述語の効率的な回路への埋込による コンパクトな検証可能擬似乱数関数

勝又 秀一[1,2]

概要：検証可能擬似乱数関数（VRF）とは，Micali-Rabin-Vadhan らが FOCS'99 で提案した暗号プリミティブであり，通常の擬似乱数関数の性質に加え，擬似乱数を正しく生成したことを示す証明も出力する．既存の VRF は，(i) 検証鍵または証明のサイズが小さいが，強い安全性仮定を必要とする構成か，(ii) 検証鍵および証明のサイズが大きいが，弱い安全性仮定を基にする構成の二つに大別される．本研究では，安全性証明の際に現われる述語の回路への記述方法に着目し，述語を従来よりも掛け算ゲートの少ない回路で記述する方法を提案する．これにより，検証鍵または証明のサイズが小さく，かつ，弱い安全性仮定を基にした VRF をペアリングから実現する．

キーワード：検証可能擬似乱数関数，述語，算術回路，admissible ハッシュ関数，双線形写像

Constructing Compact Verifiable Random Functions via Efficiently Embedding Predicates as Circuits

Shuichi Katsumata[1,2]

Abstract: In this paper, we construct verifiable random functions (VRFs) [Micali-Rabin-Vadhan, FOCS'99] that have either a small verification key size or proof size under a weaker hardness assumption than previous paring-based VRF schemes. Our main technical contribution is embedding the subset predicate appearing in the security reduction into a shallow arithmetic circuit with few multiplication gates. This technique allows us to use a much weaker hardness assumption and optimize the parameters. Namely, both of our schemes are secure under a non-interactive Q-type assumption where Q is only poly-logarithmic in the security parameter, and they achieve poly-logarithmic verification key size or proof size.

Keywords: Verifiable random functions, predicates, arithmetic circuits, admissible hash function, pairing

1. 序論

背景．検証可能擬似乱数関数（Verifiable Random Function, 以下 VRF）とは, Micali-Rabin-Vadhan らが FOCS'99 で提案した暗号プリミティブであり，通常の擬似乱数関数の機能に加え，擬似乱数を正しく生成したことを示す証明も出力する．そして，公開検証鍵を用いることにより，誰でも擬似乱数の正当性を検証することが可能である．このとき，VRF は，仮に偽の秘密鍵を使っていたとしても，検証が通る証明は正しく計算した擬似乱数のみに限られるという強い一意性の条件が課せられる．

検証可能な性質が擬似乱数関数に加わることにより，取引預託方式 [JS04]，e-cash [ASM07, BCKL09] や非対話宝くじシステム [MR02] など，VRF は検証や認証が必要とされる多くの場で活用することができる．さらに，近年 Goldberg et al. [GNP+15] により，DNS のセキュリティ拡張に対するゾーン列挙攻撃の防止策として，VRF を利用した NSEC5（データの不在証明）が提案された．これ

[1] 東京大学
The University of Tokyo
[2] 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology (AIST)

は，ごく最近 [PWH+17] により効率化され，現在 NSEC5 は IETF の標準化に提出されている [*1]．

VRF は，一意性という強い条件があるため，擬似乱数関数と比べ構成が非常に難しい．例えば，非対話ゼロ知識証明を擬似乱数関数に付け加えるという単純な構成では，非対話ゼロ知識証明が偽の証明を作ることを許容するため，一意性を担保することができない．そのため，理想的な VRF に要求される (i) 入力空間が指数サイズ，(ii) 適応的擬似乱数性を持つ，(iii) 非対話型の困難性仮定への帰着，の全てを満たす方式は，Hohenberger と Waters が 2010 年に提案する VRF [HW10] まで存在しなかった．現在，上記 (i)-(iii) の性質を全て満たす VRF を構成する手法は二つある．一つ目は，ペアリングの代数構造に特化した構成方法 [HW10, BMR10, ACF14, Jag15, HJ16, Yam17] で，二つ目は，汎用的な暗号プリミティブを基にした一般的構成方法 [GHKW17, Bit17, BGJS17] である．後者の手法は VRF がどれだけ強い暗号学的道具なのかを理解するための大切な知見となるが，どの手法にも未だ効率的な構成が知られていない一般言語に対する証拠識別不能性を満たす非対話証明システム（NIWI）と制約付き擬似乱数関数を経由する必要があるため，最終的に得られる VRF は効率的でないことに加え，非常に強い困難性仮定を要する．一方で，前者のペアリングを基にした手法は効率的，もしくは，弱い困難性仮定からの構成が多く知られている．その中でも特筆すべき方式 [Jag15, HJ16, Yam17] を図 1 に記す．

図 1 他の検証可能擬似乱数関数との比較.

方式	検証鍵 $\lvert\mathsf{vk}\rvert$ ($\mathbb{G}$ の要素数)	証明 $\lvert\pi\rvert$ ($\mathbb{G}$ の要素数)	困難性仮定
[Jag15]	$O(\lambda)$	$O(\lambda)$	$O(\log(Q/\epsilon))$-DDH
[HJ16]	$O(\lambda)$	$O(\lambda)$	DLIN
[Yam17]: 7.1. 章	$\omega(\lambda\log\lambda)$	$\omega(\log\lambda)$	$\omega(\lambda\log\lambda)$-DDH
[Yam17]: 7.3. 章	$\omega(\log\lambda)$	$\omega(\sqrt{\lambda}\log\lambda)$	$\omega(\lambda\log\lambda)$-DDH
[Yam17]: 付録 C.	$\omega(\log\lambda)$	$\mathsf{poly}(\lambda)$	$\mathsf{poly}(\lambda)$-DDH
提案方式: 章 3.1.	$\omega(\log^2\lambda)$	$\omega(\lambda\log^2\lambda)$	$\omega(\log^2\lambda)$-DDH
提案方式: 章 3.3.	$\omega(\sqrt{\lambda}\log\lambda)$	$\omega(\log\lambda)$	$\omega(\log^2\lambda)$-DDH

検証鍵と証明の大きさは，それぞれお $\mathbb{G}$ の要素数で数える．Q, ϵ は攻撃者のクエリ回数と成功確率を表す．また，$\omega(f(\lambda))$ は $f(\lambda)$ よりも漸近的に早く成長する任意の関数を表し，$\mathsf{poly}(\lambda)$ は Q, ϵ に依存しない多項式である．

本研究の貢献． 本研究では，検証鍵または証明サイズが小さい VRF を，従来よりも弱い安全性仮定を基にペアリングから二つ構成する（図 1 参照）．従来の方式は，L が poly-logarithm の L-DDH 仮定 [*2] や DLIN 仮定という比較的弱い困難性仮定より安全性が示されるが，検証鍵または証明のサイズが $O(\lambda)$ である方式 [Jag15, HJ16] か，検証鍵または証明のサイズが poly-logarithm であるが，$L = \tilde{\Omega}(\lambda)$ の L-DDH 仮定という比較的強い困難性仮定を要する方式 [Yam17] に分けられる．我々が提案する方式は，両方式の長所を兼ね備えており，特に章 3.3 の提案方式は，検証鍵と証明のサイズ，および，L-DDH 仮定の L のすべてが sub-linear である初めての方式である．

技術的概要． 述語 $P : \mathcal{X} \to \{0,1\}$ とは，入力空間 $\mathcal{X}$ を何かしらの関係によって二つの素集合に分割する関数のことを指す．述語の概念は，属性ベース暗号・述語暗号の構成 [SW05, GPSW06, KSW08] やプログラム可能ハッシュ関数 [HK08, ZCZ16] の構成など，暗号の至るところで暗に現れる．しかし，述語をどう（算術）回路で記述するかに関しては，基本あまり意識が向けられていない．述語は関数であるため，述語の回路の記述方法は決して一意に定まるものではなく，用途に応じてもっとも適した記述方法を考えることが非常に重要になってくる．本研究では，述語の中でもとりわけ汎用的な部分集合述語に着目し，部分集合述語の効率的な回路への埋込方法を提案する．VRF の擬似乱数性の証明の際に現れる admissible ハッシュ関数は，部分集合述語と捉えることができるため，我々の埋込方法を利用することで，より効率的な admissible ハッシュ関数の記述が可能となり，従来の VRF の効率性を高めると同時に，より弱い安全性仮定を基に構成が可能となる．他に，同じアイデアで [Yam17] の ID ベース暗号方式の効率化も可能である．以下，部分集合述語，および，それをどう算術回路に埋込むかの概要を紹介する．

まず，部分集合述語 $P_{\mathsf{T}} : 2^{[2\ell]} \to \{0,1\}$ を $\mathsf{T} \subseteq \mathsf{S}$ のときに限り $P_{\mathsf{T}}(\mathsf{S}) = 1$ となる関数と定義する．ここで，後々 VRF へ部分集合述語を埋込むことを念頭に，簡単のため $|\mathsf{T}| = \eta$，および，P_{T} への入力を要素数が ℓ の集合に限定する．次に，P_{T} をどう記述するかを考える．もっとも自然な記述は部分集合述語を論理回路で表した以下の多項式表現である．

$$\underbrace{\prod_{i=1}^{\eta}\Big(1-\prod_{j=1}^{\ell}\Big(1-\underbrace{\prod_{k=1}^{\zeta}\overbrace{\big(1-(t_{i,k}-s_{j,k})^2\big)}^{t_{i,k}=s_{j,k}?}}_{t_i=s_j?}\Big)\Big)}_{t_i\in\mathsf{S}?} = \begin{cases}1 & \text{if } \mathsf{T}\subseteq\mathsf{S}\\ 0 & \text{if } \mathsf{T}\not\subseteq\mathsf{S}\end{cases} \tag{1}$$

ただし，$\zeta = \lfloor \log 2\ell \rfloor + 1$，$\mathsf{T} = \{t_i\}_{i\in[\eta]}$，$\mathsf{S} = \{s_j\}_{j\in[\ell]} \subseteq [2\ell]$，$t_{i,k}, s_{j,k}$ は各々 t_i, s_j の二進数表記したときの k ビット目の値を表すとする．ここで，式 (1) は，$\wedge_{i\in[\eta]} \vee_{j\in[\ell]} \wedge_{k\in[\zeta]} (t_{i,k} = s_{j,k})$ で表されるブール論理式の多項式表現であることに注意されたい．このとき，上記多項式の次数は $2mn\zeta$ である．従って，VRF の擬似乱数性の証明の際に出てくる admissible ハッシュ関数を式 (1) で記述しようとすると，$\eta = O(\log^2\lambda)$，$\ell = O(\lambda)$，$\zeta = O(\log\lambda)$ となるた

[*1] `https://datatracker.ietf.org/doc/draft-goldbe-vrf/` を参照.

[*2] L-DDH 問題とは，$(h, g, g^{\alpha}, \cdots, g^{\alpha^L}, \Psi)$ を与えられたときに，$\Psi = e(g,h)^{1/\alpha}$ か $\mathbb{G}_T$ 上一様ランダムな元かを識別する問題である.

め，多項式の次数は $O(\lambda \log^3 \lambda)$ になる．そして，擬似乱数性を証明する上で必要となる L-DDH 仮定の L の大きさは，多項式の次数と密接に関係しているため，式 (1) の埋込を利用すると $L = O(\lambda \log^3 \lambda)$ の L-DDH 仮定が必要となる．このとき，L が大きいほど困難性仮定しては強くなるため，如何に L を小さくできるかが課題となる．

ここで，部分集合述語の特性に注目すると，$t_i = s_j$ を満たす $j \in [\ell]$ は高々一つであることが分かる．従って，この性質を考慮にいれて式 (1) を書き直すと，次の多項式が得られる．

$$\prod_{i=1}^{\eta} \sum_{j=1}^{\ell} \prod_{k=1}^{\zeta} \left(1 - (t_{i,k} - s_{j,k})^2\right) = \begin{cases} 1 & \text{if} \quad \mathsf{T} \subseteq \mathsf{S} \\ 0 & \text{if} \quad \mathsf{T} \not\subseteq \mathsf{S} \end{cases} \qquad (2)$$

和の形が現れるため，この多項式を計算する際には論理回路ではなく，算術回路を用いることに注意されたい．式 (2) の多項式の次数は $2m\zeta$ であり，式 (1) のものと比べると次数が ℓ 倍削減されている．もし admissible ハッシュ関数を式 (2) で記述する場合，$\ell = O(\lambda)$ であるので，次数は $O(\log^3 \lambda)$ まで下がる．従って，擬似乱数性を証明するために必要となる L-DDH 問題も L が poly-logarithm という比較的弱い困難性仮定ですむ．最後に，$t_{i,k}, s_{j,k} \in \{0,1\}$ であることに着目すると，要素数が 2 よりも大きい有限体上では

$$(t_{i,k} - s_{j,k})^2 = (1 - s_{j,k}) + (-1 + 2s_{j,k}) \cdot t_{i,k} \qquad (3)$$

と式変形できるため，$s_{j,k}$ と $t_{i,k}$ それぞれに関して線形にすることができる．以上の多項式を算術回路に埋込み，VRF で用いるペアリングの代数構造をうまく利用することで，提案手法が得られる．最後に，式 (2) で得られる足し算の項とペアリングの線型性を利用することで，擬似乱数を正しく生成したかの検証に必要な検証鍵のサイズを $\omega(\lambda \log^2 \lambda)$ から $\omega(\sqrt{\lambda} \log \lambda)$ に大幅に削減可能である（章 3.3 の提案方式参照）．

2. 準備

2.1 検証可能擬似乱数関数

検証可能擬似乱数関数（Verifiable Random Functions, 以下 VRF）は，三つ組の確率的多項式時間アルゴリズム $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Verify})$ によって定義される [MRV99].

- $\mathsf{Gen}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$: 鍵生成アルゴリズムは，セキュリティ・パラメータ 1^λ を入力にとり，検証鍵 vk と秘密鍵 sk を出力する．
- $\mathsf{Eval}(\mathsf{sk}, X) \to (Y, \pi)$: 実行アルゴリズムは，秘密鍵 sk と $X \in \{0,1\}^n$ を入力にとり，$Y \in \mathcal{Y}$ と証明 π を出力する．ただし，$\mathcal{Y}$ はある有限集合とする．
- $\mathsf{Verify}(\mathsf{vk}, X, (Y, \pi)) \to 0/1$: 検証アルゴリズムは，検証鍵 vk，$X \in \{0,1\}^n$，$Y \in \mathcal{Y}$，証明 π を入力にとり，0 または 1 を出力する．

定義 1. 三つ組の確率的多項式時間アルゴリズム $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Verify})$ が以下の条件を満たすとき，VRF を検証可能擬似乱数関数と呼ぶ．ただし，入力長は $n = O(\lambda)$ の場合のみを考える．

正当性. 全ての $\lambda \in \mathbb{N}$, $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$, $X \in \{0,1\}^n$ に対して，もし $(Y, \pi) \leftarrow \mathsf{Eval}(\mathsf{sk}, X)$ であるならば，$\mathsf{Verify}(\mathsf{vk}, X, (Y, \pi)) = 1$ である．

一意性. 任意のビット列 $\mathsf{vk} \in \{0,1\}^*$ と $X \in \{0,1\}^n$ に対して，正しく検証を通る証明 π に対応する $Y \in \mathcal{Y}$ は高々一つである．ここで，vk は Gen によって生成された検証鍵とは限らないとする．

擬似乱数性. チャレンジャーと攻撃者 $\mathcal{A}$ の間で行われる以下のゲームを用いて定義する．

Setup. チャレンジャーは $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ を実行し，攻撃者 $\mathcal{A}$ に検証鍵 vk を送る．

Phase 1. 攻撃者 $\mathcal{A}$ は，チャレンジャーに実行クエリを行う．$\mathcal{A}$ は，$X \in \{0,1\}^n$ をチャレンジャーにクエリし，チャレンジャーは $(Y, \pi) \leftarrow \mathsf{Eval}(\mathsf{sk}, X)$ を $\mathcal{A}$ に返す．

Challenge Query. 攻撃者 $\mathcal{A}$ は，チャレンジ入力 $X^* \in \{0,1\}^n$ をチャレンジャーに渡す．ただし，X^* は Phase 1 で $\mathcal{A}$ によってクエリされていないものに限る．チャレンジャーは，ランダムに $\mathsf{coin} \leftarrow \{0,1\}$ を選ぶ．さらに，$(Y_0^*, \pi_0^*) \leftarrow \mathsf{Eval}(\mathsf{sk}, X^*)$ を実行し，$Y_1^* \leftarrow \mathcal{Y}$ を選ぶ．最後に，チャレンジ実行値 Y_{coin}^* を $\mathcal{A}$ に返す．

Phase 2. 攻撃者 $\mathcal{A}$ は，$X \neq X^*$ に対して，Phase 1 と同様に実行クエリを行える．

Guess. 攻撃者 $\mathcal{A}$ は coin の推定値 $\widehat{\mathsf{coin}}$ を出力する．

攻撃者 $\mathcal{A}$ の優位性を $|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}|$ で定義する．このとき，VRF が（適応的）擬似乱数性を持つとは，任意の確率的多項式アルゴリズム $\mathcal{A}$ の優位性が無視可能なことを意味する．

2.2 双線形群生成機

[HJ16] によって定式化された認証付き双線形群生成機を導入する．これは，VRF の一意性を証明するために必要な道具である．まず，セキュリティ・パラメータ 1^λ を入力に，位数 p の群 $\mathbb{G}, \mathbb{G}_T$ と写像 $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ の記述 Π を出力する効率的なアルゴリズム GrpGen を用意する．そして，記述 Π および各群の元の記述を入力に，その元の正当性を出力する効率的なアルゴリズム GrpVfy を用意する．加えて，効率的に検証が可能なように，各群の元が一意にエンコーディングされているとする．詳細は [HJ16] を参照されたい．

困難性仮定.

定義 2 (L-Diffie-Hellman 仮定). 確率的多項式アルゴリズム $\mathcal{A}$ の L-Diffie-Hellman 問題を解く優位性を次のように

GrpGen を用いて定義する．

$$\mathsf{Adv}^{L\text{-}DDH}_{\mathcal{A}} = |\Pr[\mathcal{A}(\Pi, g, h, g^{\alpha}, g^{\alpha^2}, \cdots, g^{\alpha^L}, \Psi_0) \to 1] - \Pr[\mathcal{A}(\Pi, g, h, g^{\alpha}, g^{\alpha^2}, \cdots, g^{\alpha^L}, \Psi_1) \to 1]|,$$

ただし，$\Pi \leftarrow \mathsf{GrpGen}(1^\lambda), \alpha \leftarrow \mathbb{Z}_p^*, g, h \leftarrow \mathbb{G}, \Psi_0 = e(g,h)^{1/\alpha}$, $\Psi_1 \leftarrow \mathbb{G}_T$ とする．このとき，任意の確率的多項式アルゴリズム $\mathcal{A}$ に対して $\mathsf{Adv}^{L\text{-}DDH}_{\mathcal{A}}$ が無視可能のとき，*L-DDH* 仮定が成り立つという．

2.3 安全性証明に用いるハッシュ関数

VRF の安全性をより簡潔に証明するために，[Yam17] で提案された分割関数の概念を導入する．これは，暗号方式の適応的安全性を示すときによく用いられる admissible ハッシュ関数 [BB04, CHKP10, FHPS13, Jag15] の拡張として捉えられる．

定義 3 (分割関数)**.** $\mathsf{F} = \{\mathsf{F}_\lambda : \mathcal{K}_\lambda \times \mathcal{X}_\lambda \to \{0,1\}\}_{\lambda\in\mathbb{N}}$ を関数族とする．このとき，F が分割関数であるとは，$Q : \mathbb{N} \to \mathbb{N}$ で定義される多項式関数 $Q = Q(\lambda)$ と $\epsilon : \mathbb{N} \to (0, 1/2]$ で定義される noticeable な関数 $\epsilon = \epsilon(\lambda)$ を入力にとり，下記条件を満たす分割鍵 K を出力する確率的多項式アルゴリズム $\mathsf{PrtSmp}(1^\lambda, Q(\lambda), \epsilon(\lambda))$ が存在することを意味する．

(1) ある $\lambda_0 \in \mathbb{N}$ に対して，任意の $\lambda > \lambda_0$ で $\Pr\left[K \in \mathcal{K}_\lambda : K \leftarrow \mathsf{PrtSmp}\left(1^\lambda, Q(\lambda), \epsilon(\lambda)\right)\right] = 1$ が成り立つ．ただし，λ_0 は Q と ϵ に依存してよい．

(2) $\lambda > \lambda_0$ に対して，関数 Q, ϵ に依存した関数 $\gamma_{\max}(\lambda)$ と $\gamma_{\min}(\lambda)$ 存在して，任意の $X^{(1)}, \cdots, X^{(Q(\lambda))}, X^* \in \mathcal{X}_\lambda$ と $X^* \notin \{X^{(1)}, \cdots, X^{(Q(\lambda))}\}$ に関して，

$$\gamma_{\min}(\lambda) \le \Pr\Big[\bigwedge_{i=1}^{Q(\lambda)} \mathsf{F}(K, X^{(i)}) = 1 \wedge \mathsf{F}(K, X^*) = 0\Big] \le \gamma_{\max}(\lambda) \quad (4)$$

が成り立ち，以下で定義した関数 $\tau(\lambda)$ が noticeable である．

$$\tau(\lambda) := \gamma_{\min}(\lambda) \cdot \epsilon(\lambda) - \frac{\gamma_{\max}(\lambda) - \gamma_{\min}(\lambda)}{2} \quad (5)$$

ここで，確率空間は $K \leftarrow \mathsf{PrtSmp}(1^\lambda, Q(\lambda), \epsilon(\lambda))$ で用いられる乱数のよって定められる．

本論文では，modified admissible ハッシュ関数と呼ばれる分割関数を用いる．以下は [Jag15, Yam17] の結果に従う．

定義 4. (Modified Admissible ハッシュ関数) $n = n(\lambda), \ell = \ell(\lambda), \eta = \eta(\lambda)$ を $n, \ell = \Theta(\lambda), \eta = \omega(\log\lambda)$ を満たす関数とする．また，$\{C_n : \{0,1\}^n \to \{0,1\}^\ell\}_{n\in\mathbb{N}}$ をある定数 $c \in (0, 1/2)$ に対して最小距離が $c \cdot \ell$ となる誤り訂正符号の族とする．そして，

$$\mathcal{K}_{\mathsf{MAH}} = \{\mathsf{T} \subseteq [2\ell] \mid |\mathsf{T}| < \eta\}, \qquad \mathcal{X}_{\mathsf{MAH}} = \{0,1\}^n.$$

とおく．このとき，*modified admissible* ハッシュ関数 $\mathsf{F}_{\mathsf{MAH}} : \mathcal{K}_{\mathsf{MAH}} \times \mathcal{X}_{\mathsf{MAH}} \to \{0,1\}$ を以下のように定義する．

$$\mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X) = \begin{cases} 0, & \mathsf{T} \subseteq \mathsf{S}(X) \text{ の場合} \\ 1, & \text{それ以外の場合} \end{cases} \quad (6)$$

ただし，$\mathsf{S}(X) = \{2i - C(X)_i \mid i \in [\ell]\}$ で，$C(X)_i$ は $C(X) \in \{0,1\}^\ell$ の i 番目のビットとする．

定理 5. $\mathsf{F} := \mathsf{F}_{\mathsf{MAH}}$ としたとき，分割関数の定義を満たす確率的多項式アルゴリズム $\mathsf{PrtSmp} := \mathsf{PrtSmp}_{\mathsf{MAH}}$ が存在する．このとき，出力される分割鍵 T の要素数は $\eta'(\lambda) = \left\lfloor \frac{\log(2Q + Q/\epsilon)}{-\log(1-c)} \right\rfloor$ で，$\tau(\lambda)$ は $2^{-\eta'-1} \cdot \epsilon$ である．

3. VRF の構成

この章では，二つの VRF の構成を提案する．一つ目は検証鍵サイズが小さいで VRF で，二つ目は証明サイズが小さい VRF である．

3.1 検証鍵サイズが小さい VRF

以下，$n, \ell, \eta, \mathsf{S}(\cdot)$ は modified admissible ハッシュ関数で用いるパラメータとする．また，$\zeta = \lfloor \log p \rfloor + 1$ とおく．ここで，$n = \ell = \Theta(\lambda), \eta = \omega(\log\lambda), \zeta = O(\log\lambda)$ であることに注意されたい．

$\mathsf{Gen}(1^\lambda)$: 1^λ を入力に，$\Pi \leftarrow \mathsf{GrpGen}(1^\lambda)$ を実行し群の記述を得る．また，一様ランダムに $g, h \leftarrow \mathbb{G}^*$ を選び，$(i,k) \in [\eta]\times[\zeta]$ に対して一様ランダムに $w_0, w_{i,k} \leftarrow \mathbb{Z}_p$ を選ぶ．最後に，下記の検証鍵と秘密鍵を出力する．

$$\mathsf{vk} = \left(\Pi, g, h, g_0 := g^{w_0}, \left(g_{i,k} := g^{w_{i,k}}\right)_{(i,k)\in[\eta]\times[\zeta]}\right),$$
$$\mathsf{sk} = \left(w_0, (w_{i,k})_{(i,k)\in[\eta]\times[\zeta]}\right).$$

$\mathsf{Eval}(\mathsf{sk}, X)$: $X \in \{0,1\}^n$ を入力に，まず $\mathsf{S}(X) = \{s_1, \cdots, s_\ell\} \subset [2\ell]$ を計算する．以下，s_j を二進数表記したときの k ビット目の値を $s_{j,k}$ とする．ただし，$k \in [\zeta]$ である．次に，$i' \in [\eta]$ と $(i,j,k') \in [\eta]\times[\ell]\times[\zeta]$ に関して，以下を計算する．

$$\theta_{i'} = \prod_{i=1}^{i'} \sum_{j=1}^{\ell} \prod_{k=1}^{\zeta} \Big((1 - s_{j,k}) + (-1 + 2s_{j,k}) \cdot w_{i,k}\Big),$$
$$\theta_{i,j,k'} = \prod_{k=1}^{k'} \Big((1 - s_{j,k}) + (-1 + 2s_{j,k}) \cdot w_{i,k}\Big).$$

ここで，$\theta := \theta_\eta$ と置く．そして，以下を出力する．

$$Y = e(g,h)^{\theta/w_0},$$
$$\pi = \left(\pi_0 := g^{\theta/w_0}, \left(\pi_{i'} := g^{\theta_{i'}}\right)_{i'\in[\eta]}, \left(\pi_{i,j,k'} := g^{\theta_{i,j,k'}}\right)_{(i,j,k')\in[\eta]\times[\ell]\times[\zeta]}\right).$$

$\mathsf{Verify}(\mathsf{vk}, X, (Y, \pi))$: 最初に検証鍵 vk の正当性を確認す

る．もし，以下の条件のいずれか一つでも成立しなければ，0 を出力する．

(1) vk が $\left(\Pi, g, h, g_0, \left(g_{i,k}\right)_{(i,k)\in[\eta]\times[\zeta]}\right)$ の形である．

(2) 全ての $s \in (g, h, g_0) \cup (g_{i,k})_{(i,k)\in[\eta]\times[\zeta]}$ に関して，$\mathsf{GrpVfy}(\Pi) = 1$ と $\mathsf{GrpVfy}(\Pi, s) = 1$ である．

次に，X, Y, π の正当性を確認する．それにあたり，まず全ての $i' \in [\eta], (i, j, k') \in [\eta] \times [\ell] \times [\zeta]$ に対して，下記 $\Phi_{i'}, \bar{g}_{i,j,k'}$ を用意する．

$$\Phi_{i'} := \prod_{j=1}^{\ell} \pi_{i',j,\zeta}, \quad \bar{g}_{i,j,k'} := g^{1-s_{j,k'}} \cdot (g_{i,k'})^{-1+2s_{j,k'}}.$$

もし，以下の条件のいずれかが成立しなければ，0 を出力する．

(3) $X \in \{0,1\}^n$ と $Y \in \mathbb{G}_T$ であり，証明 π が $\left(\pi_0, \left(\pi_{i'}\right)_{i'\in[\eta]}, \left(\pi_{i,j,k'}\right)_{(i,j,k')\in[\eta]\times[\ell]\times[\zeta]}\right)$ の形である．

(4) 全ての $i' \in [\eta - 1]$ と $(i, j, k') \in [\eta] \times [\ell] \times [\zeta - 1]$ に関して以下が成り立つ．

$$e(\pi_1, g) = e(\Phi_1, g), \qquad e(\pi_{i,j,1}, g) = e(\bar{g}_{i,j,1}, g),$$
$$e(\pi_{i'+1}, g) = e(\Phi_{i'+1}, \pi_{i'}), \quad e(\pi_{i,j,k'+1}, g) = e(\bar{g}_{i,j,k'+1}, \pi_{i,j,k'}).$$

(5) $e(\pi_\eta, g) = e(\pi_0, g_0)$ と $e(\pi_0, h) = Y$ が成り立つ．

上記の条件が全て満たされているならば，1 を出力する．

3.2 正当性，一意性，擬似乱数性

正当性と一意性は，認証付き双線形群生成機の定義より直ちに導かれる．紙面上の都合により本稿では証明を割愛する．以下，VRF の満たすべき要件の中で最も証明が複雑な擬似乱数性の証明を与える．

定理 6 (擬似乱数性)．L-DDH 問題の困難性を仮定すると，上記の VRF は擬似乱数性を持つ．ただし，$L = \eta\zeta = \omega(\log^2 \lambda)$ である．

証明. 上記 VRF の擬似乱数性を無視可能でない優位性で破る攻撃者 $\mathcal{A}$ の存在を仮定する．このとき，$\mathcal{A}$ の優位性を $\epsilon = \epsilon(\lambda)$，$\mathcal{A}$ が行う実行クエリの回数の上限を $Q = Q(\lambda)$ とおく．ここで Q は多項式とする．また，定義より，ある noticeable な関数 $\epsilon_0 = \epsilon_0(\lambda)$ が存在して，無限個の $\lambda \in \mathbb{N}$ に対して $\epsilon(\lambda) \geq \epsilon_0(\lambda)$ が成り立つ．従って，定義 3 と定理 5 より，$\mathsf{T} \leftarrow \mathsf{PrtSmp}_{\mathsf{MAH}}(1^\lambda, Q(\lambda), \epsilon_0(\lambda))$ に対して，十分大きな λ で $\mathsf{T} \subseteq [2\ell]$ と $|\mathsf{T}| < \eta$ が確率 1 で成り立つ．以後，この前提を基に証明を進める．証明は，擬似乱数性に対する攻撃者とチャレンジャーのゲーム列を用いて行う．このとき，各ゲームで $\mathsf{coin}' \in \{0,1\}$ という変数を定義し，最初のゲーム Game_1 に関しては $\mathsf{coin}' = \widehat{\mathsf{coin}}$ と定義する．そして，E_i を Game_i で $\mathsf{coin}' = \mathsf{coin}$ となる事象とする．

Game_0： これは実際に行われる安全性ゲームである．$\mathcal{Y} = \mathbb{G}_T$ であるため，$\mathsf{coin} = 1$ のときは，ランダムな元 $Y_1^* \leftarrow \mathbb{G}_T$ を $\mathcal{A}$ に渡す．$\mathcal{A}$ はゲームの終わりに coin の予想 $\widehat{\mathsf{coin}}$ を出力し，チャレンジャーは $\mathsf{coin}' = \widehat{\mathsf{coin}}$ とおく．このとき，攻撃者 $\mathcal{A}$ の仮定より，$\left|\Pr[\mathsf{E}_0] - \frac{1}{2}\right| = \left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| = \left|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}\right| = \epsilon$ を得る．

Game_1： このゲームでは，チャレンジャーがゲームの最後で追加の動作を行うように Game_0 を変更する．まず，チャレンジャーは定理 5 の $\mathsf{T} \leftarrow \mathsf{PrtSmp}_{\mathsf{MAH}}(1^\lambda, Q(\lambda), \epsilon_0(\lambda))$ を実行する．ここで，前述したように，$|\mathsf{T}| \subseteq [2\ell]$ と $|\mathsf{T}| < \eta$ が成り立つ．そして，以下の条件が成り立つかを確かめる．

$$\bigwedge_{i=1}^{Q} (\mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X^{(i)}) = 1) \ \wedge \ \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X^*) = 0$$
$$\Leftrightarrow \quad \bigwedge_{i=1}^{Q} \left(\mathsf{T} \not\subseteq \mathsf{S}(X^{(i)})\right) \ \wedge \ \left(\mathsf{T} \subseteq \mathsf{S}(X^*)\right). \qquad (7)$$

ただし，X^* はチャレンジ入力で $\{X^{(i)}\}_{i\in[Q]}$ は $\mathcal{A}$ が行なう実行クエリとする．もしこの条件が成立しない場合，チャレンジャーは $\mathcal{A}$ の出力 $\widehat{\mathsf{coin}}$ を無視し，$\mathsf{coin}' \leftarrow \{0,1\}$ と定める．この動作をチャレンジャーが abort すると呼ぶ．もし条件 (7) が成り立つ場合，チャレンジャーは $\mathsf{coin}' = \widehat{\mathsf{coin}}$ と置く．すると，定義 3 と定理 5 より，無限個の λ に関して下記が成り立つ．

$$\left|\Pr[\mathsf{E}_1] - \frac{1}{2}\right| \geq \gamma_{\min} \cdot \epsilon - \frac{\gamma_{\max} - \gamma_{\min}}{2}$$
$$\geq \gamma_{\min} \cdot \epsilon_0 - \frac{\gamma_{\max} - \gamma_{\min}}{2} \geq \tau,$$

ここで，$\tau = \tau(\lambda)$ は noticeable な関数で，$\gamma_{\max}, \gamma_{\min}, \tau$ は Q, ϵ_0 と分割関数 $\mathsf{F}_{\mathsf{MAH}}$ によって定義された関数である．

Game_2： このゲームでは $w_0, (w_{i,k})_{(i,k)\in[\eta]\times[\zeta]}$ の選び方を変える．まず，ゲームの始めにチャレンジャーは $\mathsf{T} \leftarrow \mathsf{PrtSmp}_{\mathsf{MAH}}(1^\lambda, Q(\lambda), \epsilon_0(\lambda))$ を実行し，$\mathsf{T} = \{t_1, \cdots, t_{\eta'}\} \subset [2\ell]$ とおく．ここで，チャレンジャーがアルゴリズム PrtSmp を実行するタイミングは攻撃者に一切の影響を与えないことに注意されたい．また，仮定より $\eta' < \eta$ であるため，$i \in [\eta' + 1, \eta]$ に関して $t_i = 0$ とおく．そして，チャレンジャーは $\alpha \leftarrow \mathbb{Z}_p^*$ および $(i,k) \in [\eta] \times [\zeta]$ に対して $\tilde{w}_0, \tilde{w}_{i,k} \leftarrow \mathbb{Z}_p$ をサンプルし，下記のように各値を設定する．

$$w_0 = \tilde{w}_0 \cdot \alpha, \quad w_{i,k} = \tilde{w}_{i,k} \cdot \alpha + t_{i,k} \qquad (8)$$

ただし，$t_{i,k}$ は t_i をビット分解したときの k 番目の値とする．他の箇所は全て Game_1 を同じである．ここ

で，Game_1 と Game_2 で選ばれる $w_0, (w_{i,k})_{(i,k)\in[\eta]\times[\zeta]}$ の分布の統計距離は高々 $(\eta\zeta+1)/p$ であるため，$|\Pr[\mathsf{E}_1]-\Pr[\mathsf{E}_2]| = \mathsf{negl}(\lambda)$. である．

次のゲーム Game_3 に移行する前に，分割関数 $\mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X)$ を埋め込む多項式を定義する．この多項式は各入力 X に対して定義され，$\mathsf{T} \subseteq \mathsf{S}(X)$ であるか否かに従って特殊な構造を持つ．任意の $|\mathsf{T}| = \eta' < \eta$ となる $\mathsf{T} \subseteq [2\ell]$ と $X \in \{0,1\}^n$（すなわち，任意の $\mathsf{S}(X)$）に対して，多項式 $\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) : \mathbb{Z}_p \to \mathbb{Z}_p$ を次のように定義する．

$$\begin{aligned}&\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z})\\ =&\prod_{i=1}^{\eta}\sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1-s_{j,k})+(-1+2s_{j,k})\cdot(\tilde{w}_{i,k}\mathsf{Z}+t_{i,k})\Big),\end{aligned} \tag{9}$$

ここで，$\{s_{j,k}\}_{(j,k)\in[\ell]\times[\zeta]}$ と $\{t_{i,k}\}_{(i,k)\in[\eta]\times[\zeta]}$ は Game_2 と同様に定義する． このとき，$\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha) = \theta$ である．安全性を証明するために次の補題を用意する．

補題 7. すべての $X \in \{0,1\}^n$ に対して，

$$\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) = \begin{cases} 1+\mathsf{Z}\cdot\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}), & \text{if } \mathsf{F}_{\mathsf{MAH}}(\mathsf{T},X)=0\\ \mathsf{Z}\cdot\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}), & \text{if } \mathsf{F}_{\mathsf{MAH}}(\mathsf{T},X)=1\end{cases}.$$

となるような多項式 $\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) : \mathbb{Z}_p \to \mathbb{Z}_p$ が存在する．すなわち，$\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z})$ は $\mathsf{T} \subseteq \mathsf{S}(X)$ のときに限り Z で割り切れる．

定理 6 の証明を中断しないために，この補題の証明は定理 6 の最後に与える．次に，すべての $X \in \{0,1\}^n$, $i' \in [\eta]$, $(i,j,k') \in [\eta]\times[\ell]\times[\zeta]$ に対して，$\mathbb{Z}_p$ 上の多項式を考える．

$$\begin{cases} \theta^X_{i'}(\mathsf{Z}) = \prod_{i=1}^{i'}\sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1-s_{j,k})+(-1+2s_{j,k})(\tilde{w}_{i,k}\mathsf{Z}+t_{i,k})\Big)\\ \theta^X_{i,j,k'}(\mathsf{Z}) = \prod_{k=1}^{k'}\Big((1-s_{j,k})+(-1+2s_{j,k})(\tilde{w}_{i,k}\mathsf{Z}+t_{i,k})\Big)\end{cases}$$

ここで，$\theta^X(\mathsf{Z}) := \theta^X_\eta(\mathsf{Z})$ と定義する．このとき，$\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) = \theta^X(\mathsf{Z})$, $\theta_{i'} = \theta^X_{i'}(\alpha), \theta_{i,j,k'} = \theta^X_{i,j,k'}(\alpha)$, $\theta = \theta^X(\alpha)$ である．

Game_3： 前のゲームでは，条件 (7) が満たされないとき，チャレンジャーはゲームの最後で abort した．Game_3 では，条件 (7) が満たされない時点で abort するようにゲームを変更する．これは，攻撃者の成功確率に一切の影響を与えないため，$\Pr[\mathsf{E}_2] = \Pr[\mathsf{E}_3]$ である．

Game_4： このゲームでは，チャレンジャーの実行クエリに対する応答を変える．攻撃者 $\mathcal{A}$ が入力 X に対する実行クエリを行った場合，チャレンジャーはまず条件 (7)（すなわち，$\mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X) = 1$）が成り立つかを確認する．もし成り立たなければ，$\mathsf{Game}_3$ と同様にゲームを abort する．成り立つ場合は，$\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) = \mathsf{Z}\cdot\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z})$ となる多項式 $\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ を計算し，次を $\mathcal{A}$ に渡す．

$$Y = e(g^{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha)/\tilde{w}_0}, h), \tag{10}$$

$$\pi = \Big(\pi_0 = g^{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha)/\tilde{w}_0}, \Big(\pi_{i'} = g^{\theta^X_{i'}(\alpha)}\Big)_{i'\in[\eta]}, \Big(\pi_{i,j,k'} = g^{\theta^X_{i,j,k'}(\alpha)}\Big)_{(i,j,k')\in[\eta]\times[\ell]\times[\zeta]}\Big). \tag{11}$$

ここで，$\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z})$ の存在は補題 7 によって保証されている．ここで，$\theta^X_{i'}(\mathsf{Z})$ と $\theta^X_{i,j,k'}(\mathsf{Z})$ の定義より，$\pi_{i'}$ と $\pi_{i,j,k'}$ が正しく分布していることは直ちに確認できる．さらに，次が成り立つ．

$$\frac{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha)}{\tilde{w}_0} = \frac{\alpha\cdot\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha)}{\alpha\cdot\tilde{w}_0} = \frac{\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha)}{w_0} = \frac{\theta}{w_0}.$$

以上より，Y と π_0 も Game_3 と同様に分布している．従って，チャレンジャーは実行クエリを完璧にシミュレートできているため，$\Pr[\mathsf{E}_3] = \Pr[\mathsf{E}_4]$ である．

Game_5： このゲームでは，$\mathsf{coin} = 0$ におけるチャレンジ実行値の生成の仕方を変える．前のゲームでは，$\mathsf{coin} = 0$ のとき，現実のゲームと同様にチャレンジ実行値を $Y^*_0 = \mathsf{Eval}(\mathsf{sk}, X^*)$ と生成した．このゲームでは，$\mathsf{coin} = 0$ および $\mathsf{F}_{\mathsf{MAH}}(X^*) = 0$（すなわち，ゲームを abort しない）とき，まずチャレンジャーは $\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\mathsf{Z}) = 1 + \mathsf{Z}\cdot\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\mathsf{Z})$ となるような多項式 $\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{X}]$ を計算する．ここで，このような多項式の存在は補題 7 によって保証されている．そして，Y^*_0 を次のようにおき，$\mathcal{A}$ に渡す．

$$Y^*_0 = \Big(e(g,h)^{1/\alpha}\cdot e(g,h)^{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\alpha)}\Big)^{1/\tilde{w}_0}$$

このとき，Y^*_0 は次のように変形することができる．

$$\begin{aligned}&\Big(e(g,h)^{1/\alpha}\cdot e(g,h)^{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\alpha)}\Big)^{1/\tilde{w}_0}\\ =&e(g^{(1+\alpha\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\alpha))/\alpha\tilde{w}_0}, h)\\ =&e(g^{\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\alpha)/w_0}, h)\\ =&e(g^{\theta/w_0}, h).\end{aligned}$$

従って，攻撃者の view は前のゲームと同じである．すなわち，$\Pr[\mathsf{E}_4] = \Pr[\mathsf{E}_5]$ である．

Game_6： このゲームでは，coin の出力に関係なくチャレンジ暗号文を $\mathbb{G}_T$ のランダムな値とする．つまり，$Y^* \leftarrow \mathbb{G}_T$ とする．$|\Pr[\mathsf{E}_5] = \Pr[\mathsf{E}_6]| = \mathsf{negl}(\lambda)$ であることは，$L = \eta\zeta$ の *L-DDH* 問題の困難性を仮定することによって導かれる．紙面上の都合により，証明は割愛する．

分析. 以上の議論より，$|\Pr[\mathsf{E}_6] - 1/2| = |\Pr[\mathsf{E}_1] - 1/2 + \sum_{i=1}^{5}(\Pr[\mathsf{E}_{i+1}] - \Pr[\mathsf{E}_i])| \geq |\Pr[\mathsf{E}_1] - 1/2| - \sum_{i=1}^{5}|\Pr[\mathsf{E}_{i+1}] - \Pr[\mathsf{E}_i]| \geq \tau(\lambda) - \mathsf{negl}(\lambda)$, が無限個の λ で成り立つ．しかし，$\Pr[\mathsf{E}_6] = 1/2$ より，無限個の λ に対して $\tau(\lambda) \leq \mathsf{negl}(\lambda)$ が成り立つことになるため，攻撃者 $\mathcal{A}$ の存在性の仮定に矛盾する． □

最後に，補題 7 を証明する．

補題 7 の証明. まず，次のように式 (9) を式変形する．

$$\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(\mathsf{X})}(\mathsf{Z}) = \mathsf{Z}\cdot\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(\mathsf{X})}(\mathsf{Z}) + \underbrace{\prod_{i=1}^{\eta}\sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1-s_{j,k})+(-1+2s_{j,k})\cdot t_{i,k}\Big)}_{=C}$$

ただし，$\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(\mathsf{X})}(\mathsf{Z})$ は高々次数が $\eta\zeta$ の多項式である．このとき，定数項 C は，技術的概要で述べた部分集合述語 $P_{\mathsf{T}}(\mathsf{S})$ を計算しているに他ならない．従って，式 (2) と (3) より，$\mathsf{T}\subseteq\mathsf{S} \Leftrightarrow \mathsf{F}_{\mathsf{MAH}}(\mathsf{T},X) = 0$ のとき $C = 1$ で，それ以外のときは $C = 0$ となる．以上より，補題 7 は示された． □

3.3 証明サイズが小さい VRF

この章では，前章よりも証明サイズが小さい VRF の構成を提案する．具体的には，検証鍵に補助項を加えることにより，証明サイズが $|\pi| = \omega(\log\lambda)$ で検証鍵サイズが $|\mathsf{vk}| = \omega(\sqrt{\lambda}\log\lambda)$ の方式を得る．ここで，前章での VRF は $|\pi| = \omega(\lambda\log^2\lambda)$，$|\mathsf{vk}| = \omega(\log^2\lambda)$ であった．

準備. 方式を説明する前に，冪組を定義する．冪組 $\mathcal{P}(W)$ とは組 W（順序づけられた集合）に対して定義され，冪集合と似た概念である．より正確には，冪組 $\mathcal{P}(W)$ とは W の部分列を辞書式順序に並べた組のことである．例えば，$W = (w_1, w_2, w_3)$ という三つ組に対しては，$\mathcal{P}(W) = (w_1, w_2, w_3, w_1w_2, w_1w_3, w_2w_3, w_1w_2w_3)$ である．ただし，空の文字列を W の部分列とは考えないとする．また，$\mathbb{G}$ または $\mathbb{G}_T$ の元 g と要素が $\mathbb{Z}_p$ の組 W に対して，$g^{\mathcal{P}(W)}$ を $(g^w \mid w \in \mathcal{P}(W))$ で定義される組とする．さらに，要素が $\mathbb{Z}_p$ の組 W, W' に対して，$e(g^{\mathcal{P}(W)}, g^{\mathcal{P}(W')})$ を $(e(g,g)^{ww'} \mid w \in W, w' \in W')$ で定義される組とする．このとき, どの組も辞書式順序に並べられているとする．以下，$a \leq b$ となる自然数 a, b に関して，$[a,b] = \{a, a+1, \cdots, b\}$ と定義する．

構成. 以下, 証明サイズが小さい VRF の方式を提案する．

$\mathsf{Gen}(1^\lambda)$: 1^λ を入力に，$\Pi \leftarrow \mathsf{GrpGen}(1^\lambda)$ を実行し群の記述を得る．また，一様ランダムに $g, h \leftarrow \mathbb{G}^*$ を選び，$(i,k) \in [\eta]\times[\zeta]$ に関して一様ランダムに $w_0, w_{i,k} \leftarrow \mathbb{Z}_p$ を選ぶ．そして，$L_i = (w_{i,k})_{k\in[\lfloor\zeta/2\rfloor]}$ と $R_i = (w_{i,k})_{k\in[\lfloor\zeta/2\rfloor+1,\zeta]}$ とおき，下記の検証鍵と秘密鍵を出力する．

$$\mathsf{vk} = \Big(\Pi, g, h, g_0 := g^{w_0}, (g^{\mathcal{P}(L_i)}, g^{\mathcal{P}(R_i)})_{i\in[\eta]}\Big),$$
$$\mathsf{sk} = \Big(w_0, (w_{i,k})_{(i,k)\in[\eta]\times[\zeta]}\Big).$$

このとき，$W_i = (w_{i,k})_{k\in[\zeta]}$ とおくと，$e(g^{\mathcal{P}(L_i)}, g^{\mathcal{P}(R_i)}) = e(g,g)^{\mathcal{P}(W_i)}$ となることに注意されたい．

$\mathsf{Eval}(\mathsf{sk}, X)$: $X \in \{0,1\}^n$ を入力に，まず $\mathsf{S}(X) = \{s_1, \cdots, s_\ell\} \in [2\ell]$ を計算する．以下，$s_{j,k}$ を二進数表記したときの k ビット目の値を s_j とおく．ただし，$k \in [\zeta]$ である．次に，$i \in [\eta], i' \in [2,\eta]$ に関して，以下を計算する．

$$\begin{cases} \theta_i = \sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1-s_{j,k})+(-1+2s_{j,k})\cdot w_{i,k}\Big) \\ \theta_{[1:i']} = \prod_{i=1}^{i'}\sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1-s_{j,k})+(-1+2s_{j,k})\cdot w_{i,k}\Big) \end{cases}$$

ここで，$\theta := \theta_{[1:\eta]}$ とおく．このとき，$\theta_1 = \theta_{[1:1]}$ であるため，$i' = 1$ の場合は定義しない．そして，以下を出力する．

$$Y = e(g,h)^{\theta/w_0},$$
$$\pi = \Big(\pi_0 := g^{\theta/w_0}, \big(\pi_i := g^{\theta_i}\big)_{i\in[\eta]}, \big(\pi_{[1:i']} := g^{\theta_{[1:i']}}\big)_{i'\in[2,\eta]}\Big).$$

$\mathsf{Verify}(\mathsf{vk}, X, (Y,\pi))$: 最初に検証鍵 vk の正当性を確認する．もし，以下の条件のいずれか一つでも成立しなければ，0 を出力する．

(1) vk が $\Big(\Pi, g, h, g_0, (g^{\mathcal{P}(L_i)}, g^{\mathcal{P}(R_i)})_{i\in[\eta]}\Big)$ の形である．

(2) 全ての $s \in (g, h, g_0) \cup (g^{\mathcal{P}(L_i)}, g^{\mathcal{P}(R_i)})_{i\in[\eta]}$ に関して，$\mathsf{GrpVfy}(\Pi) = 1$ と $\mathsf{GrpVfy}(\Pi, s) = 1$ である．

次に，X, Y, π の正当性を確認する．それにあたり，以下の多変数多項式のの係数 $(\alpha_S)_{S\subseteq[\zeta]}$ を計算する．

$$p(\mathsf{Z}_1, \cdots, \mathsf{Z}_\zeta) = \sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1-s_{j,k})+(-1+2s_{j,k})\cdot\mathsf{Z}_k\Big) = \sum_{S\subseteq[\zeta]}\alpha_S\prod_{k\in S}\mathsf{Z}_k.$$

次に，全ての $i \in [\eta]$ と $S \subseteq [\zeta]$ に関して，$L_S = S \cap [\lfloor\zeta/2\rfloor]$ と $R_S = S \cap [\lfloor\zeta/2\rfloor+1, \zeta]$ とおき，以下のように $\Phi_{i,S}$ を計算する．

$$\Phi_{i,S} = e(g^{\prod_{k\in L_S} w_{i,k}}, g^{\prod_{k\in R_S} w_{i,k}}).$$

ここで，もし $L_S = \phi$（または $R_S = \phi$）のとき，$\prod_{k\in L_S} w_{i,k}$ （または $\prod_{k\in R_S} w_{i,k}$）を 1 と定義する．

このとき，$g^{\mathcal{P}(L_i)}, g^{\mathcal{P}(R_i)}$ が検証鍵の一部として与えられているため，これらの値が全て効率的に計算できることに注意されたい．もし，以下の条件のいずれか一つでも成立しなければ，0 を出力する．

(3) $X \in \{0,1\}^n$ と $Y \in \mathbb{G}_T$ であり，証明 π が $\pi = (\pi_0, (\pi_i)_{i\in[\eta]}, (\pi_{[1:i']})_{i'\in[2,\eta]})$ の形である．

(4) 全ての $i \in [\eta]$ と $i' \in [3,\eta]$ に関して以下が成り立つ．

$$e(\pi_i, g) = \prod_{S\subseteq[\zeta]} \Phi_{i,S}^{\alpha_S},$$

$$e(\pi_{[1:2]}, g) = e(\pi_1, \pi_2), \quad e(\pi_{[1:i']}, g) = e(\pi_{[1:i'-1]}, \pi_{i'}).$$

(5) $e(\pi_{[1:\eta]}, g) = e(\pi_0, g_0)$ と $e(\pi_0, h) = Y$ が成り立つ．

上記の条件が全て満たされているならば，1 を出力する．

上記 VRF の正当性，一意性，擬似乱数性の証明は，紙面の都合により割愛する．正当性と一意性は，認証付き双線形群生成機の定義より導かれる．また，擬似乱数性は，前章と同様に Eval アルゴリズムの次数に着目することにより，$L = \eta\zeta$ の L-DDH 問題の困難性仮定に帰着される．

謝辞 本研究の一部は，JST CREST JPMJCR1302 の支援および JSPS 科研費 17J05603 の助成を受けて行われた．

参考文献

[ACF14] Michel Abdalla, Dario Catalano, and Dario Fiore. Verifiable random functions: Relations to identity-based key encapsulaion and new constructions. *Journal of Cryptology*, 27(3):544–593, 2014.

[ASM07] Man Ho Au, Willy Susilo, and Yi Mu. Practical compact e-cash. In *ACISP*, pages 431–445. Springer, 2007.

[BGJS17] Saikrishna Badrinarayanan, Vipul Goyal, Aayush Jain, and Amit Sahai. A note on vrfs from verifiable functional encryption. Cryptology ePrint Archive, Report 2017/051, 2017. `https://eprint.iacr.org/2017/051.pdf`.

[Bit17] Nir Bitansky. Verifiable random functions from non-interactive witness-indistinguishable proofs. Cryptology ePrint Archive, Report 2017/18, 2017. `https://eprint.iacr.org/2017/018.pdf`.

[BB04] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In *CRYPTO*, pages 443–459. Springer, 2004.

[BCKL09] Mira Belenkiy, Melissa Chase, Markulf Kohlweiss, and Anna Lysyanskaya. Compact e-cash and simulatable VRFs revisited. In *PAIRING*, pages 114–141. Springer, 2009.

[BMR10] Dan Boneh, Hart William Montgomery, and Ananth Raghunathan. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In *CCS*, pages 131–140. ACM, 2010.

[BR09] Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for waters' ibe scheme. In *EUROCRYPT*, pages 407–424. Springer, 2009.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, pages 523–552. Springer, 2010.

[FHPS13] Eduarda SV Freire, Dennis Hofheinz, Kenneth G Paterson, and Christoph Striecks. Programmable hash functions in the multilinear setting. In *CRYPTO*, pages 513–530. Springer, 2013.

[GNP+15] Sharon Goldberg, Moni Naor, Dimitrios Papadopoulos, Leonid Reyzin, Sachin Vasant, and Asaf Ziv. NSEC5: Provably preventing DNSSEC zone enumeration. In *NDSS*, 2015.

[Gol08] Oded Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.

[GHKW17] Rishab Goyal, Susan Hohenberger, Venkata Koppula, and Brent Waters. A generic approach to constructing and proving verifiable random functions. Cryptology ePrint Archive, Report 2017/021, 2017. `https://eprint.iacr.org/2017/021.pdf`.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *CCS*, pages 89–98. ACM, 2006.

[HJ16] Dennis Hofheinz and Tibor Jager. Verifiable random functions from standard assumptions. In *TCC*, pages 336–362. Springer, 2016.

[HK08] Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. In *CRYPTO*, pages 21–38. Springer, 2008.

[HW10] Susan Hohenberger and Brent Waters. Constructing verifiable random functions with large input spaces. In *EUROCRYPT*, pages 656–672. Springer, 2010.

[Jag15] Tibor Jager. Verifiable random functions from weaker assumptions. In *TCC*, pages 121–143. Springer, 2015.

[JS04] Stanislaw Jarecki and Vitaly Shmatikov. Handcuffing big brother: an abuse-resilient transaction escrow scheme In *EUROCRYPT*, pages 590–608. Springer, 2004.

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. *EUROCRYPT*, pages 146–162, 2008.

[MR02] Silvio Micali and Ronald L. Rivest. Micropayments revisited. In *CT-RSA*, pages 149–163. Springer, 2002.

[MRV99] Silvio Micali, Michael Rabin, and Salil Vadhan. Verifiable random functions. In *FOCS*, pages 120–130. IEEE, 1999.

[PWH+17] Dimitrios Papadopoulos, Duane Wessels, Shumon Huque, Moni Naor, Jan Včelák, Leonid Rezyin, Sharon Goldberg. Making NSEC5 Practical for DNSSEC Cryptology ePrint Archive, Report 2017/099, 2017. `https://eprint.iacr.org/2017/099.pdf`.

[SW05] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473. Springer, 2005.

[Yam17] Shota Yamada. Asymptotically compact adaptively secure lattice ibes and verifiable random functions via generalized partitioning techniques. Cryptology ePrint Archive, Report 2017/096, to appear in CRYPTO 2017. `http://eprint.iacr.org/2017/096`.

[ZCZ16] Jian Zhang, Yu Chen, and Zhenfeng Zhang. Programmable hash functions from latties: Short signatures and ibes with small key sizes. In *CRYPT0*, pages 303–332. Springer, 2016.