This is the revised version of 0071, SCIS 2015.

**SCIS 2015** The 32nd Symposium on Cryptography and Information Security
Kokura, Japan, Jan. 20 - 23, 2015
The Institute of Electronics, Information and Communication Engineers

# A Secure Signature Scheme with Tight Reduction to the RSA Assumption from Indistinguishability Obfuscation

Takehiro Mimasu * Masayuki Abe † Tatsuaki Okamoto †

**Abstract:** We propose a new signature scheme employing an indistinguishability obfuscator in the standard model. Our goal is to build a signature scheme that is more tightly reducible to the RSA assumption than previously reported. A key technique to achieve this property is that we adopt a probabilistic construction in a signing algorithm with a (punctured) pseudorandom function instead of deterministic one in the previous scheme. Since a signing oracle in the security proof can choose randomness for punctured points, the proposed scheme is optimally reducible to the RSA assumption.

**Keywords:** Signature, RSA Assumption, Indistinguishability Obfuscation, Pseudorandom Function

## 1 Introduction

How to ensure the security of a scheme is a very important problem in cryptography. To prove that a scheme is secure, there are some models. (e.g., the random oracle model, the standard model, the generic group model and so on.) In the random oracle model [BR93], we use an ideal hash function, called a random oracle, in a proof. With the random oracle model, we can achieve a practical security level if we choose appropriate hash functions. However, because of using imaginary hash functions, the proof with the random oracle model does not guarantee its security completely in the real world.

Recently, researchers pay attention to program obfuscation. Program obfuscation is that we encrypt a program instead of a message with preserving its functionality. One of these program obfuscations is the indistinguishability obfuscator ($iO$)[GGH13]. $iO$ makes poly-time algorithms unintelligible for two obfuscated programs. By using the $iO$, some cryptosystems which are proved secure in the random oracle model, can be proved secure in the standard model [HSW14]. However, in [HSW14], the authors proved that their full domain hash signature scheme (FDH) is secure just with loose reduction to the RSA assumption using $iO$. In this paper, we realize a new signature scheme with tight reduction to the RSA assumption.

In order to consider this problem in the standard model, we used an analogy of a probabilistic signature scheme (PSS) [BR96], which was proposed to make reduction tight in a security proof of a full domain hash signature scheme in the random oracle model. Though the signing algorithm usually outputs only a signature, it outputs a signature and a random value in our randomized full domain hash signature. In addition, we construct hash functions for this change.

### 1.1 Background

#### 1.1.1 Random Oracle Model and Standard Model

In the random oracle model [BR93], we use an ideal random hash function. It usually takes a plaintext as an input and outputs a truly random value. Assuming the existence of random oracles, we can prove security more easily, because in the security proof game, a signing oracle can choose arbitrarily a random number as a signature and calculate the $e$-th power of the random number as an output of a random oracle. Let $e$ be a random integer used as a verification key in RSA FDH signature scheme. However, a random oracle does not exist, so if a scheme is proved in the random oracle model, it does not means that the scheme is proved in the real-world conditions.

a random oracle was introduced in [BR93] in 1993. In 1994, the first scheme which is IND-CCA2 secure in the random oracle model was proposed [BR94]. (The proof in that paper had mistakes and was modified in [FEPS01].) The proof of Full Domain Hash Signature is introduced in [BR96].

In contrast, there is the standard model. In the standard model, a cryptographic system consists of only actual primitives. Since if signature scheme is proved in the standard model, it means that this system is secure in real-world conditions, a proof in the standard model is desirable. For example, Cramer and Shoup public key encryption system [CS98] is a famous practical scheme proved in the standard model.

#### 1.1.2 Indistinguishability Obfuscation

In order to prove in the standard model, we use indistinguishability obfuscation($iO$). $iO$ makes two programs indistinguishable with preserving their function-

* Kyoto University. mimasu@ai.soc.i.kyoto-u.ac.jp
† Kyoto University and NTT Secure Platform Laboratory.

ality. *iO* was introduced in [GGH13]. Many papers related with *iO* are published, for example [HSW14], [SW14], [GGHR14] and so on.

#### 1.1.3 Related Works

In [SW14], Sahai and Waters have shown new constructions of some cryptographic schemes that had been proved in the random oracle model. Proposed schemes have been proved in the standard model by employing *iO* in [SW14]. However, they proved security for only selective condition.

Hohenberger, Sahai and Waters have proposed new constructions of FDH signature and BLS signature in [HSW14]. They are the selectively secure one and the adaptively secure one. The security of their FDH signature is proved under the RSA assumption in the standard model.

Ranchen and Waters have shown the construction of more practical signature scheme using *iO* and proved it adaptively secure in the standard model in [RW14].

### 1.2 Current Contribution

In this paper, we propose a new signature scheme using the indistinguishability obfuscator. Our goal is that we build a signature scheme that is more tightly reducible to the RSA assumption. A key trick to achieve this property is that we adopt a probabilistic construction in a signing algorithm with a (punctured) pseudorandom function, instead of deterministic one in the previous scheme. Since a signing oracle in the security proof can choose randomness for punctured points, this scheme is optimally reducible to the RSA assumption.

Reduction efficiency is the difference of difficulty between problems and also between a problem and a cryptographic scheme. The example of these schemes used herein is a signature scheme. Tight reduction enables a signature scheme to be secure. That means a signature scheme with tight reduction using short parameters is as secure as that with loose reduction using long parameters. If our tight scheme have a parameter of which length is 1, previous scheme ([HSW14]) needs to have a parameter of which length is $\theta(Q)$ in order to have equivalent security to that of our scheme.

## 2 Preliminaries

We introduce some definitions, an indistinguishability obfuscator, a full domain hash signature and a pseudorandom function, which we will make use of.

### 2.1 Indistinguishability Obfuscator

We first define an indistinguishability obfuscator from [GGH13]. The intuition of their first property is that even if circuits are obfuscated, same inputs necessarily generate same output. The second property is that a discriminator $D$ cannot distinguish an obfuscated program $C_1$ with the other $C_2$. The detail is below.

**Definition 2.1. Indistinguishability Obfuscator (*iO*)**

*A uniform probabilistic polynomial time turing machine (PPT) iO is called an indistinguishability obfuscator for a circuit class $\{\mathcal{C}_\lambda\}$ if it satisfies the following:*

- *For all security parameters $\lambda \in \mathbb{N}$, for all $\mathcal{C} \in \mathcal{C}_\lambda$ and for all inputs $x$, we have that*

$$\Pr[C'(x) = C(x) : C' \leftarrow iO(\lambda, C)] = 1$$

- *For any (not necessarily uniform) PPT adversaries Samp and $\mathcal{D}$, if the size of $C_0$ equals to the size of the other program ($C_1$), there exists a negligible function $\alpha$ such that following holds: if $\Pr[\forall x, C_0(x) = C_1(x) : (C_0, C_1, \tau) \leftarrow Samp(1^\lambda)] > 1 - \alpha(\lambda)$, then we have:*

$$|\Pr[\mathcal{D}(\tau, iO(\lambda, C_0)) = 1 : (C_0, C_1, \tau) \leftarrow Samp(1^\lambda)] - \Pr[\mathcal{D}(\tau, iO(\lambda, C_1)) = 1 : (C_0, C_1, \tau) \leftarrow Samp(1^\lambda)]| \leq \alpha(\lambda)$$

### 2.2 the RSA Assumption

We should recall the standard version of the RSA assumption in [RSA78][MNO11].

**Definition 2.2. the RSA Assumption** *Let* GenRSA *is an algorithm which receives $1^k$ as a security parameter, outputs (n, p, q, e) where $n = pq$ and e is a random integer such that $e < \phi(n)$ and $\gcd(e, \phi(n)) = 1$. RSA problem is that when an adversary takes (n, e, y) such that $y \xleftarrow{U} Z_n^*$ as inputs, it outputs $x \in Z_n^*$ such that $x \equiv y^{\frac{1}{e}} \pmod n$. The advantage of the PPT adversary, $\mathcal{A}$ for RSA problem is:*

$$\begin{aligned}\mathrm{Adv}_{\mathcal{A}}^{\mathrm{RSA}}(k) := \Pr[x^e \equiv y \pmod n] \,|\\ (n, p, q, e) \xleftarrow{R} \mathrm{GenRSA}(1^k);\ y \xleftarrow{U} \mathbb{Z}_n^*;\\ x \xleftarrow{R} \mathcal{A}(n, e, y)]\end{aligned}$$

*For all probabilistic polynomial time algorithms $\mathcal{A}$, if $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{RSA}}(k)$ is negligible, namely if the following condition is true, RSA assumption holds.*

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{RSA}}(k) < \epsilon(k)$$

### 2.3 Randomized Full Domain Hash (FDH) Signature

#### 2.3.1 Syntax

**Definition 2.3. Randomized Full Domain Hash (FDH) Signature** *This scheme consists of three algorithms,{Gen, Sign, Verify}. Their definitions are below.*

- *GenFDH($1^\lambda$): It runs RSA($1^k$) to obtain (N, e, d) and outputs (vk, sk) where vk = (N, e) and sk = (N, d).*
- *SignFDH(sk, m): It randomly selects r and returns a signature $(\sigma, r)$ where $\sigma = H(m, r)^d \bmod N$.*
- *VerifyFDH(vk, m, (σ, r)): It checks $\sigma^e \equiv H(m, r) \pmod N$ and outputs 1 if this condition satisfies, otherwise 0.*

### 2.3.2 Correctness

If a sign is created by SignFDH, the verification algorithm outputs 1 with probability 1.

$$\begin{aligned}\sigma^e &= (H(m,r)^d)^e \\ &= H(m,r) \bmod N \ (\because ed \equiv 1 \bmod \phi(N))\end{aligned}$$

$$\Pr\,[\sigma^e = H(m,r) \bmod N | (\sigma, r) \xleftarrow{\mathrm{R}} \mathrm{SignFDH}(sk, m) \\ , sk \leftarrow \mathrm{GenFDH}] = 1$$

## 2.4 EUF-CMA Secure

When a signature scheme is existentially unforgeable against an adaptively chosen message attack (EUF-CMA), the game that before an adversary sends queries to a challenger, he receives a verification key is considered. We show the game below.

- **Setup**: A challenger receives a security parameter $(1^k)$ and generate a verification key and a signing key by using a Gen algorithm in a signature scheme. He sends a verification key to an adversary.
- **Query Phase**: An adversary adaptively sends a message $(m_i)$ to a signing oracle and receives the signature $(\sigma_i, r_i)$ of the message at most $q_s (1 \leq i \leq q_s)$ times.
- **Challenge Phase**: An adversary guesses a pair of a message $(m^*)$ and a signature $(\sigma^*, r^*)$. If $m^* \notin m_i (1 \leq i \leq q_s)$ and a verification algorithm that receives $m^*$ and $(\sigma^*, r^*)$ outputs 1, an adversary wins.

Let $\mathrm{Exp}^{\mathrm{EUF-CMA}}_{\Sigma,\mathcal{A}}(k)$ be the above game. The advantage of an adversary $\mathcal{A}$ is a probability that $\mathcal{A}$ wins $\mathrm{Exp}^{\mathrm{EUF-CMA}}_{\Sigma,\mathcal{A}}(k)$. Namely,

$$\mathrm{Adv}^{\mathrm{EUF-CMA}}_{\Sigma,\mathcal{A}}(k) := \Pr[\mathrm{Exp}^{\mathrm{EUF-CMA}}_{\Sigma,\mathcal{A}}(k) \rightarrow 1]$$

If the advantage of an adversary satisfies following definition, a signature scheme is EUF-CMA secure.

**Definition 2.4. EUF-CMA** *Let $\epsilon(k)$ be a negligible value. A digital signature scheme $\Sigma$ is EUF-CMA secure, if it satisfies*

$$\mathrm{Adv}^{\mathrm{EUF-CMA}}_{\Sigma,\mathcal{A}} < \epsilon(k)$$

*for all probabilistic polynomial time adversaries $\mathcal{A}$.*

## 2.5 Pseudorandom Function (PRF) and punctured PRF

We consider a pseudorandom function, which is computable in polynomial time and deterministic. Its computational indistinguishability is that all probabilistic polynomial time algorithms cannot distinguish the blackbox access to truly random functions from that to pseudorandom functions This primitive was firstly introduced in [GGM84]. We focus on a punctured pseudorandom function shown in [HSW14] and introduce it. There are similar definitions in [KPTZ13] and [BW13].

**Definition 2.5. Punctured Pseudorandom Function** *A puncturable family of PRFs F mapping is given by a triple of Turing Machines $Key_F$, $Puncture_F$ and $Eval_F$ and a pair of computable functions $n(\cdot)$ and $m(\cdot)$ satisfying the following conditions:*

- **[Functionality preserved under puncturing]** *For every PPT adversary $\mathcal{A}$ such that $\mathcal{A}(1^\lambda)$ outputs a polynomial-size set $S \subseteq \{0,1\}^{n(\lambda)}$, then for all $x \in \{0,1\}^{n(\lambda)}$ where $x \notin S$, we have that:*

$$\Pr[\mathrm{Eval}_F(K,x) = \mathrm{Eval}_F(K_S,x) \\ | K \leftarrow \mathrm{Key}_F(1^\lambda), K_S = \mathrm{Puncture}_F(K,S)] = 1$$

- **[Pseudorandom at punctured points]** *For every PPT adversary $(A_1, A_2)$ such that $A_1(1^\lambda)$ outputs a polynomial-size set $S \subseteq \{0,1\}^{n(\lambda)}$ and state $\tau$, consider an experiment where $K \leftarrow \mathrm{Key}_F(1^\lambda)$ and $K_S = \mathrm{Puncture}_F(K,S)$. Then we have*

$$|\Pr[A_2(\tau, K_S, S, \mathrm{Eval}_F(K,S)) = 1] - \\ \Pr[A_2(\tau, K_S, S, U_{m(\lambda)\cdot|S|}) = 1]| = negl(\lambda)$$

*where* $\mathrm{Eval}_F(K,S)$ *denotes the concatenation of* $\mathrm{Eval}_F(K,x_1)$,...,$\mathrm{Eval}_F(K,x_k)$ *where $S = \{x_1,...,x_k\}$ is the enumeration of the elements of $S$ in lexicographic order, $negl(\cdot)$ is a negligible function, and $U_\ell$ denotes the uniform distribution over $l$ bits.*

## 2.6 Chernoff Bound

**Theorem 2.1. Chernoff Bound** *Let $X_1,...,X_n$ be independent variables such that $X_i \in \{0,1\}$ $(1 \leq i \leq n)$. Let $p \leq \frac{1}{2}$. For all $i$, let $\Pr[X_i = 1] = p$. There exists:*

$$\forall\delta(0 \leq \delta \leq p(1-p)), \Pr[|\frac{\Sigma_{i=1}^n X_i}{n} - p| \leq \delta] \\ < 2 \cdot \exp(-\frac{\delta^2}{2p(1-p)} \cdot n)$$

# 3 Fully Secure Signature Scheme

We introduce our efficient Randomized Full Domain Hash Signature scheme with adaptively secure proof. We show how to construct it in the standard model and how to prove that our scheme is secure.

## 3.1 Syntax

The syntax consists of three PPT algorithm, [Setup, Sign, Verify]. Each of them satisfies following settings:

- **Setup**$(1^\lambda)$ : The setup algorithm computes $N = pq$ where $p$ and $q$ is prime $\phi(N) = (p-1)(q-1)$ and chooses prime $e$ where $|e| = O(\lambda)$ and $d$ such that $d \times e = 1 \pmod{\phi(N)}$. It creates an obfuscation of the program Full Domain Hash in Figure 1. The program acts as a random oracle type hash function. We refer to the program as $H(m, r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, s^{(0)}, s^{(1)}) \rightarrow \{0,1\}^w$.

The setup algorithm sets $H$, $N$ and $e$ as a verification key, $vk$, and $d$ and the parameters in Full Domain Hash in Figure 1 as a signing key, $sk$.

- **Sign**($sk$, $m \in \mathcal{M}$) : This algorithm chooses random integers, $(r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, s^{(0)}, s^{(1)}) \in \mathbb{Z}_N$ and computes $\sigma = H(m, r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, s^{(0)}, s^{(1)})^d \bmod N$. It outputs $(\sigma, r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, s^{(0)}, s^{(1)})$.

- **Verify**($vk$, $m$, $r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, s^{(0)}, s^{(1)}$, $\sigma$) : This checks $\sigma^e = H(m, r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, s^{(0)}, s^{(1)}) \bmod N$. If the condition is true, outputs 1, otherwise 0.

We use two hash functions which have following construction. Note that when we use them, they are contained in an indistinguishability obfuscator.

After this, we omit an expression of security parameter, $\lambda$.

**Full Domain Hash**

**Constants:** Collision resistance hash functions $h_i : \{0,1\}^* \to \{0,1\}^\ell$ $(i = 1, ..., \ell)$, PRF $F$ key $K$, PRF $F_0$ keys $K_{1,1}, ..., K_{\ell,\ell}$ and random values $v_1, ..., v_\ell \overset{\text{U}}{\longleftarrow} \mathbb{Z}^*_N$.
**Inputs:** $m \in \mathcal{M}$, $r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell} \in \mathbb{Z}_N$ and $s^{(0)}, s^{(1)} \in \mathbb{Z}_N$.

1. Let $F_0(K_{i,k}; \cdot) : \mathbb{Z}_N \to \{0,1\}$ and $F(K; \cdot) : \{0,1\}^{\ell^2} \to \{0,1\}^n$, where $n = t\ell$, $F_0$ and $F$ be pseudorandom functions.

2. $$b^{(0)}_i = h_i(m) \oplus (F_0(K_{i,1}; r^{(0)}_{i,1}), ..., F_0(K_{i,\ell}; r^{(0)}_{i,\ell})) \quad (1 \le i \le \ell)$$

3. $(c^{(0)}_1, ..., c^{(0)}_\ell) = F(K; (b^{(0)}_1, ..., b^{(0)}_\ell))$

4. Repeat 2. and 3. for $r^{(1)}_{i,k}$ and compute $(c^{(1)}_1, ..., c^{(1)}_\ell)$ $(1 \le i, k \le \ell)$.

5. Compute $a_i = s^{(0)} \times c^{(0)}_i + s^{(1)} \times c^{(1)}_i$ $(1 \le i \le \ell)$.

6. Output $v_1^{a_1} \times v_2^{a_2} \times ... \times v_\ell^{a_\ell} \bmod N$

Figure 1: Full Domain Hash

**Full Domain Hash***

**Constants:** Collision resistance hash functions $h_i : \{0,1\}^* \to \{0,1\}^\ell$ $(i = 1, ..., \ell)$, punctured PRF $F$ key $K_S$, punctured PRF $F_0$ keys $K_{1,1,S_{1,1}}, ..., K_{\ell,\ell,S_{\ell,\ell}}$, random values $v_1, ..., v_\ell \overset{\text{U}}{\longleftarrow} \mathbb{Z}^*_N$, $R^{(0)}_1, ..., R^{(0)}_q, R^{(1)}_1, ..., R^{(1)}_q \overset{\text{U}}{\longleftarrow} \{0,1\}^{\ell^2}$, $S = \{R^{(0)}_1, ..., R^{(0)}_q, R^{(1)}_1, ..., R^{(1)}_q\}$, $Q^{(0)}_{i,k,j,\beta}, Q^{(1)}_{i,k,j,\beta} \overset{\text{U}}{\longleftarrow} \mathbb{Z}_N$, $S_{i,k} = \{Q^{(0)}_{i,k,j,0}, Q^{(0)}_{i,k,j,1}, Q^{(1)}_{i,k,j,0}, Q^{(1)}_{i,k,j,1} \mid 1 \le j \le q\}$, where $F(K; R^{(\gamma)}_j) = F(K_S; R^{(\gamma)}_j)$ $(\gamma \in \{0,1\})$ and $F_0(K_{i,k}; Q^{(\gamma)}_{i,k,j,\beta}) = F_0(K_{i,k,S_{i,k}}; Q^{(\gamma)}_{i,k,j,\beta})$ $(j = 1, ..., q)$ $(\gamma, \beta \in \{0,1\})$.
**Inputs:** $m \in \mathcal{M}$, $r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell} \in \mathbb{Z}_N$ and $s^{(0)}, s^{(1)} \in \mathbb{Z}_N$.

1. Let $F_0(K_{i,k,S_{i,k}}; \cdot) : \mathbb{Z}_N \to \{0,1\}$ and $F(K_S; \cdot) : \{0,1\}^{\ell^2} \to \{0,1\}^n$, where $n = t\ell$, $F_0$ and $F$ be punctured pseudorandom functions.

2. $$b^{(0)}_i = h_i(m) \oplus (F_0(K_{i,1,S_{i,1}}; r^{(0)}_{i,1}), ..., F_0(K_{i,\ell,S_{i,\ell}}; r^{(0)}_{i,\ell})) \ (1 \le i \le l)$$

3. $(c^{(0)}_1, ..., c^{(0)}_\ell) = F(K_S; (b^{(0)}_1, ..., b^{(0)}_\ell))$

4. Repeat 2. and 3. for $r^{(1)}_{i,k}$ and compute $(c^{(1)}_1, ..., c^{(1)}_\ell)$ $(1 \le i, k \le \ell)$.

5. Compute $a_i = s^{(0)} \times c^{(0)}_i + s^{(1)} \times c^{(1)}_i$ $(1 \le i \le \ell)$.

6. Output $v_1^{a_1} \times v_2^{a_2} \times ... \times v_\ell^{a_\ell} \bmod N$

Figure 2: Full Domain Hash*[1]

[1] Precisely, $\#S_{i,k} = \lambda q$, i.e. $S_{i,k} = \{Q_{i,k,\theta} \mid \theta = 1, ..., \lambda q\}$.

**Full Domain Hash****

**Constants:** Collision resistance hash functions $h_i : \{0,1\}^* \rightarrow \{0,1\}^\ell$ $(i = 1, ..., \ell)$, punctured PRF $F$ key $K_S$, punctured PRF $F_0$ keys $K_{1,1,S_{1,1}}, ..., K_{\ell,\ell,S_{\ell,\ell}}$, random values $v_1, ..., v_\ell \overset{\text{U}}{\leftarrow} \mathbb{Z}_N^*$, $X_1^{(0)}, ..., X_q^{(0)}, X_1^{(1)}, ..., X_q^{(1)} \overset{\text{U}}{\leftarrow} \{0,1\}^n$, $R_1^{(0)}, ..., R_q^{(0)}, R_1^{(1)}, ..., R_q^{(1)} \overset{\text{U}}{\leftarrow} \{0,1\}^{\ell^2}$, $S = \{R_1^{(0)}, ..., R_q^{(0)}, R_1^{(1)}, ..., R_q^{(1)}\}$, $Q_{i,k,j,\beta}^{(0)}, Q_{i,k,j,\beta}^{(1)} \overset{\text{U}}{\leftarrow} \mathbb{Z}_N$, $S_{i,k} = \{Q_{i,k,j,0}^{(0)}, Q_{i,k,j,1}^{(0)}, Q_{i,k,j,0}^{(1)}, Q_{i,k,j,1}^{(1)} \mid 1 \leq j \leq q\}$, where $X_j^{(\gamma)} = F(K_S; R_j^{(\gamma)})$ $(\gamma \in \{0,1\})$ and $\beta = F_0(K_{i,k,S_{i,k}}; Q_{i,k,j,\beta}^{(\gamma)})$ $(j = 1, ..., q)$ $(\gamma, \beta \in \{0,1\})$.

**Inputs:** $m \in \mathcal{M}$, $r_{1,1}^{(0)},...,r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)},...,r_{\ell,\ell}^{(1)} \in \mathbb{Z}_N$ and $s^{(0)}, s^{(1)} \in \mathbb{Z}_N$.

1. Let $F_0(K_{i,k,S_{i,k}}; \cdot) : \mathbb{Z}_N \rightarrow \{0,1\}$ and $F(K_S; \cdot) : \{0,1\}^{\ell^2} \rightarrow \{0,1\}^n$, where $n = t\ell$, $F_0$ and $F$ be punctured pseudorandom functions.

2. $$b_i^{(0)} = h_i(m) \oplus (F_0(K_{i,1,S_{i,1}}, r_{i,1}^{(0)}), ..., F_0(K_{i,\ell,S_{i,\ell}}, r_{i,\ell}^{(0)}))\ (1 \leq i \leq l)$$

3. $(c_1^{(0)}, ..., c_\ell^{(0)}) = F(K_S; (b_1^{(0)}, ..., b_\ell^{(0)}))$

4. Repeat 2. and 3. for $r_{i,k}^{(1)}$ and compute $(c_1^{(1)}, ..., c_\ell^{(1)})$ $(1 \leq i, k \leq \ell)$.

5. Compute $a_i = s^{(0)} \times c_i^{(0)} + s^{(1)} \times c_i^{(1)}$ $(1 \leq i \leq \ell)$.

6. Output $v_1^{a_1} \times v_2^{a_2} \times ... \times v_\ell^{a_\ell} \bmod N$

Figure 3: Full Domain Hash**[2]

## 3.2 Proof of Security

Our purpose in this section is that we prove the above signature scheme is secure. Previously in the random oracle model, we can refer to the hash oracle about queries which are sent from the adversary. We must replace a random oracle with an actual hash function in the standard model, so we cannot use the property above. Because of that reason, the full domain hash signature scheme in [HSW14] uses punctured pseudorandom functions (PRF), $iO$, and admissible hash functions in a proof for an adaptive attack. Though the scheme in [HSW14] has low efficiency of reduction, we achieve optimal efficiency in the proposed scheme.

**Theorem 3.1.** *If our obfuscation scheme is indistinguishably secure, punctured PRFs $F$ and $F_0$ are secure and the RSA assumption holds, our signature scheme is existentially unforgeable against adaptively chosen message attack (EUF-CMA).*

We describe a game sequence proof where the first hybrid is corresponding to the original signature security game. In this game in our scheme, the challenger runs setup algorithm and some value is set. The adversary receives $vk$ and a hidden hash function from the $iO$. The adversary adaptively sends $m_j$ to a signing oracle and gets its signature, $\sigma_j$ and a randomness $(r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, s^{(0)}, s^{(1)})$. In the challenge phase, the adversary sends $(m^*, \sigma^*)$ with a randomness $(r_{1,1}^{(0)*}, ..., r_{\ell,\ell}^{(0)*}, r_{1,1}^{(1)*}, ..., r_{\ell,\ell}^{(1)*}, s^{(0)*}, s^{(1)*})$ such that $m^* \neq m_j$ $(j = 1, ..., q)$ to the challenger.

Previously in [HSW14], they used a partitioning technique to isolate a query space from a challenge space. A query space is that a signing oracle can return a correct signature. A partitioning technique was used to give a reduction the ceiling of a probability in a previous work. For example, an admissible hash function is used in [HSW14]. It devides a query space and a challenge space from a message space and a query space is larger than a challenge space. When an adversary queries messages randomly, they are in a challenge space barely in non-negligible probability. However in our scheme, we do not use the partitioning technique, that is, a query space is equivalent to a message space, $\{0,1\}^n$. Hence, we can embedded an RSA challenge problem into our scheme optimally for the security reduction.

For the solution to do that, we add a random value to the input of a hash function (i.e. $H(m, r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, s^{(0)}, s^{(1)})$ such that $r_{i,k}^{(0)}, r_{i,k}^{(1)}, s^{(0)}, s^{(1)}$ is random integers in $\mathbb{Z}_N$.). Therefore, a signing oracle can control the output of a hash function.

In the last game, we prove that we can construct an adversary that breaks the RSA assumption by using that breaks our signature scheme.

- $\mathsf{Hyb}_0$ : In the first hybrid, following EUF-CMA game is played.

  1. The challenger runs the setup algorithm. It chooses $e$ as a random chosen prime between 1 and $\phi(N)$ such that $\gcd(\phi(N), e) = 1$ and $|e| = O(\lambda)$.
  2. The attacker receives the verification key.
  3. $H$ is created as an obfuscated program of Full Domain Hash in Figure 1.
  4. The attacker queries the signing oracle at most $q$ times on messages $m_1, ..., m_q$. In the $j$-th query, the challenger receives $m_j$ and chooses $r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, s^{(0)}, s^{(1)}$ uniformly. It computes $\sigma_j = H(m_j, r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, s^{(0)}, s^{(1)})^d$ mod

[2] Precisely, $F_0(K_{i,k,S_{i,k}}; Q_{i,k,\theta})$ is uniformly selected from {0,1}. We set $Q_{i,k,j,\beta} = Q_{i,k,\theta}$ such that $F_0(K_{i,k,S_{i,k}}; Q_{i,k,\theta}) = \beta$ for $\theta = 1, ...$ and $j = 1, ..., q$. $\#\{Q_{i,k,\theta} \mid F_0(K_{i,k,S_{i,k}}; Q_{i,j,k,\beta}) = \beta\} \geq 2q$ for $\beta = \{0,1\}$ with overwhelming probability in $\lambda$ (by Chernoff bound).

$N$. The attacker receives $(\sigma_j, r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, s^{(0)}, s^{(1)})$.

5. The adversary sends $(m^*, \sigma^*, r^{(0)*}_{1,1}, ..., r^{(0)*}_{\ell,\ell}, r^{(1)*}_{1,1}, ..., r^{(1)*}_{\ell,\ell}, s^{(0)*}, s^{(1)*})$ as a challenge to the challenger. If $\text{Verify}(m^*, \sigma^*, r^{(0)*}_{1,1}, ..., r^{(0)*}_{\ell,\ell}, r^{(1)*}_{1,1}, ..., r^{(1)*}_{\ell,\ell}, s^{(0)}, s^{(1)})$ outputs 1, the adversary wins, otherwise it loses.

- $\text{Hyb}_1$ : This game is equivalent to $\text{Hyb}_0$ except for 3.. We use an obfuscated program of Full Domain Hash* in Figure 2 instead of that of Full Domain Hash in Figure 1.

- $\text{Hyb}_2$ : This game is equivalent to $\text{Hyb}_1$ except for 3.. We use an obfuscated program of Full Domain Hash** in Figure 3 instead of that of Full Domain Hash* in Figure 2.

- $\text{Hyb}_3$ : The last game has following process.

  1. The challenger runs the setup algorithm. It chooses $e$ as a random chosen prime between 1 and $\phi(N)$ such that $\gcd(\phi(N), e) = 1$ and $|e| = O(\lambda)$.
  2. The attacker receives the verification key.
  3. Set punctured PRF $F(K_S; R^{(\gamma)}_j)$ and $F_0(K_{i,k,S_{i,k}}; Q^{(\gamma)}_{i,k,j,\beta})$ ($\gamma \in \{0,1\}$) to the same ones in Figure 3.
  4. Compute $(v_1, ..., v_\ell)$ with the following process. $(v_1, ..., v_\ell)$ are integers used in the hash function.
     – Each ($\alpha_1$, ..., $\alpha_\ell$) is chosen as a random integer in $\{0,1\}^{3|N|}$ such that $\gcd(e, \alpha_i) = 1$.
     – $g$ is chosen as a random integer.
     – Compute $v_1 = g^{\alpha_1} \bmod N$, $v_2 = g^{\alpha_2} \bmod N$, ..., $v_\ell = g^{\alpha_\ell} \bmod N$.
  5. $H$ is created as an obfuscated program of Full Domain Hash** in Figure 3 by using values specified 3. and 4..
  6. The attacker queries the signing oracle at most $q$ times on messages $m_1, ..., m_q$. In the $j$-th query, the challenger receives $m_j$. $R^{(0)}_j = (b^{(0)}_1, ..., b^{(0)}_\ell)$ and $R^{(1)}_j = (b^{(1)}_1, ..., b^{(1)}_\ell)$.
     – Choose each $r^{(0)}_{i,k} = Q^{(0)}_{i,k,j,0}$ or $r^{(0)}_{i,k} = Q^{(0)}_{i,k,j,1}$ such that $(F_0(K_{i,1,S_{i,1}}; r^{(0)}_{i,1}),..., F_0(K_{i,\ell,S_{i,\ell}}, r^{(0)}_{i,\ell})) = b^{(0)}_i \oplus h_i(m)$.
     – Choose each $r^{(1)}_{i,k}$ with the same way for $Q^{(1)}_{i,k,j,\beta}$.
     – $(c^{(0)}_1, ..., c^{(0)}_\ell) = F(K_S; (b^{(0)}_1, ..., b^{(0)}_\ell))$ and $(c^{(1)}_1, ..., c^{(1)}_\ell) = F(K_S; (b^{(1)}_1, ..., b^{(1)}_\ell))$.
     – Randomly select $s^{(0)}, s^{(1)} \in \mathbb{Z}_N$ such that $e \mid s^{(0)}\Sigma\alpha_i c^{(0)}_i + s^{(1)}\Sigma\alpha_i c^{(1)}_i$.
     – Computes $\sigma_j = g^{(s^{(0)}\Sigma\alpha_i c^{(0)}_i + s^{(1)}\Sigma\alpha_i c^{(1)}_i)/e} \bmod N$.
     – The attacker receives $(\sigma_j, r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, s^{(0)}, s^{(1)})$.
  7. The adversary sends $(m^*, \sigma^*, r^{(0)*}_{1,1}, ..., r^{(0)*}_{\ell,\ell}, r^{(1)*}_{1,1}, ..., r^{(1)*}_{\ell,\ell}, s^{(0)*}, s^{(1)*})$ as a challenge to the challenger. If $\text{Verify}(m^*, \sigma^*, r^{(0)*}_{1,1}, ..., r^{(0)*}_{\ell,\ell}, r^{(1)*}_{1,1}, ..., r^{(1)*}_{\ell,\ell}, s^{(0)*}, s^{(1)*})$ outputs 1, the adversary wins, otherwise it loses.

When the following three lemma is proved, Theorem3.1 is proved.

**Lemma 3.1.** *If an indistinguishability obfuscator is secure, the advantage of the adversary in $Hyb_0$ is close to that in $Hyb_1$.*

*Proof.* For the proof of this lemma, we give a reduction to an indistinguishability obfuscator. We begin with considering two algorithms *Samp* and $D$.

*Samp*$(1^\lambda)$ behaves as following. *Samp* runs Setup, gets $(vk,\ sk)$ and sets $\tau = (vk, sk)$. Finally, *Samp* creates $C_1$ as Full Domain Hash in Figure 1 and $C_2$ as Full Domain Hash* in Figure 2. If the size of $C_1$ is larger than that of $C_2$, $C_2$ is padded in order to be a same size as $C_1$ and vice versa. Because of the functionality preserved property of punctured PRF and the construction of our hash functions, $C_1$ and $C_2$ is identical on every inputs.

$D$ receives $\tau$ and $iO(C_1)$ or $iO(C_2)$, the obfuscation of $C_1$ or $C_2$ as inputs. $D$ invokes the adversary in $\text{Hyb}_0/\text{Hyb}_1$. $D$ plays the role of the challenger in $\text{Hyb}_0$ or $\text{Hyb}_1$ by using $iO(C_1)$ or $iO(C_2)$. Finally, the adversary sends a forgery signature and wins if it is validly verified. If the attacker wins, $D$ outputs 1. If $D$ receives $iO(C_1)$, the probability that $D$ outputs 1 is exactly same as the one that the adversary wins in $\text{Hyb}_0$. Similarly, if $D$ receives $iO(C_2)$, the probability $D$ outputs 1 is same as that the adversary wins in $\text{Hyb}_1$.

Therefore, if the difference of the adversary's advantage of $\text{Hyb}_0$ and $\text{Hyb}_1$ is non-negligible, $D$ can distinguish $iO(C_1)$ from $iO(C_2)$ in non-negligible probability. Thus, this lemma follows.

□

**Lemma 3.2.** *If the punctured pseudorandom functions $F$ and $F_0$ are secure, the advantage of the adversary in $Hyb_1$ is close to that in $Hyb_2$.*

*Proof.* We prove this lemma by giving a reduction to the pseudorandomness property at punctured points for punctured PRFs. We consider $\mathcal{A}$, the adversary of a punctured PRF.

$\mathcal{A}$ takes as an input $F(K_S; \cdot)$ with $F(K; \cdot)$ values for $S$ or $F(K_S; \cdot)$ with random values for $S$. $\mathcal{A}$ creates an obfuscated program $H$ of Figure 2/Figure 3 by using the given punctured PRF. $\mathcal{A}$ runs $\text{Hyb}_1/\text{Hyb}_2$ with the adversary by using $H$. If the adversary receives

$F(K_S;\cdot)$ with $F(K;\cdot)$ values for $S$, this game is exact same as $\mathrm{Hyb}_1$. If the adversary receives $F(K_S;\cdot)$ with random values for $S$, this game is exact same as $\mathrm{Hyb}_2$. We do the same procedure for punctured PRF $F_0$ in the hybrid argument.

Hence, this lemma follows.

□

**Lemma 3.3.** *The advantage of the adversary in Hyb$_2$ is close to that in Hyb$_3$.*

*Proof.* Firstly, we prove that the distribution of $v_i \in \mathbb{Z}_N^*$ $(1 \leq i \leq l)$ in $\mathrm{Hyb}_2$ is statistically close to that in $\mathrm{Hyb}_3$. Random variable $v_i$ in $\mathrm{Hyb}_2$ is uniformly distributed in $Z_N^*$. In contrast, $v_i$ in $\mathrm{Hyb}_3$ is computed by using $\alpha_i$ such that $\gcd(e, \alpha_i) = 1$. Namely, when we compute $v_i = g^{\alpha_i \bmod \phi(N)}$, we exclude $\alpha_i$ such that $\alpha_i = (\alpha_i \bmod \phi(N)) + E_i\phi(N) \equiv 0 \pmod e$. We choose $E_i$ uniformly. In order to satisfy $\alpha_i \not\equiv 0 \pmod e$, $\alpha_i \bmod \phi(N)$ is chosen as $\alpha_i \bmod \phi(N) \not\equiv -E_i\phi(N) \pmod e$. $E_i\phi(N) \bmod e$, that is the constrained condition of $\alpha_i \bmod \phi(N)$, is (almost) uniformly distributed, because $\gcd(e, \phi(N)) = 1$. Hence, $\alpha_i \bmod \phi(N)$ is (almost) uniformly distributed. Since $v_i$ is determined by $\alpha_i \bmod \phi(N)$, the distribution of $v_i$ in $\mathrm{Hyb}_2$ is statistically close to that in $\mathrm{Hyb}_3$.

Secondly, we consider $s^{(0)}$ and $s^{(1)}$ when the adversary sends $j$-th query in the hybrid game. The adversary can get $g^{\Sigma\alpha_i c_i^{(0)} \bmod \phi(N)}$ and $g^{\Sigma\alpha_i c_i^{(1)} \bmod \phi(N)}$, since the adversary can choose $s^{(0)}, s^{(1)}$ as he wants and if $s^{(0)} = 1$, $s^{(1)} = 0$, the output of the hash function is $g^{\Sigma\alpha_i c_i^{(0)} \bmod \phi(N)}$ and the other one can be computed in the same way. Variables $s^{(0)}$ and $s^{(1)}$ are chosen randomly in $\mathrm{Hyb}_2$. Variables $s^{(0)}$ and $s^{(1)}$ are chosen as $e \mid s^{(0)}\Sigma\alpha_i c_i^{(0)} + s^{(1)}\Sigma\alpha_i c_i^{(1)}$ in $\mathrm{Hyb}_3$. Namely, $s^{(0)}\Sigma\alpha_i c_i^{(0)} + s^{(1)}\Sigma\alpha_i c_i^{(1)} \equiv 0 \pmod e$. If the challenger chooses $s^{(0)}$ randomly, $s^{(1)}$ should satisfy the following relation with $s^{(0)}$: $s^{(1)}\Sigma\alpha_i c_i^{(1)} \equiv s^{(0)}\Sigma\alpha_i c_i^{(0)} \pmod e$. That is, $s^{(1)}(\Sigma(\alpha_i \bmod \phi(N))c_i^{(1)} + \Sigma E_i\phi(N)c_i^{(1)}) \equiv s^{(0)}(\Sigma(\alpha_i \bmod \phi(N))c_i^{(0)} + \Sigma E_i\phi(N)c_i^{(0)}) \pmod e$. The values of $\{c_i^{(0)}\}$ and $\{c_i^{(1)}\}$ are unknown to the adversary and uniformly distributed, because they are the output of the punctured PRF at punctured points. The adversary, however, can get partial information on $\{c_i^{(0)}\}$ and $\{c_i^{(1)}\}$, $g^{\Sigma\alpha_i c_i^{(0)} \bmod \phi(N)}$ and $g^{\Sigma\alpha_i c_i^{(1)} \bmod \phi(N)}$. Then the values of $\{c_i^{(0)}\}$ and $\{c_i^{(1)}\}$ still have the freedom of randomness of $(2\ell - 2)$ variables. In addition, the value of $E_i$ is unknown to the adversary and distributed in a large space ($|E_i| \simeq 2|N|$). Therefore, $s^{(1)}$ is uniformly and independently distributed from $s^{(0)}$. Hence, the distribution of $s^{(0)}$ and $s^{(1)}$ in $\mathrm{Hyb}_2$ and those of $\mathrm{Hyb}_3$ is equivalent.

Thus, this lemma follows.

□

**Lemma 3.4.** *If the RSA assumption holds, the advantage of an PPT adversary in Hyb$_3$ is negligible.*

*Proof.* For this proof, we begin consider two probabilistic polynomial-time algorithms $\mathcal{A}$ and $\mathcal{B}$. Let $\mathcal{B}$ be the adversary for the RSA assumption, and $\mathcal{A}$ be that for our signature scheme. $\mathcal{B}$ uses $\mathcal{A}$ in itself to break the RSA assumption. $\mathcal{B}$ receives as input an RSA challenge $(N, e, y)$ where $N = pq$ such that $p$ and $q$ are prime, prime[3] $e \in [1, \phi(N)]$ such that $\phi(N) = (p-1)(q-1)$ and $\gcd(\phi(N), e) = 1$, and $y \in Z_N^*$.

$\mathcal{B}$ plays a role of the challenger in $\mathrm{Hyb}_3$ such that $g = y$ with the adversary $\mathcal{A}$.

We assume that the adversary $\mathcal{A}$ outputs a valid signature $(m^*, \sigma^*, r_{1,1}^{(0)*}, ..., r_{\ell,\ell}^{(0)*}, r_{1,1}^{(1)*}, ..., r_{\ell,\ell}^{(1)*}, s^{(0)*}, s^{(1)*})$. $\mathcal{B}$ can compute $c_i^{(0)*}$, $c_i^{(1)*}$ and $s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$. $\mathcal{B}$ checks whether $e \nmid s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$. If $e \nmid s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$, $\mathcal{B}$ breaks the RSA assumption. We consider its probability.

There are two cases, where in case (1), $\{c_i^{(0)*}\}$ and $\{c_i^{(1)*}\}$ are the output of punctured PRF $F$ at punctured points and otherwise in case (2). We now show that case (1) occurs in negligible probability. For each collision resistance hash function $h_i$ $(1 \leq i \leq l)$, there is difference, at least 1bit, between $h_i(m^*)$ and $h_i(m_j)$ and we assume that it is in $k$-th bit. Since the punctured points of punctured PRF $F_0$ is uniformly distributed in $\mathbb{Z}_N$ and the distribution of its output at punctured points is also uniform, the probability that the adversary chooses $r_{i,k}^{(0)}$ such that $(F_0(K_{i,1,S_{i,k}}; r_{i,k}^{(0)}) = [b_i]_k^{(0)} \oplus [h_i(m^*)]_k$ is $\frac{1}{2}$. Hence, the adversary can get the output of punctured PRF $F$ at punctured points in negligible probability, at most $\frac{1}{2^\ell}$.

Then, we show that $e \mid s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$ with negligible probability in case (2). The point here is that the adversary has no idea of $E_i$ such that $\alpha_i = \alpha_i \bmod \phi(N)) + E_i\phi(N)$. The exponent of the output of the hash function is $s^{(0)*}(\Sigma(\alpha_i \bmod \phi(N))c_i^{(0)} + \Sigma E_i\phi(N)c_i^{(0)}) + s^{(1)*}(\Sigma(\alpha_i \bmod \phi(N))c_i^{(1)} + \Sigma E_i\phi(N)c_i^{(1)})$. Random value $E_i$ is uniformly distributed in space of which the size similarly equals to $2|N|$. Hence, the probability that $e \mid s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$ is close to $\frac{1}{e}$, because $\gcd(\phi(N), e) = 1$. Since the size of $e$ is $O(\lambda)$, the adversary can send a forged signature with negligible probability such that $e \mid s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$. Hence, $\gcd(e, s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}) = 1$ with overwhelming probability.

Let $A$ be $s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$. Note that $\gcd(e, A) = 1$.

$$\begin{aligned}\sigma^* &= y^{(s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*})\frac{1}{e}} \\ &= y^{A\cdot\frac{1}{e}} \\ &= x^A \pmod N,\end{aligned}$$

where $x = y^{\frac{1}{e}} \bmod N$.

Since $\gcd(e, A) = 1$, we can compute $\gamma$, $\delta$ such that

$$A\gamma + e\delta = 1$$

[3] We assume that $e$ is a prime for simplicity of description, but it is easy to relax the primarity.

with the extended euclidean algorithm.

$$\begin{aligned}\sigma^{*\gamma} \times y^{\delta} &= (x^{A})^{\gamma} \times (x^{e})^{\delta} \\ &= x \pmod N\end{aligned}$$

$\mathcal{B}$ can compute $x = y^{\frac{1}{e}} \bmod N$, which is the answer for the RSA challenge. This contradicts the RSA assumption.

This lemma follows.

□

## 3.3 Assessment of efficiency

The challenger can return valid signatures for all signing queries in $\text{Hyb}_3$ with the probability 1. The probability that case (1) occurs is at most $\frac{1}{2^{\ell}}$ and that case (2) is $\frac{1}{e}$ ($e = O(\lambda)$).

Let $\epsilon$ be a negligible value. $\text{Adv}_{\Sigma,\mathcal{A}}^{\text{Hyb}_0}$ denotes the advantage of an adversary for $\text{Hyb}_0$. $\text{Hyb}_0$ is an EUF-CMA game. Since Lemma 3.1.,..., 3.4. works, $\text{Adv}_{\Sigma,\mathcal{A}}^{\text{EUF-CMA}}$, which equals to $\text{Adv}_{\Sigma,\mathcal{A}}^{\text{Hyb}_0}$, satisfies following relation with the advantage of an adversary for the RSA assumption ($\text{Adv}_{\Sigma,\mathcal{A}}^{\text{RSA}} = \text{Adv}_{\Sigma,\mathcal{A}}^{\text{Hyb}_3}$)

$$\begin{aligned}\text{Adv}_{\Sigma,\mathcal{A}}^{\text{EUF-CMA}} &= \text{Adv}_{\Sigma,\mathcal{A}}^{\text{Hyb}_0} \\ &= \text{Adv}_{\Sigma,\mathcal{A}}^{\text{Hyb}_1} + \text{Adv}_{\Sigma,\mathcal{A}}^{\text{iO}} + \epsilon \\ &= \text{Adv}_{\Sigma,\mathcal{A}}^{\text{Hyb}_1} + \epsilon \ (\because \textit{iO is secure}) \\ &= \text{Adv}_{\Sigma,\mathcal{A}}^{\text{Hyb}_2} + \text{Adv}_{\Sigma,\mathcal{A}}^{\text{PPRF}} + \epsilon \\ &= \text{Adv}_{\Sigma,\mathcal{A}}^{\text{Hyb}_2} + \epsilon \ (\because \textit{PPRF is secure}) \\ &= \text{Adv}_{\Sigma,\mathcal{A}}^{\text{Hyb}_3} + \epsilon \\ &\leq \text{Adv}_{\Sigma,\mathcal{A}}^{\text{RSA}} + \frac{1}{2^{\ell}} + \frac{1}{e} + \epsilon \\ &= \text{Adv}_{\Sigma,\mathcal{A}}^{\text{RSA}} + \epsilon\end{aligned}$$

## References

[BGI13] Elette Boyle, Shafi Goldwasser and Ioana Ivan, "Functional Signatures and Pseudorandom Functions," *IACR Cryptography ePrint Archive*, 2013:401, 2013.

[BR93] Mihir Bellare and Philip Rogaway, "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols," *ACM Conference on Computer and Communications Security*, pages 62-73, 1993.

[BR94] Mihir Bellare and Philip Rogaway, "Optimal Asymmetric Encryption – How to Encrypt with RSA," *EUROCRYPT*, LNCS vol. 950, pages 92-111, 1994.

[BR96] Mihir Bellare and Philip Rogaway, "The Exact Security of Digital Signatures - How to Sign with RSA and Rabin," *EUROCRYPT*, pages 399-416, 1996.

[BW13] Dan Boneh and Brent Waters, "Constrained Pseudorandom Functions and Their Applications," *ASIACRYPT*, pages 280-300, 2013.

[CS98] Ronald Cramer and Victor Shoup, "A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack," *CRYPTO*, pages 13-25, 1998.

[FEPS01] Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointchebal and Jacques Stern, "RSA-OAEP Is Secure under the RSA Assumption," *CRYPTO*, vol. 2139 pages 260-274, 2001.

[GGH13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai and Brent Waters, "Candidate Indistinguishability Obfuscation and Functional Encryption for all circuits," *FOCS*, pages 40-49, 2013.

[GGHR14] Sanjam Garg, Craig Gentry, Shai Halevi and Mariana Raykova, "Two-Round Secure MPC from Indistinguishability Obfuscation," *TCC*, pages 74-94, 2014.

[GGM84] Oded Goldreich, Shafi Goldwasser and Silvio Micali, "How to Construct Random Functions (extentded abstract)," *FOCS*, pages 464-479, 1984.

[KPTZ13] Aggelos Kiayias, Stavrs Papadopoulos, Nikos Triandopoulos and Thomas Zacharias, "Delegatable Pseudorandom Functions and Applications," *IACR Cryptography ePrint Archive*, 2013:379, 2013.

[KS98] B. Kaliski and J. Staddon, *PKCS#1: RSA Cryptography Specifications Version 2.0.(RFC2437)*, 1998.

[KSOO07] Yuichi Komano, Atsushi Shimbo, Koji Okada and Kazuo Ohta, "Provably Secure Digital Signature Scheme with Additional Functionality" *TOSHIBA Review*,Vol.62 No.7, pages 35-38, 2007.

[MNO11] Daisuke Moriyama, Ryo Nishimaki and Tatsuaki Okamoto, *Theory of Public-Key Cryptography*, Kyoritsu Shuppan, 2011.

[HSW14] Susan Hohenberger, Amit Sahai and Brent Waters, "Replacing a Random Oracle: Full Domain Hash From Indistinguishability Obfuscation," *EUROCRYPT*, pages 201-220, 2014.

[RSA78] Ronald L. Rivest, Adi Shamir and Leonard M. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Commun. ACM* 21(2):120-126, 1978.

[RW14] Kim Ranchen and Brent Waters, "Fully Secure and Fast Signing from Obfuscation," *CCS* , pages 659-673, 2014.

[SW14] Amit Sahai and Brent Waters, "How to Use Indistinguishability Obfuscation: Deniable Encryption, and More," *STOC*, pages 475-484, 2014.

## A Strongly Existentially Unforgeable (sEUF)

In addition to the condition of EUF-CMA, even if an adversary make a forged signature by using the same message sent to a signing oracle, the forged signature is also a valid forgery in sEUF.

## B Improved version of our Randomized Full Domain Hash Signature

In this paper, our proposed scheme (Randomized Full Domain Hash Signature) is only unforgeable (not strongly unforgeable). We can make sure of that with a following example. We assume that in $j$-th query, there is a valid signature of

$$(\sigma, (r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, s^{(0)}, s^{(1)}))$$

for $m_j$ and an adversary chooses the values $(r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, 2s^{(0)}, 2s^{(1)})$ as $(r_{1,1}^{(0)*}, ..., r_{\ell,\ell}^{(0)*}, r_{1,1}^{(1)*}, ..., r_{\ell,\ell}^{(1)*}, s^{(0)*}, s^{(1)*})$ for a forgery. In $i$-th value $(i = 1, ..., \ell)$,

$$\begin{aligned} a_i^* &= s^{(0)*} \times c_i^{(0)*} + s^{(1)*} \times c_i^{(1)*} \\ &= 2s^{(0)} \times c_i^{(0)} + 2s^{(1)} \times c_i^{(1)} \\ &= 2(s^{(0)} \times c_i^{(0)} + s^{(1)} \times c_i^{(1)}) \\ &= 2a_i, \end{aligned}$$

since PRFs $F_0$ and $F$ are deterministic.

A forgery $\sigma^*$ satisfies following relation with a $j$-th valid signature $\sigma$ responded for $m_j$:

$$\begin{aligned} \sigma^* &= (v_1^{a_1} \times v_2^{a_2} \times, ..., \times v_\ell^{a_\ell} \bmod N)^d \\ &= (v_1^{2a_1} \times v_2^{2a_2} \times, ..., \times v_\ell^{2a_\ell} \bmod N)^d \\ &= (v_1^{a_1} \times v_2^{a_2} \times, ..., \times v_\ell^{a_\ell} \bmod N)^{2d} \\ &= \sigma^2 \bmod N \end{aligned}$$

Hence, an adversary can make a valid forgery,

$$(m_j, \sigma^2, (r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, 2s^{(0)}, 2s^{(1)})),$$

for the same message, $m_j$, queried to a signing oracle.

We can, however, make an improved signature scheme under sEUF-CMA secure with simple changes from our randomized full domain hash signature scheme under EUF-CMA.

### B.1 Syntax

The syntax consists of three PPT algorithm, [Setup, Sign, Verify]. Each of them satisfies following settings:

- **Setup**$(1^\lambda)$ : The setup algorithm computes $N = pq$ where $p$ and $q$ is prime $\phi(N) = (p-1)(q-1)$ and chooses prime $e$ where $|e| = O(\lambda)$ and $d$ such that $d \times e = 1 \pmod{\phi(N)}$. It creates an obfuscation of the program Full Domain Hash in Figure 1. The program acts as a random oracle type hash function. We refer to the program as $H(m, r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, s^{(0)}, s^{(1)}) \to \{0,1\}^w$.
  The setup algorithm sets $H$, $N$ and $e$ as a verification key, $vk$, and $d$ and the parameters in Full Domain Hash in Figure 1 as a signing key, $sk$.
- **Sign**$(sk,\ m \in \mathcal{M})$ : This algorithm chooses random integers, $(r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, s^{(0)}, s^{(1)}) \in \mathbb{Z}_N$ and computes
  $\sigma = H(m, r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, r_{1,1}^{(2)}, ..., r_{\ell,\ell}^{(2)}, s^{(1)})^d \bmod N$. It outputs $(\sigma, r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, r_{1,1}^{(2)}, ..., r_{\ell,\ell}^{(2)}, s^{(1)})$.
- **Verify**$(vk, m, r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, r_{1,1}^{(2)}, ..., r_{\ell,\ell}^{(2)}, s^{(1)}, \sigma)$ : This checks
  $\sigma^e = H(m, r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, r_{1,1}^{(2)}, ..., r_{\ell,\ell}^{(2)}, s^{(1)}) \bmod N$. If the condition is true, outputs 1, otherwise 0.

We use two hash functions which have following construction. Note that when we use them, they are contained in an indistinguishability obfuscator.

After this, we omit an expression of security parameter, $\lambda$.

**Full Domain Hash**

**Constants:** Collision resistance hash functions $h_i : \{0,1\}^* \to \{0,1\}^\ell$ $(i = 1, ..., \ell)$, PRF $F$ key $K$, PRF $F_0$ keys $K_{1,1}, ..., K_{\ell,\ell}$ and random values $v_1, ..., v_\ell \overset{U}{\leftarrow} \mathbb{Z}_N^*$.
**Inputs:** $m \in \mathcal{M}$, $r_{1,1}^{(0)}, ..., r_{\ell,\ell}^{(0)}, r_{1,1}^{(1)}, ..., r_{\ell,\ell}^{(1)}, r_{1,1}^{(2)}, ..., r_{\ell,\ell}^{(2)} \in \mathbb{Z}_N$ and $s^{(1)} \in \mathbb{Z}_N$.

1. Let $F_0(K_{i,k}; \cdot) : \mathbb{Z}_N \to \{0,1\}$ and $F(K; \cdot) : \{0,1\}^{\ell^2} \to \{0,1\}^n$, where $n = t\ell$, $F_0$ and $F$ be pseudorandom functions.
2. $$b_i^{(0)} = h_i(m) \oplus (F_0(K_{i,1}; r_{i,1}^{(0)}), ..., F_0(K_{i,\ell}; r_{i,\ell}^{(0)})) \quad (1 \le i \le \ell)$$
3. $(c_1^{(0)}, ..., c_\ell^{(0)}) = F(K; (b_1^{(0)}, ..., b_\ell^{(0)}))$
4. Repeat 2. and 3. for $r_{i,k}^{(1)}$ and compute $(c_1^{(1)}, ..., c_\ell^{(1)})$ $(1 \le i, k \le \ell)$.
5. Repeat 2. and 3. for $r_{i,k}^{(2)}$ and compute $s^{(0)}$ $(1 \le i, k \le \ell)$.
6. Compute $a_i = s^{(0)} \times c_i^{(0)} + s^{(1)} \times c_i^{(1)}$ $(1 \le i \le \ell)$.
7. Output $v_1^{a_1} \times v_2^{a_2} \times ... \times v_\ell^{a_\ell} \bmod N$

Figure 4: Full Domain Hash

**Full Domain Hash***

**Constants:** Collision resistance hash functions $h_i : \{0,1\}^* \to \{0,1\}^\ell$ ($i = 1, ..., \ell$), punctured PRF $F$ key $K_S$, punctured PRF $F_0$ keys $K_{1,1,S_{1,1}}, ..., K_{\ell,\ell,S_{\ell,\ell}}$, random values $v_1, ..., v_\ell \xleftarrow{U} \mathbb{Z}_N^*$, $R_1^{(0)}, ..., R_q^{(0)}, R_1^{(1)}, ..., R_q^{(1)}, R_1^{(2)}, ..., R_q^{(2)} \xleftarrow{U} \{0,1\}^{\ell^2}$, $S = \{R_1^{(0)}, ..., R_q^{(0)}, R_1^{(1)}, ..., R_q^{(1)}, R_1^{(2)}, ..., R_q^{(2)}\}$, $Q^{(0)}_{i,k,j,\beta}, Q^{(1)}_{i,k,j,\beta}, Q^{(2)}_{i,k,j,\beta} \xleftarrow{U} \mathbb{Z}_N$, $S_{i,k} = \{Q^{(0)}_{i,k,j,0}, Q^{(0)}_{i,k,j,1}, Q^{(1)}_{i,k,j,0}, Q^{(1)}_{i,k,j,1}, Q^{(2)}_{i,k,j,0}, Q^{(2)}_{i,k,j,1} \mid 1 \le j \le q\}$, where $F(K; R_j^{(\gamma)}) = F(K_S; R_j^{(\gamma)})$ ($\gamma \in \{0,1,2\}$) and $F_0(K_{i,k}; Q^{(\gamma)}_{i,k,j,\beta}) = F_0(K_{i,k,S_{i,k}}; Q^{(\gamma)}_{i,k,j,\beta})$ ($j = 1, ..., q$) ($\gamma \in \{0,1,2\}, \beta \in \{0,1\}$).

**Inputs:** $m \in \mathcal{M}$, $r^{(0)}_{1,1}$, ..., $r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}$, ..., $r^{(1)}_{\ell,\ell}, r^{(2)}_{1,1}$, ..., $r^{(2)}_{\ell,\ell} \in \mathbb{Z}_N$ and $s^{(1)} \in \mathbb{Z}_N$.

1. Let $F_0(K_{i,k,S_{i,k}}; \cdot) : \mathbb{Z}_N \to \{0,1\}$ and $F(K_S; \cdot) : \{0,1\}^{\ell^2} \to \{0,1\}^n$, where $n = t\ell$, $F_0$ and $F$ be punctured pseudorandom functions.
2. $$b_i^{(0)} = h_i(m) \oplus (F_0(K_{i,1,S_{i,1}}; r^{(0)}_{i,1}), ..., F_0(K_{i,\ell,S_{i,\ell}}; r^{(0)}_{i,\ell}))\ (1 \le i \le l)$$
3. $(c_1^{(0)}, ..., c_\ell^{(0)}) = F(K_S; (b_1^{(0)}, ..., b_\ell^{(0)}))$
4. Repeat 2. and 3. for $r^{(1)}_{i,k}$ and compute $(c_1^{(1)}, ..., c_\ell^{(1)})$ $(1 \le i, k \le \ell)$.
5. Repeat 2. and 3. for $r^{(2)}_{i,k}$ and compute $s^{(0)}$ $(1 \le i, k \le \ell)$.
6. Compute $a_i = s^{(0)} \times c_i^{(0)} + s^{(1)} \times c_i^{(1)}$ $(1 \le i \le \ell)$.
7. Output $v_1^{a_1} \times v_2^{a_2} \times ... \times v_\ell^{a_\ell} \bmod N$

Figure 5: Full Domain Hash*

**Full Domain Hash****

**Constants:** Collision resistance hash functions $h_i : \{0,1\}^* \to \{0,1\}^\ell$ ($i = 1, ..., \ell$), punctured PRF $F$ key $K_S$, punctured PRF $F_0$ keys $K_{1,1,S_{1,1}}, ..., K_{\ell,\ell,S_{\ell,\ell}}$, random values $v_1, ..., v_\ell \xleftarrow{U} \mathbb{Z}_N^*$, $X_1^{(0)}, ..., X_q^{(0)}, X_1^{(1)}, ..., X_q^{(1)}, X_1^{(2)}, ..., X_q^{(2)} \xleftarrow{U} \{0,1\}^n$, $R_1^{(0)}, ..., R_q^{(0)}, R_1^{(1)}, ..., R_q^{(1)}, R_1^{(2)}, ..., R_q^{(2)} \xleftarrow{U} \{0,1\}^{\ell^2}$, $S = \{R_1^{(0)}, ..., R_q^{(0)}, R_1^{(1)}, ..., R_q^{(1)}, R_1^{(2)}, ..., R_q^{(2)}\}$, $Q^{(0)}_{i,k,j,\beta}, Q^{(1)}_{i,k,j,\beta}, Q^{(2)}_{i,k,j,\beta} \xleftarrow{U} \mathbb{Z}_N$, $S_{i,k} = \{Q^{(0)}_{i,k,j,0}, Q^{(0)}_{i,k,j,1}, Q^{(1)}_{i,k,j,0}, Q^{(1)}_{i,k,j,1}, Q^{(2)}_{i,k,j,0}, Q^{(2)}_{i,k,j,1} \mid 1 \le j \le q\}$, where $X_j^{(\gamma)} = F(K_S; R_j^{(\gamma)})$ ($\gamma \in \{0,1,2\}$) and $\beta = F_0(K_{i,k,S_{i,k}}; Q^{(\gamma)}_{i,k,j,\beta})$ ($j = 1, ..., q$) ($\gamma \in \{0,1,2\}, \beta \in \{0,1\}$).

**Inputs:** $m \in \mathcal{M}$, $r^{(0)}_{1,1}$, ..., $r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}$, ..., $r^{(1)}_{\ell,\ell}, r^{(2)}_{1,1}$, ..., $r^{(2)}_{\ell,\ell} \in \mathbb{Z}_N$ and $s^{(1)} \in \mathbb{Z}_N$.

1. Let $F_0(K_{i,k,S_{i,k}}; \cdot) : \mathbb{Z}_N \to \{0,1\}$ and $F(K_S; \cdot) : \{0,1\}^{\ell^2} \to \{0,1\}^n$, where $n = t\ell$, $F_0$ and $F$ be punctured pseudorandom functions.
2. $$b_i^{(0)} = h_i(m) \oplus (F_0(K_{i,1,S_{i,1}}, r^{(0)}_{i,1}), ..., F_0(K_{i,\ell,S_{i,\ell}}, r^{(0)}_{i,\ell}))\ (1 \le i \le l)$$
3. $(c_1^{(0)}, ..., c_\ell^{(0)}) = F(K_S; (b_1^{(0)}, ..., b_\ell^{(0)}))$
4. Repeat 2. and 3. for $r^{(1)}_{i,k}$ and compute $(c_1^{(1)}, ..., c_\ell^{(1)})$ $(1 \le i, k \le \ell)$.
5. Repeat 2. and 3. for $r^{(2)}_{i,k}$ and compute $s^{(0)}$ $(1 \le i, k \le \ell)$.
6. Compute $a_i = s^{(0)} \times c_i^{(0)} + s^{(1)} \times c_i^{(1)}$ $(1 \le i \le \ell)$.
7. Output $v_1^{a_1} \times v_2^{a_2} \times ... \times v_\ell^{a_\ell} \bmod N$

Figure 6: Full Domain Hash**

### B.2 Proof of Security

EUF-CMA の方式と同じように証明を行う。特に違うところは最後の補題で、sEUF のチェックを行っているところである。

**Theorem B.1.** *If our obfuscation scheme is indistinguishably secure, punctured PRFs $F$ and $F_0$ are secure and the RSA assumption holds, our signature scheme is strongly existentially unforgeable against adaptively chosen message attack (sEUF-CMA).*

- $\mathrm{Hyb}_0$ : In the first hybrid, following EUF-CMA game is played.

1. The challenger runs the setup algorithm. It chooses $e$ as a random chosen prime between 1 and $\phi(N)$ such that $\gcd(\phi(N), e) = 1$ and $|e| = O(\lambda)$.
2. The attacker receives the verification key.
3. $H$ is created as an obfuscated program of Full Domain Hash in Figure 1.
4. The attacker queries the signing oracle at most $q$ times on messages $m_1, ..., m_q$. In the $j$-th query, the challenger receives $m_j$ and chooses $r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, r^{(2)}_{1,1}, ..., r^{(2)}_{\ell,\ell}, s^{(1)}$ uniformly. It computes $\sigma_j = H(m_j, r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, r^{(2)}_{1,1}, ..., r^{(2)}_{\ell,\ell}, s^{(1)})^d \bmod N$. The attacker receives $(\sigma_j, r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, r^{(2)}_{1,1}, ..., r^{(2)}_{\ell,\ell}, s^{(1)})$.
5. The adversary sends $(m^*, \sigma^*, r^{(0)*}_{1,1}, ..., r^{(0)*}_{\ell,\ell}, r^{(1)*}_{1,1}, ..., r^{(1)*}_{\ell,\ell}, r^{(2)*}_{1,1}, ..., r^{(2)*}_{\ell,\ell}, s^{(1)*})$ as a challenge to the challenger. If $\mathrm{Verify}(m^*, \sigma^*, r^{(0)*}_{1,1}, ..., r^{(0)*}_{\ell,\ell}, r^{(1)*}_{1,1}, ..., r^{(1)*}_{\ell,\ell}, r^{(2)*}_{1,1}, ..., r^{(2)*}_{\ell,\ell}, s^{(1)})$ outputs 1, the adversary wins, otherwise it loses.

- $\mathrm{Hyb}_1$ : This game is equivalent to $\mathrm{Hyb}_0$ except for 3.. We use an obfuscated program of Full Domain Hash* in Figure 2 instead of that of Full Domain Hash in Figure 1.

- $\mathrm{Hyb}_2$ : This game is equivalent to $\mathrm{Hyb}_1$ except for 3.. We use an obfuscated program of Full Domain Hash** in Figure 3 instead of that of Full Domain Hash* in Figure 2.

- $\mathrm{Hyb}_3$ : The last game has following process.

  1. The challenger runs the setup algorithm. It chooses $e$ as a random chosen prime between 1 and $\phi(N)$ such that $\gcd(\phi(N), e) = 1$ and $|e| = O(\lambda)$.
  2. The attacker receives the verification key.
  3. Set punctured PRF $F(K_S; R^{(\gamma)}_j)$ and $F_0(K_{i,k,S_{i,k}}; Q^{(\gamma)}_{i,k,j,\beta})$ $(\gamma \in \{0, 1\})$ to the same ones in Figure 3.
  4. Compute $(v_1, ..., v_\ell)$ with the following process. $(v_1, ..., v_\ell)$ are integers used in the hash function.
     - Each $(\alpha_1$, ..., $\alpha_\ell)$ is chosen as a random integer in $\{0, 1\}^{3|N|}$ such that $\gcd(e, \alpha_i) = 1$.
     - $g$ is chosen as a random integer.
     - Compute $v_1 = g^{\alpha_1} \bmod N$, $v_2 = g^{\alpha_2} \bmod N$, ..., $v_\ell = g^{\alpha_\ell} \bmod N$.
  5. $H$ is created as an obfuscated program of Full Domain Hash** in Figure 3 by using values specified 3. and 4..
  6. The attacker queries the signing oracle at most $q$ times on messages $m_1, ..., m_q$. In the $j$-th query, the challenger receives $m_j$. $R^{(0)}_j = (b^{(0)}_1, ..., b^{(0)}_\ell)$, $R^{(1)}_j = (b^{(1)}_1, ..., b^{(1)}_\ell)$ and $R^{(2)}_j = (b^{(2)}_1, ..., b^{(2)}_\ell)$.
     - Choose each $r^{(0)}_{i,k} = Q^{(0)}_{i,k,j,0}$ or $r^{(0)}_{i,k} = Q^{(0)}_{i,k,j,1}$ such that $(F_0(K_{i,1,S_{i,1}}; r^{(0)}_{i,1})$,..., $F_0(K_{i,\ell,S_{i,\ell}}, r^{(0)}_{i,\ell})) = b^{(0)}_i \oplus h_i(m)$.
     - Choose each $r^{(1)}_{i,k}$ with the same way for $Q^{(1)}_{i,k,j,\beta}$.
     - Choose each $r^{(2)}_{i,k}$ with the same way for $Q^{(2)}_{i,k,j,\beta}$.
     - $(c^{(0)}_1, ..., c^{(0)}_\ell) = F(K_S; (b^{(0)}_1, ..., b^{(0)}_\ell))$, $(c^{(1)}_1, ..., c^{(1)}_\ell) =$ $F(K_S; (b^{(1)}_1, ..., b^{(1)}_\ell))$ and $s^{(0)} = F(K_S; (b^{(2)}_1, ..., b^{(2)}_\ell)$
     - Randomly select $s^{(1)} \in \mathbb{Z}_N$ such that $e \mid s^{(0)}\Sigma\alpha_i c^{(0)}_i + s^{(1)}\Sigma\alpha_i c^{(1)}_i$.
     - Computes $\sigma_j = g^{(s^{(0)}\Sigma\alpha_i c^{(0)}_i + s^{(1)}\Sigma\alpha_i c^{(1)}_i)/e} \bmod N$.
     - The attacker receives $(\sigma_j, r^{(0)}_{1,1}, ..., r^{(0)}_{\ell,\ell}, r^{(1)}_{1,1}, ..., r^{(1)}_{\ell,\ell}, s^{(0)}, s^{(1)})$.
  7. The adversary sends $(m^*, \sigma^*, r^{(0)*}_{1,1}, ..., r^{(0)*}_{\ell,\ell}, r^{(1)*}_{1,1}, ..., r^{(1)*}_{\ell,\ell}, r^{(2)*}_{1,1}$, $..., r^{(2)*}_{\ell,\ell}, s^{(1)*})$ as a challenge to the challenger. If $\mathrm{Verify}(m^*, \sigma^*, r^{(0)*}_{1,1}, ..., r^{(0)*}_{\ell,\ell}, r^{(1)*}_{1,1}, ..., r^{(1)*}_{\ell,\ell}$, $r^{(2)*}_{1,1}, ..., r^{(2)*}_{\ell,\ell}, s^{(0)*}, s^{(1)*})$ outputs 1, the adversary wins, otherwise it loses.

When the following three lemma is proved, Theorem3.1 is proved.

**Lemma B.1.** *If an indistinguishability obfuscator is secure, the advantage of the adversary in Hyb$_0$ is close to that in Hyb$_1$.*

*Proof.* This proof is the same as that in section 3. ☐

**Lemma B.2.** *If the punctured pseudorandom functions $F$ and $F_0$ are secure, the advantage of the adversary in Hyb$_1$ is close to that in Hyb$_2$.*

*Proof.* This proof is the same as that in section 3. ☐

**Lemma B.3.** *The advantage of the adversary in Hyb$_2$ is close to that in Hyb$_3$.*

*Proof.* Firstly, we prove that the distribution of $v_i \in \mathbb{Z}^*_N$ $(1 \leq i \leq l)$ in $\mathrm{Hyb}_2$ is statistically close to that in $\mathrm{Hyb}_3$. Random variable $v_i$ in $\mathrm{Hyb}_2$ is uniformly distributed in $Z^*_N$. In contrast, $v_i$ in $\mathrm{Hyb}_3$ is computed by using $\alpha_i$ such that $\gcd(e, \alpha_i) = 1$. Namely, when we compute $v_i = g^{\alpha_i \bmod \phi(N)}$, we exclude $\alpha_i$ such that $\alpha_i = (\alpha_i \bmod \phi(N)) + E_i\phi(N) \equiv 0 \pmod{e}$. We choose $E_i$ uniformly. In order to satisfy $\alpha_i \not\equiv 0 \pmod{e}$, $\alpha_i \bmod \phi(N)$ is chosen as $\alpha_i \bmod \phi(N) \not\equiv -E_i\phi(N) \pmod{e}$. $E_i\phi(N) \bmod e$, that is the constrained condition of $\alpha_i \bmod \phi(N)$, is (almost) uniformly distributed, because $\gcd(e, \phi(N)) = 1$. Hence, $\alpha_i \bmod \phi(N)$ is (almost) uniformly distributed. Since

$v_i$ is determined by $\alpha_i \bmod \phi(N)$, the distribution of $v_i$ in Hyb$_2$ is statistically close to that in Hyb$_3$.

Secondly, we consider $s^{(0)}$ and $s^{(1)}$ when the adversary sends $j$-th query in the hybrid game. A variable $s^{(1)}$ are chosen randomly in Hyb$_2$. On the other hand, a variable $s^{(1)}$ are chosen as $e \mid s^{(0)}\Sigma\alpha_i c_i^{(0)} + s^{(1)}\Sigma\alpha_i c_i^{(1)}$ in Hyb$_3$. Namely, $s^{(0)}\Sigma\alpha_i c_i^{(0)} + s^{(1)}\Sigma\alpha_i c_i^{(1)} \equiv 0 \pmod e$. We recall that a variable $s^{(0)}$ is computed as a uniformly distributed value. Then, a variable $s^{(1)}$ should satisfy the following relation with $s^{(0)}$: $s^{(1)}\Sigma\alpha_i c_i^{(1)} \equiv s^{(0)}\Sigma\alpha_i c_i^{(0)} \pmod e$. That is, $s^{(1)}(\Sigma(\alpha_i \bmod \phi(N))c_i^{(1)} + \Sigma E_i\phi(N)c_i^{(1)}) \equiv s^{(0)}(\Sigma(\alpha_i \bmod \phi(N))c_i^{(0)} + \Sigma E_i\phi(N)c_i^{(0)}) \pmod e$. The values of $\{c_i^{(0)}\}$ and $\{c_i^{(1)}\}$ are unknown to the adversary and uniformly distributed, because they are the output of the punctured PRF at punctured points. Though the adversary can get $g^{s^{(0)}\Sigma\alpha_i c_i^{(1)} \bmod \phi(N)}$, since the adversary can choose $s^{(1)}$ as he wants and if $s^{(1)} = 0$, the output of the hash function is $g^{s^{(0)}\Sigma\alpha_i c_i^{(0)} \bmod \phi(N)}$, the adversary cannot get information about $\{c_i^{(0)}\}$, since $s^{(0)}$ is a uniformly distributed value. Then the values of $\{c_i^{(0)}\}$ and $\{c_i^{(1)}\}$ still have the freedom of randomness of $2\ell$ variables. In addition, the value of $E_i$ is unknown to the adversary and distributed in a large space ($|E_i| \simeq 2|N|$). Therefore, $s^{(1)}$ is uniformly and independently distributed from $s^{(0)}$. Hence, the distribution of $s^{(1)}$ in Hyb$_2$ and those of Hyb$_3$ is equivalent.

Thus, this lemma follows.

□

**Lemma B.4.** *If the RSA assumption holds, the advantage of an PPT adversary in Hyb$_3$ is negligible.*

*Proof.* For this proof, we begin consider two probabilistic polynomial-time algorithms $\mathcal{A}$ and $\mathcal{B}$. Let $\mathcal{B}$ be the adversary for the RSA assumption, and $\mathcal{A}$ be that for our signature scheme. $\mathcal{B}$ uses $\mathcal{A}$ in itself to break the RSA assumption.

$\mathcal{B}$ receives as input an RSA challenge $(N, e, y)$ where $N = pq$ such that $p$ and $q$ are prime, prime[4] $e \in [1, \phi(N)]$ such that $\phi(N) = (p-1)(q-1)$ and $\gcd(\phi(N), e) = 1$, and $y \in Z_N^*$.

$\mathcal{B}$ plays a role of the challenger in Hyb$_3$ such that $g = y$ with the adversary $\mathcal{A}$.

We assume that the adversary $\mathcal{A}$ outputs a valid signature $(m^*, \sigma^*, r_{1,1}^{(0)*}, ..., r_{\ell,\ell}^{(0)*}, r_{1,1}^{(1)*}, ..., r_{\ell,\ell}^{(1)*}, r_{1,1}^{(2)*}, ..., r_{\ell,\ell}^{(2)*}, s^{(1)*})$. $\mathcal{B}$ can compute $c_i^{(0)*}$, $c_i^{(1)*}$, $s^{(0)*}$ and $s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$. $\mathcal{B}$ checks whether $e \nmid s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$. If $e \nmid s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$, $\mathcal{B}$ breaks the RSA assumption. We consider its probability.

There are two cases, where in case (1), $\{c_i^{(0)*}\}$ and $\{c_i^{(1)*}\}$ are the outputs of punctured PRF $F$ at punctured points and otherwise in case (2). We now show that case (1) occurs in negligible probability. In sEUF-CMA game, an adversary can adopt $m_j$ used in a query phase as $m^*$ to make a forgery, $\sigma^*$, that is different from $\sigma_j$. That is, when an adversary compute $b_i^{(0)}$, it has to choose an another randomness, at least 1 variable, and we assume that variable is $r_{i,k}^{(0)}$. Since the punctured points of punctured PRF $F_0$ is uniformly distributed in $\mathbb{Z}_N$ and the distribution of its output at punctured points is also uniform, the probability that the adversary chooses $r_{i,k}^{(0)}$ such that $(F_0(K_{i,1,S_{i,k}}; r_{i,k}^{(0)}) = [b_i]_k^{(0)} \oplus [h_i(m^*)]_k$ is $\frac{1}{2}$. Hence, the adversary can get the output of punctured PRF $F$ at punctured points in negligible probability, at most $\frac{1}{2^\ell}$.

Then, we show that $e \mid s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$ with negligible probability in case (2). The point here is that the adversary has no idea of $E_i$ such that $\alpha_i = \alpha_i \bmod \phi(N)) + E_i\phi(N)$. The exponent of the output of the hash function is $s^{(0)*}(\Sigma(\alpha_i \bmod \phi(N))c_i^{(0)} + \Sigma E_i\phi(N)c_i^{(0)}) + s^{(1)*}(\Sigma(\alpha_i \bmod \phi(N))c_i^{(1)} + \Sigma E_i\phi(N)c_i^{(1)})$. Random value $E_i$ is uniformly distributed in space of which the size similarly equals to $2|N|$. Hence, the probability that $e \mid s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$ is close to $\frac{1}{e}$, because $\gcd(\phi(N), e) = 1$. Since the size of $e$ is $O(\lambda)$, the adversary can send a forged signature with negligible probability such that $e \mid s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$. Hence, $\gcd(e, s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}) = 1$ with overwhelming probability.

Let $A$ be $s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*}$. Note that $\gcd(e, A) = 1$.

$$\begin{aligned}\sigma^* &= y^{(s^{(0)*}\Sigma\alpha_i c_i^{(0)*} + s^{(1)*}\Sigma\alpha_i c_i^{(1)*})\frac{1}{e}} \\ &= y^{A\cdot\frac{1}{e}} \\ &= x^A \pmod N,\end{aligned}$$

where $x = y^{\frac{1}{e}} \bmod N$.

Since $\gcd(e, A) = 1$, we can compute $\gamma$, $\delta$ such that

$$A\gamma + e\delta = 1$$

with the extended euclidean algorithm.

$$\begin{aligned}\sigma^{*\gamma} \times y^\delta &= (x^A)^\gamma \times (x^e)^\delta \\ &= x \pmod N\end{aligned}$$

$\mathcal{B}$ can compute $x = y^{\frac{1}{e}} \bmod N$, which is the answer for the RSA challenge. This contradicts the RSA assumption.

This lemma follows.

□

[4] We assume that $e$ is a prime for simplicity of description, but it is easy to relax the primarity.