum_i \mathsf{Adv}^{\mathsf{Leak}}_{f^t_{s,h_1,...,h_t}}(\mathcal{B}_i)$, which is what we claim. Hence, due to Leak-parallel security, we obtain the result.

□

4.5 Related Key Secure PRF

Let us define the following game using a bit b for an adversary $\mathcal{A}$ playing against a challenger:

-Pick K in $\mathcal{K}$ at random.
-Let $\mathcal{A}$ make queries to $\mathsf{GEN}(\phi, x)$ with $\phi \in \Phi$
-$\mathcal{A}$ outputs b'

$\underline{\text{proc } \mathsf{GEN}(\phi, x)}$
$\mathsf{K}' \leftarrow \phi(\mathsf{K})$;
If $\mathsf{K}' = \perp$ then return $\perp$;
If $\mathsf{T}[\mathsf{K}'] = \perp$ then
    if $\mathsf{b} = 1$ then $\mathsf{T}[\mathsf{K}'] \leftarrow \mathsf{F}(\mathsf{K}, x)$;
    if $\mathsf{b} = 0$ then $\mathsf{T}[\mathsf{K}'] \leftarrow \{0,1\}^r$;
Return $\mathsf{T}[\mathsf{K}']$

For all ppt adversary $\mathcal{A}$, a pseudorandom function F is a RKA secure PRF with respect to a function family Φ if $\mathsf{Adv}_{\mathcal{A},\Phi} = |\mathsf{Pr}(\mathsf{b}' = 1|\mathsf{b} = 1) - \mathsf{Pr}(\mathsf{b}' = 1|\mathsf{b} = 0)|$ is bounded by ϵ.

Example 3. $\mathsf{f}_{s,h}(x) = h^{\frac{1}{x+s}}$ is not RKA secure PRF for $\phi(s, h) = (s + \Delta, h)$.

Let the adversary make two queries to GEN with $((s, h), x)$ and $((s, h), x-\Delta)$. If we are in real world ($\mathsf{b} = 1$), then the outputs are $h^{\frac{1}{x+s}}$ for both queries. Clearly, these two outputs are same if we are in real world. We get two random strings if we are in ideal world ($\mathsf{b} = 0$). It allows us to correctly guess bit b.

5 Conclusion

We define a new security notion called Key Dependent Feedback(KDF) security inspired from circular security of pseudorandom functions introduced in Distance Bounding Protocols. We give an algebraic structure of PRF under KDF security. We prove that a small-domain Verifiable Random Functions(VRF) from Dodis-Yampolskiy is a circular secure PRF which easily extends to efficiently large-domain VRF by augmented cascading by Boneh et. al.

We have constructed a circular-secure PRF function with no random oracle and under (g, q)-DDH assumption. Unfortunately, we proved circular security from Leak security. For this reason, this construction is not well suited to distance bounding. Indeed, the construction of DB protocols using circular-secure PRF rely on the fact that leaking L would leak the entire secret, so, cannot be Leak-secure. Hence, the problem of making a circular-secure PRF which is not Leak-secure is still an open problem.

Acknowledgments.
The first author was supported in part by NSF grant CNS-1453132.
We thank Dr. Reza Reyhanitabar for helpful discussions and valuable comments.

References

1. M. Bellare, R. Canetti, and H. Krawczyk. Pseudorandom functions revisited: the cascade construction and its concrete security. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 514–523, Oct 1996.
2. Dan Boneh, Shai Halevi, Mike Hamburg, and Rafail Ostrovsky. *Advances in Cryptology – CRYPTO 2008: 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, chapter Circular-Secure Encryption from Decision Diffie-Hellman, pages 108–125. Springer Berlin Heidelberg, 2008.
3. Dan Boneh, Hart William Montgomery, and Ananth Raghunathan. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In *Proceedings of the 17th ACM conference on Computer and communications security*, CCS '10, pages 131–140. ACM, 2010.
4. Ioana Boureanu, Aikaterini Mitrokotsa, and Serge Vaudenay. *Progress in Cryptology – LATINCRYPT 2012: 2nd International Conference on Cryptology and Information Security in Latin America, Santiago, Chile, October 7-10, 2012. Proceedings*, chapter On the Pseudorandom Function Assumption in (Secure) Distance-Bounding Protocols, pages 100–120. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
5. Ioana Boureanu, Aikaterini Mitrokotsa, and Serge Vaudenay. *Information Security: 16th International Conference, ISC 2013, Dallas, Texas, November 13-15, 2013, Proceedings*, volume 7807, chapter Practical and Provably Secure Distance-Bounding, pages 248–258. Springer International Publishing, 2015.
6. Ioana Boureanu, Aikaterini Mitrokotsa, and Serge Vaudenay. Practical and provably secure distance-bounding. *Journal of Computer Security*, 23(2):229–257, 2015.
7. Ioana Boureanu and Serge Vaudenay. *Information Security and Cryptology: 10th International Conference, Inscrypt 2014, Beijing, China, December 13-15, 2014, Revised Selected Papers*, volume 8957, chapter Optimal Proximity Proofs, pages 170–190. Springer International Publishing, 2015.
8. Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *Journal of The ACM (JACM)*, 51:557–594, 2004.
9. Yevgeniy Dodis and Aleksandr Yampolskiy. *Public Key Cryptography - PKC 2005: 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets, Switzerland, January 23-26, 2005. Proceedings*, volume 3386, chapter A Verifiable Random Function with Short Proofs and Keys, pages 416–431. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.
10. Oded Goldreich, Shafi Goldwasser, and S Silvio Micali. How to construct random functions. *Journal of The ACM (JACM)*, 33:792–807, 1986.