

# The Cryptographic Layer of Biometric Authentication

Keng-Yu Chen

**Supervisor:** Serge Vaudenay  
LASEC, EPFL

February 12, 2025

## Abstract

In this project, we focus on the cryptographic layer which is added on the top of biometric authentication for privacy reasons. We first formalize a biometric authentication scheme and propose security models for two security properties of interest: *unforgeability* and *indistinguishability*. Unforgeability refers to an adversary’s ability to impersonate a user, while indistinguishability evaluates the server’s knowledge of users’ biometrics, related to privacy preservation. Subsequently, we analyze two existing instantiations of biometric authentication built on two cryptographic primitives: function-hiding inner product functional encryption and relational hash. Our results demonstrate conditions under which these schemes achieve security within our security model, and we propose a simple way to strengthen the system based on functional encryption by adding a digital signature in the cryptographic layer.

## 1 Introduction

Biometric authentication offers an error-tolerant approach to user verification. Despite its convenience, unlike traditional authentication methods, servers have to verify users’ identities by comparing the similarity of enrolled and probed data instead of their equivalence. An authentication method based on comparing hashes of two templates thus fails. Additionally, unlike a user-defined password, biometrics reveal sensitive personal information and cannot be changed, raising significant privacy concerns. Furthermore, the inherent nature of biometrics data can introduce a non-negligible false positive rate. These issues make designing a biometric authentication scheme and analyzing its security challenging and highlight the importance of a rigorous study in this domain.

**Previous Work** Previous works have demonstrated several potential cryptographic primitives that can be utilized to instantiate a biometric authentication scheme, such

as function-hiding inner product functional encryption (fh-IPFE) [Kim+16; Lee+18; Che+21; Cac+22; EM23], homomorphic encryption [Yas+13; PM21], fuzzy extractor [Boy04; Li+17], oblivious transfer [BCP12], relational hash [MR14], etc. Some of them provide non-interactive protocols in the sense that only the clients transmit enrollment and probe messages to the server before the server decides the authentication results. On the other hand, an interactive protocol allows the server to send hints or challenges to the clients during the authentication process.

**Explanation** [A brief explanation of this work]

**Contribution** In this project, we present the following contributions:

- We provide a new general framework for analyzing a non-interactive biometric authentication scheme. Our framework formalizes a biometric authentication scheme by splitting it into two layers: the biometric layer and the cryptographic layer. The biometric layer accounts for collecting biometric data from users, comparing the closeness of enrolled and probed biometric templates, and deciding the authentication result. The cryptographic layer, on the other hand, is to protect users' privacy and strengthen the security of the scheme.
- We list two security games to model two security notions that we consider relevant to a biometric authentication scheme: the unforgeability (UF) game and the indistinguishability (IND) game. The UF game models an adversary's ability to impersonate the user by offering a (possibly invalid) probe message that can result in a successful authentication, which is similar to the unforgeability notion in [MR14] and the malicious adversary in [EM23]. However, we consider several options for the adversary to add more flexibility to our security model.

The IND game evaluates the server's knowledge of users' biometrics, where we model the adversary's ability to recognize the biometrics. Previous works [MR14; Lee+18; Che+21; EM23] consider a similar security notion by considering an adversary who has enrollment and probe messages. Compared to them, our model also provides the adversary with oracles to users' biometric templates to take biometric distributions into consideration and increase practicality.

- We analyze the UF and IND security of existing instantiations of biometric authentication schemes from previous works. Our results demonstrate necessary and sufficient conditions and provide transformations for these instantiations to achieve our desired security.

**Structure of the Project** In Section 2, we formally define a biometric authentication scheme, including the biometric layer and cryptographic layer. In Section 3, we introduce the unforgeability (UF) game and the indistinguishability (IND) game. In Section 4 and 5, we recall two instantiations using function-hiding inner-product functional encryption and relational hash, respectively, and provide analyses of the UF and IND security of them.

**Notation** In this project, we assume

- $\lambda$  is the security parameter.
- $[m]$  denotes the set of integers  $\{1, 2, \dots, m\}$ .
- $\mathbb{Z}_q$  is the finite field modulo a prime number  $q$ .
- A function  $f(n)$  is called *negligible* iff for any integer  $c$ ,  $f(n) < \frac{1}{n^c}$  for all sufficiently large  $n$ . We write it as  $f(n) = \text{negl}$ , and we may also use  $\text{negl}$  to represent an arbitrary negligible function.
- $\text{poly}$  is the class of polynomial functions. We may also use  $\text{poly}$  to represent an arbitrary polynomial function.
- We write sampling a value  $r$  from a distribution  $\mathcal{D}$  as  $r \leftarrow^s \mathcal{D}$ . If  $S$  is a finite set, then  $r \leftarrow^s S$  means sampling  $r$  uniformly from  $S$ .
- The distribution  $\mathcal{D}^t$  denotes  $t$  identical and independent distributions of  $\mathcal{D}$ .
- A PPT algorithm denotes a probabilistic polynomial time algorithm. Unless otherwise specified, all algorithms run in PPT.

## 2 Formalization

### 2.1 Biometric Authentication Scheme

In this section, we formally define a biometric authentication scheme. For this, we first define how we simulate biometric distributions of users.

Assume the existence of a family  $\mathbb{B}$  of biometric distributions that are efficiently samplable. We have the following interfaces for all algorithms to interact with  $\mathbb{B}$ .

- **BioSamp()**: Generate a random distribution  $\mathcal{B}$  of  $\mathbb{B}$ . By this we mean providing either parameters of an efficiently samplable distribution or a PPT algorithm as the sampler. For simplicity, we write  $\mathcal{B} \leftarrow \text{BioSamp}()$  as  $\mathcal{B} \leftarrow^s \mathbb{B}$ .
- **BioDelete( $\mathcal{B}$ )**: Delete  $\mathcal{B}$  from  $\mathbb{B}$ . Consequently, no further access to **BioSamp** can derive  $\mathcal{B}$ . For simplicity, we write **BioDelete**( $\mathcal{B}$ ) as  $\mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$ .
- **TempSamp( $\mathcal{B}$ )**: Let  $\mathcal{B}$  be a biometric distribution in  $\mathbb{B}$ . This algorithm samples a biometric template from  $\mathcal{B}$ . For simplicity, we write  $\mathbf{b} \leftarrow \text{TempSamp}(\mathcal{B})$  as  $\mathbf{b} \leftarrow^s \mathcal{B}$ .

**Definition 1** (Biometric Authentication Scheme). A *biometric authentication scheme*  $\Pi$  associated with a family  $\mathbb{B}$  of biometric distributions is composed of the following algorithms.

- **getEnroll $^{\mathcal{O}_{\mathbb{B}}}()$   $\rightarrow \mathbf{b}$** : Given an oracle  $\mathcal{O}_{\mathbb{B}}$ , which samples biometric data from a distribution  $\mathcal{B} \in \mathbb{B}$ , it outputs a biometric template  $\mathbf{b}$  for enrollment.

- $\text{getProbe}^{\mathcal{O}_{\mathcal{B}}}() \rightarrow \mathbf{b}'$ : Given an oracle  $\mathcal{O}_{\mathcal{B}}$ , which samples biometric data from a distribution  $\mathcal{B} \in \mathbb{B}$ , it outputs a biometric template  $\mathbf{b}'$  for probe.
- $\text{BioCompare}(\mathbf{b}, \mathbf{b}') \rightarrow s$ : Given two biometric templates  $\mathbf{b}$  and  $\mathbf{b}'$ , it outputs a score  $s$ .
- $\text{Verify}(s) \rightarrow r \in \{0, 1\}$ : It is a deterministic algorithm that reads the comparison score  $s$  and determines whether this is a successful authentication ( $r = 1$ ) or not ( $r = 0$ ).

We also call these algorithms the *biometric layer* of  $\Pi$ . We will add a *cryptographic layer* on top of it in Section 2.2

Given an authentication scheme  $\Pi$ , we can consider its true positive rate and false positive rate.

**Definition 2** (True Positive Rate). For a biometric distribution  $\mathcal{B} \in \mathbb{B}$  and  $\mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ , define the *true positive rate* TP.

$$\text{TP}(\mathcal{B}, \mathbf{b}) := \Pr[\mathbf{b}' \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}}}() : \text{Verify}(\text{BioCompare}(\mathbf{b}, \mathbf{b}')) = 1]$$

$$\begin{aligned} \text{TP}(\mathcal{B}) &:= \Pr \left[ \begin{array}{l} \mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \\ \mathbf{b}' \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}}}() \end{array} : \text{Verify}(\text{BioCompare}(\mathbf{b}, \mathbf{b}')) = 1 \right] \\ &= \mathbb{E}_{\mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()} [\text{TP}(\mathcal{B}, \mathbf{b})] \end{aligned}$$

$$\begin{aligned} \text{TP} &:= \Pr \left[ \begin{array}{l} \mathcal{B} \leftarrow \mathbb{B} \\ \mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \\ \mathbf{b}' \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}}}() \end{array} : \text{Verify}(\text{BioCompare}(\mathbf{b}, \mathbf{b}')) = 1 \right] \\ &= \mathbb{E}_{\mathcal{B} \leftarrow \mathbb{B}} [\text{TP}(\mathcal{B})] \end{aligned}$$

**Definition 3** (False Positive Rate). For a biometric distribution  $\mathcal{B} \in \mathbb{B}$ ,  $\mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$  and  $\mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ , define the *false positive rate* FP.

$$\begin{aligned}
\text{FP}(\mathbf{b}) &:= \Pr \left[ \begin{array}{l} \mathcal{B}' \leftarrow_{\$} \mathbb{B} \\ \mathbf{b}' \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}'}}() \end{array} : \text{Verify}(\text{BioCompare}(\mathbf{b}, \mathbf{b}')) = 1 \right] \\
\text{FP}(\mathcal{B}) &:= \Pr \left[ \begin{array}{l} \mathcal{B}' \leftarrow_{\$} \mathbb{B} \\ \mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \\ \mathbf{b}' \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}'}}() \end{array} : \text{Verify}(\text{BioCompare}(\mathbf{b}, \mathbf{b}')) = 1 \right] \\
&= \mathbb{E}_{\mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()} [\text{FP}(\mathbf{b})] \\
\text{FP} &:= \Pr \left[ \begin{array}{l} \mathcal{B} \leftarrow_{\$} \mathbb{B}, \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}, \mathcal{B}' \leftarrow_{\$} \mathbb{B} \\ \mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \\ \mathbf{b}' \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}'}}() \end{array} : \text{Verify}(\text{BioCompare}(\mathbf{b}, \mathbf{b}')) = 1 \right] \\
&= \mathbb{E}_{\mathcal{B} \leftarrow_{\$} \mathbb{B}} [\text{FP}(\mathcal{B})]
\end{aligned}$$

Ideally, we hope TP to be 1, and  $\text{FP}(\mathcal{B})$  to be negligible for any  $\mathcal{B} \in \mathbb{B}$ . However, due to the inherent nature of biometrics, there might be a nonzero false negative rate  $1 - \text{TP} > 0$  and a non-negligible  $\text{FP}(\mathcal{B})$ . Our security model and analysis also take these possibilities into consideration.

## 2.2 Cryptographic Layer

In this work, we add a cryptographic layer on top of  $\Pi$  to protect privacy of users. The cryptographic layer includes the following algorithms.

- $\text{Setup}(1^\lambda) \rightarrow \text{esk}, \text{psk}, \text{csk}$ : It outputs the enrollment secret key  $\text{esk}$ , probe secret key  $\text{psk}$ , and compare secret key  $\text{csk}$ .
- $\text{Enroll}(\text{esk}, \mathbf{b}) \rightarrow \mathbf{c}_x$ : On input a biometric template  $\mathbf{b}$ , it encodes it into a vector  $\mathbf{x}$  and outputs the enrollment message  $\mathbf{c}_x$ .
- $\text{Probe}(\text{psk}, \mathbf{b}') \rightarrow \mathbf{c}_y$ : On input a biometric template  $\mathbf{b}'$ , it encodes it into a vector  $\mathbf{y}$  and outputs the probe message  $\mathbf{c}_y$ .
- $\text{Compare}(\text{csk}, \mathbf{c}_x, \mathbf{c}_y) \rightarrow s$ : It compares the enrollment message  $\mathbf{c}_x$  and probe message  $\mathbf{c}_y$  and outputs a score  $s$ .

**Correctness:** An authentication scheme  $\Pi$  is *correct* if for any biometric distributions  $\mathcal{B}$  and  $\mathcal{B}'$ , let  $\text{esk}, \text{psk}, \text{csk} \leftarrow \text{Setup}(1^\lambda)$ ,  $\mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ ,  $\mathbf{b}' \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}'}}()$ ,  $\mathbf{c}_x \leftarrow \text{Enroll}(\text{esk}, \mathbf{b})$ ,  $\mathbf{c}_y \leftarrow \text{Probe}(\text{psk}, \mathbf{b}')$ . Then

$$\Pr [\text{Compare}(\text{csk}, \mathbf{c}_x, \mathbf{c}_y) = \text{BioCompare}(\mathbf{b}, \mathbf{b}')] = 1 - \text{negl}.$$

In a real-world biometric system, these algorithms may be run by different parties such as a biometric scanner, a user's secure hardware, a trusted authority that issues keys, and the server.

We provide two instantiations of a biometric authentication scheme with the cryptographic layer in Sections 4.1 and 5.1.

### 3 Security Games

In this section, we discuss two security notions of a biometric authentication scheme: *unforgeability* and *indistinguishability*.

#### 3.1 Unforgeability

To describe the unforgeability of an authentication scheme, we model the ability of an adversary who tries to impersonate a user. The adversary  $\mathcal{A}$  is given auxiliary information **option** that depends on our threat model and tries to find a valid probe message  $\tilde{\mathbf{z}}$ . The whole game  $\text{UF}_{\Pi, \mathbb{B}, \text{option}}(\mathcal{A})$  is defined in Algorithm 1.

---

**Algorithm 1**  $\text{UF}_{\Pi, \mathbb{B}, \text{option}}(\mathcal{A})$ 


---

```

1:  $\mathcal{B} \leftarrow \$ \mathbb{B}, \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$ 
2:  $\text{esk}, \text{psk}, \text{csk} \leftarrow \text{Setup}(1^\lambda)$ 
3:  $\mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ 
4:  $\mathbf{c}_x \leftarrow \text{Enroll}(\text{esk}, \mathbf{b})$ 
5:  $\tilde{\mathbf{z}} \leftarrow \mathcal{A}(\text{option})$ 
6: if  $\tilde{\mathbf{z}}$  is equal to any output of  $\mathcal{O}_{\text{Probe}}$  then
7:   return 0
8: end if
9:  $s \leftarrow \text{Compare}(\text{csk}, \mathbf{c}_x, \tilde{\mathbf{z}})$ 
10: return  $\text{Verify}(s)$ 

```

---

The auxiliary information **option** can be nothing or include  $\mathbf{c}_x, \text{esk}, \text{psk}, \text{csk}$  or the following oracles:

- $\mathcal{O}_{\mathcal{B}}$ : It outputs a biometric sample  $\mathbf{b} \leftarrow \$ \mathcal{B}$ . This oracle and  $\text{psk}$  should not be given at the same time; otherwise, there exists a trivial attack with a winning rate  $\text{TP}$  by returning  $\text{Probe}(\text{psk}, \text{getProbe}^{\mathcal{O}_{\mathcal{B}}}())$ .
- $\mathcal{O}_{\text{Enroll}}(\text{esk}, \cdot)$ : On input  $\mathbf{b}'$ , it outputs the enrollment message  $\text{Enroll}(\text{esk}, \mathbf{b}')$ .
- $\mathcal{O}_{\text{Probe}}(\text{psk}, \cdot)$ : On input  $\mathbf{b}'$ , it outputs the probe message  $\text{Probe}(\text{psk}, \mathbf{b}')$ . If this oracle is given, we require the adversary to return a  $\tilde{\mathbf{z}}$  that is not equal to any previous answer of  $\mathcal{O}_{\text{Probe}}$ .
- $\mathcal{O}_{\log}(\text{csk}, \mathbf{c}_x, \cdot)$ : On input  $\mathbf{b}'$ , it first computes  $\mathbf{c}_z \leftarrow \text{Probe}(\text{psk}, \mathbf{b}')$  and outputs  $\text{Verify}(\text{Compare}(\text{csk}, \mathbf{c}_x, \mathbf{c}_z))$ .
- $\mathcal{O}'_{\text{Enroll}}(\cdot)$ : On input  $\text{esk}'$ , it first samples  $\mathbf{b}' \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$  and outputs  $\text{Enroll}(\text{esk}', \mathbf{b}')$ . This oracle is only useful when **option** does not include  $\mathcal{O}_{\mathcal{B}}$ .
- $\mathcal{O}'_{\text{Probe}}(\cdot)$ : On input  $\text{psk}'$ , it first samples  $\mathbf{b}' \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$  and outputs  $\text{Probe}(\text{psk}', \mathbf{b}')$ . This oracle is only useful when **option** does not include  $\mathcal{O}_{\mathcal{B}}$ , and this oracle and  $\text{psk}$  should not be given at the same time; otherwise, there exists a trivial attack with a winning rate  $\text{TP}$  by returning  $\mathcal{O}'_{\text{Probe}}(\text{psk})$ .

The requirement that the adversary should return a  $\tilde{\mathbf{z}}$  that is not equal to any previous answer of  $\mathcal{O}_{\text{Probe}}$  is to prevent a trivial attack that leverages TP or FP when it is non-negligible. If **option** includes  $\mathcal{O}_{\mathcal{B}}$  and either **psk** or  $\mathcal{O}_{\text{Probe}}$ , the adversary can enjoy a winning rate TP. Therefore, we rule out the case that **option** includes both **psk** and  $\mathcal{O}_{\mathcal{B}}$ , and we forbid the adversary to return what  $\mathcal{O}_{\text{Probe}}$  returns. If **option** has only **psk** or  $\mathcal{O}_{\text{Probe}}$ , the UF adversary  $\mathcal{A}$  in Algorithm 2 can still enjoy a winning rate FP, if we place no restriction on the adversary's answer. Therefore, we only consider **psk** in **option** when FP is non-negligible, and we restrict the adversary's answer when  $\mathcal{O}_{\text{Probe}}$  is given.

---

**Algorithm 2**  $\mathcal{A}(\text{psk})$  ( or  $\mathcal{A}^{\mathcal{O}_{\text{Probe}}}$  )

---

```

1:  $\mathcal{B}' \leftarrow \mathbb{B}$ 
2:  $\mathbf{b}' \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}'}}()$ 
3:  $\mathbf{c}_y \leftarrow \text{Probe}(\text{psk}, \mathbf{b}')$        $\triangleright$  or  $\mathbf{c}_y \leftarrow \mathcal{O}_{\text{Probe}}(\mathbf{b}')$ 
4: return  $\mathbf{c}_y$ 

```

---

We define the advantage of an adversary  $\mathcal{A}$  in the  $\text{UF}_{\Pi, \mathbb{B}, \text{option}}$  game of a scheme  $\Pi$  associated with a family  $\mathbb{B}$  of distributions as

$$\text{Adv}_{\Pi, \mathbb{B}, \mathcal{A}, \text{option}}^{\text{UF}} := \Pr[\text{UF}_{\Pi, \mathbb{B}, \text{option}}(\mathcal{A}) \rightarrow 1]$$

An authentication scheme  $\Pi$  associated with a family  $\mathbb{B}$  of distributions is called *option-unforgeable* (option-UF) if for any PPT adversary  $\mathcal{A}$ ,

$$\text{Adv}_{\Pi, \mathbb{B}, \mathcal{A}, \text{option}}^{\text{UF}} = \text{negl}.$$

For the rest of this work, if the scheme  $\Pi$ , the family  $\mathbb{B}$  of distributions, and the auxiliary information **option** are clear from context, we omit the subscript and write the game as  $\text{UF}(\mathcal{A})$ . This abbreviation also holds for all other games.

**UF Security with Digital Signature** We note that we can achieve UF security by a similar approach in [EM23] with a digital signature scheme. Given any authentication scheme  $\Pi$  and an sEUF-CMA digital signature scheme  $\text{Sig} = (\text{Sig.KeyGen}, \text{Sig.Sign}, \text{Sig.Verify})$ , consider the following scheme  $\Pi'$ .

- **Setup'**( $1^\lambda$ ): Run  $(\text{esk}, \text{psk}, \text{csk}) \leftarrow \text{Setup}(1^\lambda)$  and  $(\text{sk}_{\text{Sig}}, \text{pk}_{\text{Sig}}) \leftarrow \text{Sig.KeyGen}(1^\lambda)$ . Output  $\text{esk}' \leftarrow \text{esk}$ ,  $\text{psk}' \leftarrow (\text{psk}, \text{sk}_{\text{Sig}})$ ,  $\text{csk}' \leftarrow \text{csk}$ .
- **Probe'**( $\text{psk}', \mathbf{b}'$ ): Run  $\mathbf{c}_y \leftarrow \text{Probe}(\text{psk}, \mathbf{b}')$  and  $\sigma \leftarrow \text{Sig.Sign}(\text{sk}_{\text{Sig}}, \mathbf{c}_y)$ . Output  $\mathbf{c}_y' \leftarrow (\mathbf{c}_y, \sigma)$ .
- **Compare'**( $\text{csk}, \mathbf{c}_x, \mathbf{c}_y'$ ): If  $\text{Sig.Verify}(\text{pk}_{\text{Sig}}, \mathbf{c}_y, \sigma) = 1$ , output  $\text{Compare}(\text{csk}, \mathbf{c}_x, \mathbf{c}_y)$ ; otherwise, output  $\perp$ .

An  $\text{UF}_{\text{option}}$  adversary has to forge a signature  $\sigma$  to win the game, so the scheme is option-UF for any **option** that does not include **psk**.

**Theorem 1.** *Let  $\text{option} = \{\mathbf{c}_x, \text{esk}, \text{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\text{Probe}}\}$ . For any authentication scheme  $\Pi$ ,  $\Pi'$  is option-UF.*

### 3.2 Indistinguishability

In the game of indistinguishability, we model the ability of an authentication server who tries to identify the user, which describes the privacy leakage of the scheme. The adversary  $\mathcal{A}$  is given oracles to two biometric distributions  $\mathcal{B}^{(0)}$  and  $\mathcal{B}^{(1)}$ , the comparison key  $\text{csk}$ , an enrollment message  $\mathbf{c}_x$ , and a list of  $t$  probe messages  $\{\mathbf{c}_y^{(i)}\}_{i=1}^t$ . It tries to guess from either  $\mathcal{B}^{(0)}$  or  $\mathcal{B}^{(1)}$  these messages are generated. The whole game is defined in Algorithm 3.

---

**Algorithm 3**  $\text{IND}_{\Pi, \mathbb{B}}(\mathcal{A})$ 


---

```

1:  $b \leftarrow_{\$} \{0, 1\}$ 
2:  $\mathcal{B}^{(0)} \leftarrow_{\$} \mathbb{B}, \quad \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}^{(0)}$ 
3:  $\mathcal{B}^{(1)} \leftarrow_{\$} \mathbb{B}, \quad \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}^{(1)}$ 
4:  $\text{esk}, \text{psk}, \text{csk} \leftarrow \text{Setup}(1^\lambda)$ 
5:  $\mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}^{(b)}}}()$ 
6:  $\mathbf{c}_x \leftarrow \text{Enroll}(\text{esk}, \mathbf{b})$ 
7: for  $i = 1$  to  $t$  do
8:    $\mathbf{b}'^{(i)} \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}^{(b)}}}()$ 
9:    $\mathbf{c}_y^{(i)} \leftarrow \text{Probe}(\text{psk}, \mathbf{b}'^{(i)})$ 
10: end for
11:  $\tilde{b} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{B}^{(0)}}, \mathcal{O}_{\mathcal{B}^{(1)}}}(\text{csk}, \mathbf{c}_x, \{\mathbf{c}_y^{(i)}\}_{i=1}^t)$ 
12: return  $1_{\tilde{b}=b}$ 

```

---

We define the advantage of an adversary  $\mathcal{A}$  in the IND game of a scheme  $\Pi$  associated with a family of distributions  $\mathbb{B}$  as

$$\text{Adv}_{\Pi, \mathbb{B}, \mathcal{A}}^{\text{IND}} := \left| \Pr[\text{IND}_{\Pi}(\mathcal{A}) \rightarrow 1] - \frac{1}{2} \right|.$$

An authentication scheme  $\Pi$  associated with a family  $\mathbb{B}$  of distributions is called *indistinguishable (IND)* if for any PPT adversary  $\mathcal{A}$ ,

$$\text{Adv}_{\Pi, \mathbb{B}, \mathcal{A}}^{\text{IND}} = \text{negl}.$$

**Necessity of IND Security** Consider the following authentication scheme for any biometric layer. Let  $\text{esk} = \text{psk} = \text{csk}$  be empty strings and

$$\begin{aligned} \text{Enroll}(\text{esk}, \mathbf{b}) &\rightarrow \mathbf{b}, & \text{Probe}(\text{psk}, \mathbf{b}') &\rightarrow \mathbf{b}' \\ \text{Compare}(\text{csk}, \mathbf{b}, \mathbf{b}') &= \text{BioCompare}(\mathbf{b}, \mathbf{b}') \end{aligned}$$

By the transformation using an sEUF-CMA digital signature scheme we introduced in Section 3.1, we can obtain an authentication scheme  $\Pi'$  that is **option**-UF for any **option** that does not include  $\text{psk}$ . However, the enrollment and probe messages leak biometric vectors  $\mathbf{b}$  and  $\mathbf{b}'$  and compromise privacy. Obviously, this scheme is not IND, and we use this example emphasize the necessity of the game of indistinguishability.



**IND Security for a Particular Biometric Layer** Let  $\text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}(), \text{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$  be such that

$$\text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \rightarrow \mathbf{b}_{\mathcal{B}}^* \oplus \mathcal{E}_{\text{Enroll}} \quad \text{and} \quad \text{getProbe}^{\mathcal{O}_{\mathcal{B}}}() \rightarrow \mathbf{b}_{\mathcal{B}}^* \oplus \mathcal{E}_{\text{Probe}}$$

where  $\mathbf{b}_{\mathcal{B}}^* \in \{0, 1\}^k$  is a fixed vector only dependent on  $\mathcal{B}$ , and  $\mathcal{E}_{\text{Enroll}}, \mathcal{E}_{\text{Probe}} \subseteq \{0, 1\}^k$  are some *error distributions* independent of  $\mathcal{B}$ . Let  $\text{BioCompare}(\mathbf{b}, \mathbf{b}') \rightarrow 1_{\text{HD}(\mathbf{b}, \mathbf{b}') \leq \tau}$ . Then

$$\text{TP} = \Pr[\text{HW}(\mathbf{b}_{\mathcal{B}}^* \oplus \mathcal{E}_{\text{Enroll}} \oplus \mathbf{b}_{\mathcal{B}}^* \oplus \mathcal{E}_{\text{Probe}}) \leq \tau] = \Pr[\text{HW}(\mathcal{E}_{\text{Enroll}} \oplus \mathcal{E}_{\text{Probe}}) \leq \tau]$$

We note that previous works such as [Boy04; MR14] model biometric template vectors in a similar way.

Now, for this biometric layer, we can construct a simple but IND secure authentication scheme  $\Pi$  with the following cryptographic layer.

- **Setup**( $1^\lambda$ ): Sample  $\mathbf{r} \leftarrow_{\$} \{0, 1\}^k$ . Output  $\text{esk} = \text{psk} \leftarrow \mathbf{r}$ ,  $\text{csk} \leftarrow \epsilon$ .
- **Enroll**( $\text{esk}, \mathbf{b}$ ): Output  $\mathbf{b} \oplus \mathbf{r}$ .
- **Probe**( $\text{psk}, \mathbf{b}'$ ): Output  $\mathbf{b}' \oplus \mathbf{r}$ .
- **Compare**( $\text{csk}, \mathbf{c}_x, \mathbf{c}_y$ ): If  $\text{HD}(\mathbf{c}_x, \mathbf{c}_y) \leq \tau$ , return 1; otherwise, return 0.

The correctness of  $\Pi$  holds by

$$\text{HD}(\mathbf{c}_x, \mathbf{c}_y) = \text{HW}(\mathbf{b} \oplus \mathbf{r} \oplus \mathbf{b}' \oplus \mathbf{r}) = \text{HW}(\mathbf{b} \oplus \mathbf{b}') = \text{BioCompare}(\mathbf{b}, \mathbf{b}').$$

**Theorem 2.** *The authentication scheme  $\Pi$  is IND.*

*Proof.* Let  $\mathbf{b}_0^*$  and  $\mathbf{b}_1^*$  be the fixed vectors of  $\mathcal{B}^{(0)}$  and  $\mathcal{B}^{(1)}$  in the IND game, respectively. For any  $\mathbf{v}, \mathbf{v}^{(1)}, \dots, \mathbf{v}^{(t)}$ ,

$$\begin{aligned} & \Pr[\mathbf{c}_x = \mathbf{v}, \mathbf{c}_y^{(1)} = \mathbf{v}^{(1)}, \dots, \mathbf{c}_y^{(t)} = \mathbf{v}^{(t)} \mid b = 0, \mathbf{b}_0^*, \mathbf{b}_1^*] \\ &= \Pr[\mathbf{b}_0^* \oplus \mathcal{E}_{\text{Enroll}} \oplus \mathbf{r} = \mathbf{v}, \mathbf{b}_0^* \oplus \mathcal{E}_{\text{Probe}} \oplus \mathbf{r} = \mathbf{v}^{(1)}, \dots, \mathbf{b}_0^* \oplus \mathcal{E}_{\text{Probe}} \oplus \mathbf{r} = \mathbf{v}^{(t)} \mid \mathbf{b}_0^*, \mathbf{b}_1^*] \\ &= \Pr[\mathbf{r} = \mathbf{v} \oplus \mathbf{b}_0^* \oplus \mathcal{E}_{\text{Enroll}} = \mathbf{v}^{(1)} \oplus \mathbf{b}_0^* \oplus \mathcal{E}_{\text{Probe}} = \dots = \mathbf{v}^{(t)} \oplus \mathbf{b}_0^* \oplus \mathcal{E}_{\text{Probe}} \mid \mathbf{b}_0^*, \mathbf{b}_1^*] \\ &= \Pr[\mathbf{r} = \mathbf{v} \oplus \mathbf{b}_1^* \oplus \mathcal{E}_{\text{Enroll}} = \mathbf{v}^{(1)} \oplus \mathbf{b}_1^* \oplus \mathcal{E}_{\text{Probe}} = \dots = \mathbf{v}^{(t)} \oplus \mathbf{b}_1^* \oplus \mathcal{E}_{\text{Probe}} \mid \mathbf{b}_0^*, \mathbf{b}_1^*] \\ &= \Pr[\mathbf{b}_1^* \oplus \mathcal{E}_{\text{Enroll}} \oplus \mathbf{r} = \mathbf{v}, \mathbf{b}_1^* \oplus \mathcal{E}_{\text{Probe}} \oplus \mathbf{r} = \mathbf{v}^{(1)}, \dots, \mathbf{b}_1^* \oplus \mathcal{E}_{\text{Probe}} \oplus \mathbf{r} = \mathbf{v}^{(t)} \mid \mathbf{b}_0^*, \mathbf{b}_1^*] \\ &= \Pr[\mathbf{c}_x = \mathbf{v}, \mathbf{c}_y^{(1)} = \mathbf{v}^{(1)}, \dots, \mathbf{c}_y^{(t)} = \mathbf{v}^{(t)} \mid b = 1, \mathbf{b}_0^*, \mathbf{b}_1^*] \end{aligned}$$

Hence, the adversary cannot distinguish between  $\mathbf{c}_x, \mathbf{c}_y^{(1)}, \dots, \mathbf{c}_y^{(t)}$  generated from  $\mathcal{B}^{(0)}$  and  $\mathcal{B}^{(1)}$ . □

## 4 Security Analysis: fh-IPFE-based Instantiation

**Definition 4** (Function-Hiding Inner Product Functional Encryption (adapted from [Kim+16])). A *function-hiding inner product functional encryption* (fh-IPFE) scheme FE for a field  $\mathbb{F}$  and input length  $k$  is composed of PPT algorithms FE.Setup, FE.KeyGen, FE.Enc, and FE.Dec:

- FE.Setup( $1^\lambda$ )  $\rightarrow$  msk, pp: It outputs the public parameter pp and the master secret key msk.
- FE.KeyGen(msk, pp,  $\mathbf{x}$ )  $\rightarrow$   $f_{\mathbf{x}}$ : It generates the functional decryption key  $f_{\mathbf{x}}$  for an input vector  $\mathbf{x} \in \mathbb{F}^k$ .
- FE.Enc(msk, pp,  $\mathbf{y}$ )  $\rightarrow$   $\mathbf{c}_{\mathbf{y}}$ : It encrypts the input vector  $\mathbf{y} \in \mathbb{F}^k$  to the ciphertext  $\mathbf{c}_{\mathbf{y}}$ .
- FE.Dec(pp,  $f_{\mathbf{x}}$ ,  $\mathbf{c}_{\mathbf{y}}$ )  $\rightarrow$   $z$ : It outputs a value  $z \in \mathbb{F}$  or an error symbol  $\perp$ .

**Correctness** An fh-IPFE scheme FE is *correct* if  $\forall(\text{msk}, \text{pp}) \leftarrow \text{FE.Setup}(1^\lambda)$  and  $\mathbf{x}, \mathbf{y} \in \mathbb{F}^k$ , we have

$$\text{FE.Dec}(\text{pp}, \text{FE.KeyGen}(\text{msk}, \text{pp}, \mathbf{x}), \text{FE.Enc}(\text{msk}, \text{pp}, \mathbf{y})) = \mathbf{x}\mathbf{y}^T \in \mathbb{F}.$$

### 4.1 Instantiation with an fh-IPFE Scheme

Let  $\text{FE} = (\text{FE.Setup}, \text{FE.KeyGen}, \text{FE.Enc}, \text{FE.Dec})$  be an fh-IPFE scheme. Following [EM23], we can instantiate a biometric authentication scheme using FE with the distance metric the Euclidean distance. Let  $\text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$  and  $\text{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$  both output vectors in  $\{0, 1, \dots, m\}^k$  for all biometric distributions  $\mathcal{B} \in \mathbb{B}$ . For a pre-defined real number  $\tau \geq 0$ , define

$$\text{BioCompare}(\mathbf{b}, \mathbf{b}') \rightarrow \|\mathbf{b} - \mathbf{b}'\|^2 \quad \text{and} \quad \text{Verify}(s) \rightarrow \begin{cases} 1 & \text{if } \sqrt{s} \leq \tau \\ 0 & \text{if } \sqrt{s} > \tau \end{cases}.$$

Now, let the associated field of FE be  $\mathbb{Z}_q$ , where  $q$  is a prime number larger than the maximum possible Euclidean distance  $m^2 \cdot k$ . The scheme is instantiated as follows.

- Setup( $1^\lambda$ ): It calls FE.Setup( $1^\lambda$ )  $\rightarrow$  msk, pp and outputs  $\text{esk} \leftarrow (\text{msk}, \text{pp})$ ,  $\text{psk} \leftarrow (\text{msk}, \text{pp})$  and  $\text{csk} \leftarrow \text{pp}$ .
- Enroll(esk,  $\mathbf{b}$ ): On input a template vector  $\mathbf{b} = (b_1, b_2, \dots, b_k)$ , the algorithm first encodes it as  $\mathbf{x} = (x_1, x_2, \dots, x_{k+2}) = (b_1, b_2, \dots, b_k, 1, \|\mathbf{b}\|^2)$ . Next, it calls FE.KeyGen(msk, pp,  $\mathbf{x}$ )  $\rightarrow$   $f_{\mathbf{x}}$  and outputs  $\mathbf{c}_{\mathbf{x}} \leftarrow f_{\mathbf{x}}$ .
- Probe(psk,  $\mathbf{b}'$ ): On input a template vector  $\mathbf{b}' = (b'_1, b'_2, \dots, b'_k)$ , the algorithm first encodes it as  $\mathbf{y} = (y_1, y_2, \dots, y_{k+2}) = (-2b'_1, -2b'_2, \dots, -2b'_k, \|\mathbf{b}'\|^2, 1)$ . Next, it calls FE.Enc(msk, pp,  $\mathbf{y}$ )  $\rightarrow$   $\mathbf{c}_{\mathbf{y}}$  and outputs  $\mathbf{c}_{\mathbf{y}}$ .

- $\text{Compare}(\text{csk}, \mathbf{c}_x, \mathbf{c}_y)$ : It calls  $\text{FE.Dec}(\text{pp}, \mathbf{c}_x, \mathbf{c}_y) \rightarrow s$  and outputs the value  $s$ .

By the correctness of the functional encryption scheme FE, we have

$$s = \text{FE.Dec}(\text{pp}, \mathbf{c}_x, \mathbf{c}_y) = \mathbf{x}\mathbf{y}^T = \sum_{i=1}^k -2b_i b'_i + \|\mathbf{b}\|^2 + \|\mathbf{b}'\|^2 = \|\mathbf{b} - \mathbf{b}'\|^2.$$

which is equal to  $\text{BioCompare}(\mathbf{b}, \mathbf{b}')$ . Therefore, if two templates  $\mathbf{b}$  and  $\mathbf{b}'$  are close enough such that  $\|\mathbf{b} - \mathbf{b}'\| \leq \tau$ , the scheme results in  $r = 1$ , a successful authentication.

Instantiated with an fh-IPFE scheme in this way, the comparison secret key  $\text{csk}$  is public, and the enrollment secret key  $\text{esk}$  and probe secret key  $\text{psk}$  are the same. Anyone with access to the enrollment message  $\mathbf{c}_x$  and either  $\text{esk}$  or  $\text{psk}$  can probe any (invalidly encoded)  $\mathbf{y}' \in \mathbb{Z}_q^{k+2}$  and find  $\mathbf{x}\mathbf{y}'^T$  to get partial or full information about the biometric template  $\mathbf{b}$ . Even if the adversary has no  $\text{esk}$  or  $\text{psk}$ , if it can sample ciphertexts  $\mathbf{c}_y$  corresponding to some unknown random vectors  $\mathbf{y}$ , and if the field size  $q$  is not large enough, it can also find a forged  $\mathbf{c}_{y^*}$  such that  $\mathbf{x}\mathbf{y}^{*T} \leq \tau$  with a non-negligible probability to impersonate the user by sampling many times offline.

Now, let  $\Pi$  be an authentication scheme instantiated by an fh-IPFE scheme FE. We discuss the UF and IND security of  $\Pi$  in this section. For this, we first define two security notions of FE.

## 4.2 fh-IND Security of FE

Given an fh-IPFE scheme FE, we define the fh-IND game [Kim+16] in Algorithm 4.

---

### Algorithm 4 fh-IND<sub>FE</sub>( $\mathcal{A}$ )

---

- 1:  $b \leftarrow_{\$} \{0, 1\}$
  - 2:  $\text{msk}, \text{pp} \leftarrow \text{FE.Setup}(1^\lambda)$
  - 3:  $\tilde{b} \leftarrow \mathcal{A}^{\mathcal{O}_{\text{KeyGen}}, \mathcal{O}_{\text{Enc}}}(\text{pp})$
  - 4: **return**  $1_{\tilde{b}=b}$
- 

- $\mathcal{O}_{\text{KeyGen}}(\cdot, \cdot)$ : On input pair  $(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$ , it outputs  $\text{FE.KeyGen}(\text{msk}, \text{pp}, \mathbf{x}^{(b)})$ .
- $\mathcal{O}_{\text{Enc}}(\cdot, \cdot)$ : On input pair  $(\mathbf{y}^{(0)}, \mathbf{y}^{(1)})$ , it outputs  $\text{FE.Enc}(\text{msk}, \text{pp}, \mathbf{y}^{(b)})$ .

To avoid trivial attacks, we consider *admissible adversaries*.

**Definition 5** (Admissible Adversary). Let  $\mathcal{A}$  be an adversary in an fh-IND game, and let  $(\mathbf{x}_1^{(0)}, \mathbf{x}_1^{(1)}), \dots, (\mathbf{x}_{Q_K}^{(0)}, \mathbf{x}_{Q_K}^{(1)})$  be its queries to  $\mathcal{O}_{\text{KeyGen}}$  and  $(\mathbf{y}_1^{(0)}, \mathbf{y}_1^{(1)}), \dots, (\mathbf{y}_{Q_E}^{(0)}, \mathbf{y}_{Q_E}^{(1)})$  be its queries to  $\mathcal{O}_{\text{Enc}}$ . We say  $\mathcal{A}$  is *admissible* if  $\forall i \in [Q_K], \forall j \in [Q_E]$ ,

$$\mathbf{x}_i^{(0)} \mathbf{y}_j^{(0)T} = \mathbf{x}_i^{(1)} \mathbf{y}_j^{(1)T}$$

**Definition 6** (fh-IND Security). An fh-IPFE scheme FE is called fh-IND secure if for any admissible adversary  $\mathcal{A}$ , the advantage of  $\mathcal{A}$  in the fh-IND game in Algorithm 4 is

$$\text{Adv}_{\text{FE}, \mathcal{A}}^{\text{fh-IND}} := \left| \Pr[\text{fh-IND}_{\text{FE}}(\mathcal{A}) \rightarrow 1] - \frac{1}{2} \right| = \text{negl}.$$

We note that fh-IND security is a standard notion for an fh-IPFE, and constructions in [DDM15; TAO16; Kim+16] are proven fh-IND. However, fh-IND security may not be sufficient for the UF security of the instantiation in Section 4.1.

**Theorem 3.** *An instantiation  $\Pi$  using the construction in [Kim+16] is not option-UF for any option.*

We recall the construction in [Kim+16] in Appendix A.

*Proof.* Let  $\mathcal{A}$  be a UF game adversary that returns  $(K_1, K_2) = (1, (1, \dots, 1))$ . Then, in the decryption,

$$D_1 = e(g_1, g_2)^0 = 1 \quad \text{and} \quad D_2 = e(g_1, g_2)^0 = 1$$

As  $D_1^0 = D_2$ , the decryption returns 0 and let the adversary win the game with probability 1. □

### 4.3 RUF Security of FE

Let  $\mathbb{F} = \mathbb{Z}_q$ . We also define the  $\text{RUF}_{\text{FE}}^{\mathcal{O}}$  game in Algorithm 5.

---

#### Algorithm 5 $\text{RUF}_{\text{FE}}^{\mathcal{O}}(\mathcal{A})$

---

```

1:  $\mathbf{r} \leftarrow \$ \mathbb{F}^k$ 
2:  $\text{msk}, \text{pp} \leftarrow \text{FE.Setup}(1^\lambda)$ 
3:  $\mathbf{c} \leftarrow \text{FE.KeyGen}(\text{msk}, \text{pp}, \mathbf{r})$ 
4:  $\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}}(\text{pp}, \mathbf{c})$ 
5: if  $\tilde{\mathbf{z}}$  is equal to any output of  $\mathcal{O}'_{\text{Enc}}$  then
6:   return 0
7: end if
8:  $s \leftarrow \text{FE.Dec}(\text{pp}, \mathbf{c}, \tilde{\mathbf{z}})$ 
9: return  $1_{s \neq \perp}$ 

```

---

The oracle  $\mathcal{O}$  can be nothing or include the following options based on the threat model.

- $\mathcal{O}'_{\text{KeyGen}}(\cdot)$ : On input  $\mathbf{x}'$ , it outputs  $\text{FE.KeyGen}(\text{msk}, \text{pp}, \mathbf{x}')$ .
- $\mathcal{O}'_{\text{Enc}}(\cdot)$ : On input  $\mathbf{y}'$ , it outputs  $\text{FE.Enc}(\text{msk}, \text{pp}, \mathbf{y}')$ . The adversary is required to return  $\tilde{\mathbf{z}}$  that is not equal to any output of this oracle.

**Definition 7** (RUF Security). An fh-IPFE scheme  $\text{FE}$  is called  $\mathcal{O}$ -RUF secure if for any adversary  $\mathcal{A}$ , the advantage of  $\mathcal{A}$  in the  $\text{RUF}_{\text{FE}}^{\mathcal{O}}$  game in Algorithm 5 is

$$\text{Adv}_{\text{FE}, \mathcal{A}}^{\text{RUF}, \mathcal{O}} := \Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}}(\mathcal{A}) \rightarrow 1] = \text{negl}.$$

We say  $\text{FE}$  is RUF secure if it is  $\{\mathcal{O}'_{\text{KeyGen}}, \mathcal{O}'_{\text{Enc}}\}$ -RUF secure.

### 4.3.1 Achievability of RUF Security

We note that RUF security is a new security notion of fh-IPFE but can be achieved with a digital signature scheme. Let  $\text{Sig} = (\text{Sig.KeyGen}, \text{Sig.Sign}, \text{Sig.Verify})$  be an sEUF-CMA signature scheme. By adding  $\text{Sig}$ , an fh-IPFE scheme  $\text{FE}$  can be upgraded to an RUF scheme  $\text{FE}'$ .

- $\text{FE}'.\text{Setup}(1^\lambda)$ : Run  $\text{FE}.\text{Setup}(1^\lambda) \rightarrow (\text{msk}, \text{pp})$  and  $\text{Sig}.\text{KeyGen}(1^\lambda) \rightarrow (\text{sk}_{\text{Sig}}, \text{pk}_{\text{Sig}})$ . Output  $\text{msk}' = (\text{msk}, \text{sk}_{\text{Sig}})$  and  $\text{pp}' = (\text{pp}, \text{pk}_{\text{Sig}})$ .
- $\text{FE}'.\text{KeyGen}(\text{msk}', \mathbf{x})$ : Run  $\text{FE}.\text{KeyGen}(\text{msk}, \mathbf{x}) \rightarrow f_{\mathbf{x}}$  and output  $f_{\mathbf{x}}$ .
- $\text{FE}'.\text{Enc}(\text{msk}', \mathbf{y})$ : Run  $\text{FE}.\text{Enc}(\text{msk}, \mathbf{y}) \rightarrow \mathbf{c}_{\mathbf{y}}$  and sign  $\mathbf{c}_{\mathbf{y}}$  by  $\text{Sig}.\text{Sign}(\text{sk}_{\text{Sig}}, \mathbf{c}_{\mathbf{y}}) \rightarrow \sigma$ . Output  $\mathbf{c}_{\mathbf{y}}' = (\mathbf{c}_{\mathbf{y}}, \sigma)$ .
- $\text{FE}'.\text{Dec}(\text{pp}', f_{\mathbf{x}}, \mathbf{c}_{\mathbf{y}}')$ : Output the decryption  $\text{FE}.\text{Dec}(\text{pp}, f_{\mathbf{x}}, \mathbf{c}_{\mathbf{y}})$  if the verification  $\text{Sig}.\text{Verify}(\text{pk}_{\text{Sig}}, \mathbf{c}_{\mathbf{y}}, \sigma) = 1$ . Otherwise, output  $\perp$ .

**Theorem 4.** For any fh-IPFE  $\text{FE}$ ,  $\text{FE}'$  is an RUF fh-IPFE.

*Proof.* Given an adversary  $\mathcal{A}$  in the  $\text{RUF}_{\text{FE}'}^{\mathcal{O}'_{\text{KeyGen}}, \mathcal{O}'_{\text{Enc}}}$  game, consider the reduction adversary  $\mathcal{R}$  in Algorithm 6 which plays the sEUF-CMA game of  $\text{Sig}$ .  $\mathcal{R}$  is given a verification public key  $\text{pk}_{\text{Sig}}$  and a signing oracle  $\mathcal{O}_{\text{Sig}}$  and returns a forged message-signature pair that is not equal to any previous answer of  $\mathcal{O}_{\text{Sig}}$ . To run  $\mathcal{A}$ ,  $\mathcal{R}$  simulates each oracle in the following way.

- $\mathcal{O}'_{\text{KeyGen}}(\mathbf{x}')$ : Return  $\text{FE}.\text{KeyGen}(\text{msk}, \mathbf{x})$ .
- $\mathcal{O}'_{\text{Enc}}(\mathbf{y}')$ : Run  $\text{FE}.\text{Enc}(\text{msk}, \mathbf{y}) \rightarrow \mathbf{c}_{\mathbf{y}}$  and call the signing oracle  $\mathcal{O}_{\text{Sig}}(\mathbf{c}_{\mathbf{y}}) \rightarrow \sigma$ . Output  $\mathbf{c}_{\mathbf{y}}' = (\mathbf{c}_{\mathbf{y}}, \sigma)$ .

---

#### Algorithm 6 $\mathcal{R}^{\mathcal{O}_{\text{Sig}}}(\text{pk}_{\text{Sig}})$

---

```

1:  $\mathbf{r} \leftarrow \mathbb{F}^k$ 
2:  $\text{msk}, \text{pp} \leftarrow \text{FE}.\text{Setup}(1^\lambda)$ 
3:  $\mathbf{c} \leftarrow \text{FE}.\text{KeyGen}(\text{msk}, \text{pp}, \mathbf{r})$ 
4:  $\text{pp}' \leftarrow (\text{pp}, \text{pk}_{\text{Sig}})$ 
5:  $\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}'_{\text{KeyGen}}, \mathcal{O}'_{\text{Enc}}}(\text{pp}', \mathbf{c})$ 
6: Parse  $(\mathbf{c}_{\mathbf{z}}, \sigma') \leftarrow \tilde{\mathbf{z}}$ 
7: return  $(\mathbf{c}_{\mathbf{z}}, \sigma')$ 

```

---

$\mathcal{R}$  perfectly simulates a RUF game for  $\mathcal{A}$ , and if  $\mathcal{A}$  wins the RUF game,  $(\mathbf{c}_z, \sigma')$  is not equal to any previous answer of  $\mathcal{O}'_{\text{Enc}}$ , and therefore not equal to any previous message-signature pair  $(\mathbf{c}_y, \sigma)$  given from the signing oracle  $\mathcal{O}_{\text{Sign}}$ . Now, since **Sig** is sEUF-CMA,

$$\Pr[\text{RUF}^{\mathcal{O}'_{\text{KeyGen}}}(\mathcal{A}) \rightarrow 1] \leq \Pr[\text{Sig.Verify}(\text{pk}_{\text{Sig}}, \mathbf{c}_z, \sigma') = 1] = \text{negl}.$$

□

#### 4.4 UF Security of $\Pi$

We first consider option-UF security when option includes  $\mathcal{O}_{\text{Enroll}}$ . Note that in this instantiation, **csk** is the public parameter **pp** of FE and assumed to be given to all adversaries.

**Theorem 5.** *Let  $\text{option} = \{\mathbf{c}_x, \text{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\text{Enroll}}\}$ . For any distribution family  $\mathbb{B}$ , if FE is fh-IND and  $\mathcal{O}'_{\text{KeyGen}}$ -RUF, then  $\Pi$  is option-UF.*

*Proof.* Given an adversary  $\mathcal{A}$  in the  $\text{UF}_{\text{option}}$  game, consider the reduction adversary  $\mathcal{R}$  in Algorithm 7 which plays the fh-IND game.  $\mathcal{R}$  runs  $\mathcal{A}$  and simulates  $\mathcal{O}_{\text{Enroll}}(\text{esk}, \mathbf{b}')$  by first encoding  $\mathbf{b}' = (b'_1, \dots, b'_k)$  into  $\mathbf{x}' = (b'_1, \dots, b'_k, 1, \|\mathbf{b}'\|^2)$  and calling  $\mathcal{O}_{\text{KeyGen}}(\mathbf{x}', \mathbf{r})$  given in the fh-IND game. Note that since  $\mathcal{R}$  never calls  $\mathcal{O}_{\text{Enc}}$ , it is an admissible adversary.

---

##### Algorithm 7 $\mathcal{R}^{\mathcal{O}_{\text{KeyGen}}, \mathcal{O}_{\text{Enc}}}(\text{pp})$

---

```

1:  $\mathcal{B} \leftarrow \mathbb{B}$ ,  $\mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$ 
2:  $\mathbf{b} = (b_1, \dots, b_k) \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ 
3:  $\mathbf{x} \leftarrow (b_1, \dots, b_k, 1, \|\mathbf{b}\|^2)$ 
4:  $\mathbf{r} \leftarrow \mathbb{F}^{k+2}$ 
5:  $\mathbf{c} \leftarrow \mathcal{O}_{\text{KeyGen}}(\mathbf{x}, \mathbf{r})$ 
6:  $\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\text{Enroll}}}(\mathbf{c}, \text{pp})$ 
7:  $s \leftarrow \text{FE.Dec}(\text{pp}, \mathbf{c}, \tilde{\mathbf{z}})$ 
8: if  $\text{Verify}(s) = 1$  then
9:   return  $\tilde{b} = 0$ 
10: else
11:   return  $\tilde{b} \leftarrow \{0, 1\}$ 
12: end if
```

---

If the challenge bit  $b = 0$ , then  $\mathcal{R}$  perfectly simulates a  $\text{UF}_{\text{option}}$  game for  $\mathcal{A}$ . Therefore, the probability that  $\text{Verify}(s) = 1$  in Line 8 is  $\Pr[\text{UF}_{\text{option}}(\mathcal{A}) \rightarrow 1]$ .

For the case when the challenge bit  $b = 1$ , consider an adversary  $\mathcal{A}'$  in Algorithm 8 in the  $\text{RUF}^{\mathcal{O}'_{\text{KeyGen}}}$  game.  $\mathcal{A}'$  runs Line 1 and 6 of  $\mathcal{R}$  and simulates  $\mathcal{O}_{\text{Enroll}}(\text{esk}, \mathbf{b}')$  by first encoding  $\mathbf{b}'$  into  $\mathbf{x}'$  as before and calling  $\mathcal{O}'_{\text{KeyGen}}(\mathbf{x}')$  given in the  $\text{RUF}^{\mathcal{O}'_{\text{KeyGen}}}$  game.

Now, if the challenge bit  $b = 1$ , then  $\mathcal{R}$  perfectly simulates  $\mathcal{A}'$  in the  $\text{RUF}^{\mathcal{O}'_{\text{KeyGen}}}$  game. The probability that  $\text{Verify}(s) = 1$  in Line 8 is smaller than  $\Pr[s \neq \perp] = \Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{KeyGen}}}(\mathcal{A}') \rightarrow 1]$

---

**Algorithm 8**  $\mathcal{A}'^{\mathcal{O}'_{\text{KeyGen}}}(\text{pp}, \mathbf{c})$ 


---

- 1:  $\mathcal{B} \leftarrow \mathbb{B}, \quad \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$
  - 2:  $\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\text{Enroll}}}(\mathbf{c}, \text{pp})$
  - 3: **return**  $\tilde{\mathbf{z}}$
- 

In conclusion,

$$\begin{aligned}
\Pr[\text{fh-IND}(\mathcal{R}) \rightarrow 1] &= \Pr[b = 0] \cdot \left( \Pr[\text{Verify}(s) = 1 \mid b = 0] + \frac{1}{2} \cdot \Pr[\text{Verify}(s) = 0 \mid b = 0] \right) \\
&\quad + \Pr[b = 1] \cdot \frac{1}{2} \cdot \Pr[\text{Verify}(s) = 0 \mid b = 1] \\
&= \frac{1}{2} + \frac{1}{4} (\Pr[\text{Verify}(s) = 1 \mid b = 0] - \Pr[\text{Verify}(s) = 1 \mid b = 1]) \\
&\geq \frac{1}{2} + \frac{1}{4} (\Pr[\text{UF}_{\text{option}}(\mathcal{A}) \rightarrow 1] - \Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{KeyGen}}}(\mathcal{A}') \rightarrow 1])
\end{aligned}$$

Since both  $\mathbf{Adv}_{\text{FE}, \mathcal{R}}^{\text{fh-IND}} = |\Pr[\text{fh-IND}(\mathcal{R}) \rightarrow 1] - \frac{1}{2}|$  and  $\mathbf{Adv}_{\text{FE}, \mathcal{A}'}^{\text{RUF}, \mathcal{O}'_{\text{KeyGen}}} = \Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{KeyGen}}}(\mathcal{A}') \rightarrow 1]$  are negligible,

$$\Pr[\text{UF}_{\text{option}}(\mathcal{A}) \rightarrow 1] \leq 4 \cdot \mathbf{Adv}_{\text{FE}, \mathcal{R}}^{\text{fh-IND}} + \mathbf{Adv}_{\text{FE}, \mathcal{A}'}^{\text{RUF}, \mathcal{O}'_{\text{KeyGen}}} = \text{negl}.$$

□

For **option** that includes  $\mathcal{O}_{\text{Probe}}$ , we first note that for any  $d \in \mathbb{Z}_q$  and any nonzero vector  $\mathbf{r} \in \mathbb{Z}_q^{k+2}$ , there exists a vector  $\mathbf{y} \in \mathbb{Z}_q^{k+2}$  such that  $\mathbf{r}\mathbf{y}^T = d$ .

**Theorem 6.** *Let  $\text{option} = \{\mathbf{c}_x, \text{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\text{Probe}}\}$ . For any distribution family  $\mathbb{B}$ , if FE is fh-IND and  $\mathcal{O}'_{\text{Enc}}$ -RUF, then  $\Pi$  is option-UF.*

*Proof.* Given an adversary  $\mathcal{A}$  in the  $\text{UF}_{\text{option}}$  game, consider the reduction adversary  $\mathcal{R}$  in Algorithm 9 which plays the fh-IND game.  $\mathcal{R}$  runs  $\mathcal{A}$  and simulates  $\mathcal{O}_{\text{Probe}}$  in the following way.

- $\mathcal{O}_{\text{Probe}}(\text{psk}, \mathbf{b}')$ : On input  $\mathbf{b}' = (b'_1, \dots, b'_k)$ , it first encodes it as  $\mathbf{y}' = (-2b'_1, \dots, -2b'_k, \|\mathbf{b}'\|^2, 1)$ . Next, it computes  $d \leftarrow \mathbf{x}\mathbf{y}'^T$  and finds a vector  $\mathbf{y}''$  such that  $\mathbf{r}\mathbf{y}''^T = d$ . Finally, it calls  $\mathcal{O}_{\text{Enc}}(\mathbf{y}', \mathbf{y}'')$ , which is given by the fh-IND game, and returns the result.

Note that  $(\mathbf{x}, \mathbf{r})$  is the only query of  $\mathcal{R}$  to  $\mathcal{O}_{\text{KeyGen}}$ , and for any query  $(\mathbf{y}', \mathbf{y}'')$  to  $\mathcal{O}_{\text{Enc}}$ , it satisfies  $\mathbf{x}\mathbf{y}'^T = \mathbf{r}\mathbf{y}''^T$ . Hence,  $\mathcal{R}$  is an admissible adversary.

If the challenge bit  $b = 0$ , then  $\mathcal{R}$  perfectly simulates a  $\text{UF}_{\text{option}}$  game for  $\mathcal{A}$ . Therefore, the probability that  $\text{Verify}(s) = 1$  in Line 11 is  $\Pr[\text{UF}_{\text{option}}(\mathcal{A}) \rightarrow 1]$ .

For the case when the challenge bit  $b = 1$ , consider an adversary  $\mathcal{A}'$  in Algorithm 10 in the  $\text{RUF}_{\text{Enc}}^{\mathcal{O}'_{\text{Enc}}}$  game.  $\mathcal{A}'$  runs  $\mathcal{A}$  and simulates  $\mathcal{O}_{\text{Probe}}$  in the following way.

- $\mathcal{O}_{\text{Probe}}(\text{psk}, \mathbf{b}')$ : It first encodes  $\mathbf{b}'$  into  $\mathbf{y}'$  as before. Next, it computes  $d \leftarrow \mathbf{x}^{(*)}\mathbf{y}'^T$  and finds a vector  $\mathbf{y}''$  such that  $\mathbf{r}\mathbf{y}''^T = d$ . Finally, it calls  $\mathcal{O}'_{\text{Enc}}(\mathbf{y}'')$ , which is given by the  $\text{RUF}_{\text{Enc}}^{\mathcal{O}'_{\text{Enc}}}$  game, and returns the result.

---

**Algorithm 9**  $\mathcal{R}^{\mathcal{O}_{\text{KeyGen}}, \mathcal{O}_{\text{Enc}}}(\text{pp})$ 


---

```

1:  $\mathcal{B} \leftarrow_{\$} \mathbb{B}, \quad \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$ 
2:  $\mathbf{b} = (b_1, \dots, b_k) \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ 
3:  $\mathbf{x} \leftarrow (b_1, \dots, b_k, 1, \|\mathbf{b}\|^2)$ 
4:  $\mathbf{r} \leftarrow_{\$} \mathbb{F}^{k+2}$ 
5:  $\mathbf{c} \leftarrow \mathcal{O}_{\text{KeyGen}}(\mathbf{x}, \mathbf{r})$ 
6:  $\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\text{Probe}}}(\mathbf{c}, \text{pp})$ 
7: if  $\tilde{\mathbf{z}}$  is equal to any output of  $\mathcal{O}_{\text{Probe}}$  then
8:   return  $\perp$ 
9: end if
10:  $s \leftarrow \text{FE.Dec}(\text{pp}, \mathbf{c}, \tilde{\mathbf{z}})$ 
11: if  $\text{Verify}(s) = 1$  then
12:   return  $\tilde{b} = 0$ 
13: else
14:   return  $\tilde{b} \leftarrow_{\$} \{0, 1\}$ 
15: end if

```

---

To make  $\mathcal{R}$  simulate  $\mathcal{A}'$  in the  $\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}$  game, we still need to ensure two conditions.

- $\mathbf{r} \neq \mathbf{0}$ . Otherwise,  $\mathcal{A}'$  cannot simulate  $\mathcal{O}_{\text{Probe}}$ .
- $\tilde{\mathbf{z}} \neq \mathbf{c}^{(i)}$  for all  $i$ . The answers of  $\mathcal{O}_{\text{Probe}}$  have already been checked in  $\mathcal{R}$ .

Let  $\mathcal{A}'$  play a tweaked  $\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}$  game which does not check that  $\tilde{\mathbf{z}}$  is not equal to  $\mathbf{c}^{(i)}$  for all  $i$ . That is, the game only checks whether  $\tilde{\mathbf{z}}$  is not equal to any output of  $\mathcal{O}'_{\text{Enc}}$  called by  $\mathcal{O}_{\text{Probe}}$  of  $\mathcal{A}$ . Let the returned value of this game be  $V$ . We have Equation 1 and 2. The former one is a relation between  $\mathcal{R}$  playing fh-IND game when the challenge bit  $b = 1$  and  $V$ , and the latter is a relation between  $\mathcal{A}'$  playing a regular  $\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}$  game and the tweaked one.

$$\Pr[\text{Verify}(s) = 1 \mid b = 1 \wedge \mathbf{r} \neq \mathbf{0}] = \Pr[V = 1] \quad (1)$$

$$\Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}(\mathcal{A}') \rightarrow 1] = \Pr \left[ V = 1 \mid \bigwedge_{i=1}^{k+2} \tilde{\mathbf{z}} \neq \mathbf{c}^{(i)} \right] \quad (2)$$

For Equation 1, consider that

$$\begin{aligned}
\Pr[\text{Verify}(s) = 1 \mid b = 1] &= \Pr[\text{Verify}(s) = 1 \mid b = 1 \wedge \mathbf{r} \neq \mathbf{0}] \cdot \Pr[\mathbf{r} \neq \mathbf{0}] \\
&\quad + \Pr[\text{Verify}(s) = 1 \mid b = 1 \wedge \mathbf{r} = \mathbf{0}] \cdot \Pr[\mathbf{r} = \mathbf{0}] \\
&\leq \Pr[V = 1] + \Pr[\mathbf{r} = \mathbf{0}] \\
&= \Pr[V = 1] + \frac{1}{q^{k+2}}
\end{aligned}$$



**Algorithm 10**  $\mathcal{A}'^{\mathcal{O}'_{\text{Enc}}}(\text{pp}, \mathbf{c})$ 


---

```

1:  $\mathcal{B} \leftarrow \mathbb{B}$ ,  $\mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$ 
2:  $\mathbf{b}^{(*)} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ 
3:  $\mathbf{x}^{(*)} \leftarrow (b_1^{(*)}, \dots, b_k^{(*)}, 1, \|\mathbf{b}^{(*)}\|^2)$ 
4: Sample  $k+2$  linearly independent vectors  $\{\mathbf{e}^{(i)}\}_{i=1}^{k+2}$ .
5: for  $i = 1$  to  $k+2$  do
6:    $\mathbf{c}^{(i)} \leftarrow \mathcal{O}'_{\text{Enc}}(\mathbf{e}^{(i)})$ .
7:    $d_i \leftarrow \text{FE.Dec}(\text{pp}, \mathbf{c}, \mathbf{c}^{(i)})$ .
8: end for
9: Find the vector  $\mathbf{r}$  by solving the linear system  $\{\mathbf{r}\mathbf{e}^{(i)T} = d_i\}_{i=1}^{k+2}$ .
10: if  $\mathbf{r} = \mathbf{0}$  then
11:   return  $\perp$ 
12: end if
13:  $\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\text{Probe}}}(\mathbf{c}, \text{pp})$ 
14: return  $\tilde{\mathbf{z}}$ 

```

---

For Equation 2, consider that

$$\begin{aligned}
\Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}(\mathcal{A}') \rightarrow 1] &= \Pr\left[V = 1 \mid \bigwedge_{i=1}^{k+2} \tilde{\mathbf{z}} \neq \mathbf{c}^{(i)}\right] \\
&\geq \Pr[V = 1] - \Pr\left[\neg\left(\bigwedge_{i=1}^{k+2} \tilde{\mathbf{z}} \neq \mathbf{c}^{(i)}\right)\right] \\
&= \Pr[V = 1] - \Pr\left[\bigvee_{i=1}^{k+2} \tilde{\mathbf{z}} = \mathbf{c}^{(i)}\right] \\
&\geq \Pr[V = 1] - \sum_{i=1}^{k+2} \Pr[\tilde{\mathbf{z}} = \mathbf{c}^{(i)}].
\end{aligned}$$

Note that each  $\mathbf{c}^{(i)} = \text{FE.Enc}(\text{msk}, \text{pp}, \mathbf{e}^{(i)})$  for some uniform nonzero vector  $\mathbf{e}^{(i)}$ . Also note that distinct vectors in  $\mathbb{Z}_q^{k+2}$  will have different encryptions due to the correctness of FE. Therefore,  $\Pr[\tilde{\mathbf{z}} = \mathbf{c}^{(i)}] \leq \frac{1}{q^{k+2}-1}$  and

$$\Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}(\mathcal{A}') \rightarrow 1] \geq \Pr[V = 1] - \frac{k+2}{q^{k+2}-1}.$$

Combining both results from Equation 1 and 2, we derive

$$\Pr[\text{Verify}(s) = 1 \mid b = 1] \leq \Pr[V = 1] + \frac{1}{q^{k+2}} \leq \Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}(\mathcal{A}') \rightarrow 1] + \frac{k+2}{q^{k+2}-1} + \frac{1}{q^{k+2}}.$$

Finally, similar to the proof of Theorem 5, we derive

$$\begin{aligned}
\Pr[\text{fh-IND}(\mathcal{R}) \rightarrow 1] &= \frac{1}{2} + \frac{1}{4} (\Pr[\text{Verify}(s) = 1 \mid b = 0] - \Pr[\text{Verify}(s) = 1 \mid b = 1]) \\
&\geq \frac{1}{2} + \frac{1}{4} \left( \Pr[\text{UF}_{\text{option}}(\mathcal{A}) \rightarrow 1] - \Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}(\mathcal{A}') \rightarrow 1] - \frac{k+2}{q^{k+2}-1} - \frac{1}{q^{k+2}} \right).
\end{aligned}$$

Since both  $\mathbf{Adv}_{\text{FE}, \mathcal{R}}^{\text{fh-IND}} = |\Pr[\text{fh-IND}(\mathcal{R}) \rightarrow 1] - \frac{1}{2}|$  and  $\mathbf{Adv}_{\text{FE}, \mathcal{A}'}^{\text{RUF}, \mathcal{O}'_{\text{Enc}}} = \Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}(\mathcal{A}') \rightarrow 1]$  are negligible,

$$\Pr[\text{UF}_{\text{option}}(\mathcal{A}) \rightarrow 1] \leq 4 \cdot \mathbf{Adv}_{\text{FE}, \mathcal{R}}^{\text{fh-IND}} + \mathbf{Adv}_{\text{FE}, \mathcal{A}'}^{\text{RUF}, \mathcal{O}'_{\text{Enc}}} + \frac{k+2}{q^{k+2}-1} + \frac{1}{q^{k+2}} = \text{negl}.$$

□

Unfortunately, for the instantiation in Section 4.1, we cannot achieve UF security when the adversary has **psk**, even if the false positive rate is negligible. The adversary can simply compute  $\mathbf{c} \leftarrow \text{Probe}(\mathbf{psk}, \mathbf{0})$  and return  $\mathbf{c}$ . The same results also hold for **option** that includes **esk** since both **psk** and **esk** are equal to **msk** and allow the adversary to run  $\text{FE.Enc}(\mathbf{msk}, \mathbf{pp}, \mathbf{v})$  for any vector  $\mathbf{v}$ . We state this result formally in the following theorem.

**Theorem 7.** *Let option include esk or psk. For any distribution family  $\mathbb{B}$  and functional encryption FE,  $\Pi$  is not option-UF.*

## 4.5 IND Security of $\Pi$

For the IND security, we first consider the following definition and assumption on the biometric distribution family  $\mathbb{B}$ .

**Definition 8.** For an authentication scheme  $\Pi$ , a distribution  $\mathcal{B} \in \mathbb{B}$ , and an integer  $t$ , define the distribution  $\mathcal{D}_{\mathcal{B}}(t)$  as

$$\mathcal{D}_{\mathcal{B}}(t) = (\text{BioCompare}(\mathbf{b}, \mathbf{b}^{(1)}), \text{BioCompare}(\mathbf{b}, \mathbf{b}^{(2)}), \dots, \text{BioCompare}(\mathbf{b}, \mathbf{b}^{(t)}))$$

where  $\mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$  and  $\mathbf{b}^{(i)} \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$  for all  $i \in [t]$ .

**Assumption 1.** Let  $t$  be an integer. Assume that for any two distributions  $\mathcal{B}^{(0)}$  and  $\mathcal{B}^{(1)}$  in the biometric distribution family  $\mathbb{B}$ ,  $\mathcal{D}_{\mathcal{B}^{(0)}}(t)$  and  $\mathcal{D}_{\mathcal{B}^{(1)}}(t)$  are the same.

Note that indistinguishability between  $\mathcal{D}_{\mathcal{B}^{(0)}}(t)$  and  $\mathcal{D}_{\mathcal{B}^{(1)}}(t)$  is a necessary condition to achieve IND security because

$$(\text{Compare}(\text{csk}, \mathbf{c}_{\mathbf{x}}, \mathbf{c}_{\mathbf{y}}^{(1)}), \dots, \text{Compare}(\text{csk}, \mathbf{c}_{\mathbf{x}}, \mathbf{c}_{\mathbf{y}}^{(t)})) = \mathcal{D}_{\mathcal{B}^{(b)}}(t)$$

where  $b$  is the challenge bit.

**Theorem 8.** *For any distribution family  $\mathbb{B}$  satisfying Assumption 1 and having a true positive rate  $TP > \frac{1}{\text{poly}}$ , if FE is fh-IND, then  $\Pi$  is IND.*

*Proof.* Given an adversary  $\mathcal{A}$  in the IND game, consider the reduction adversary  $\mathcal{R}$  in Algorithm 11 which plays the fh-IND game by running  $\mathcal{A}$ .

Note that  $(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$  is the only query of  $\mathcal{R}$  to  $\mathcal{O}_{\text{KeyGen}}$ , and for any query  $(\mathbf{y}^{(0)}, \mathbf{y}^{(1)})$  to  $\mathcal{O}_{\text{Enc}}$ , it satisfies  $\mathbf{x}^{(0)} \mathbf{y}^{(0)T} = \mathbf{x}^{(1)} \mathbf{y}^{(1)T}$ . Hence,  $\mathcal{R}$  is an admissible adversary.

**Algorithm 11**  $\mathcal{R}^{\mathcal{O}_{\text{KeyGen}}, \mathcal{O}_{\text{Enc}}}(\text{pp})$ 


---

```

1:  $\mathcal{B}^{(0)} \leftarrow_{\$} \mathbb{B}, \quad \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}^{(0)}$ 
2:  $\mathcal{B}^{(1)} \leftarrow_{\$} \mathbb{B}, \quad \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}^{(1)}$ 
3:  $\mathbf{b}^{(0)} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}^{(0)}}}(), \mathbf{x}^{(0)} \leftarrow (b_1^{(0)}, \dots, b_k^{(0)}, 1, \|\mathbf{b}^{(0)}\|^2)$ 
4:  $\mathbf{b}^{(1)} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}^{(1)}}}(), \mathbf{x}^{(1)} \leftarrow (b_1^{(1)}, \dots, b_k^{(1)}, 1, \|\mathbf{b}^{(1)}\|^2)$ 
5:  $\mathbf{c}_x \leftarrow \mathcal{O}_{\text{KeyGen}}(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$ 
6: for  $i = 1$  to  $t$  do
7:    $\mathbf{b}'^{(0)} \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}^{(0)}}}()$ 
8:    $\mathbf{y}^{(0)} \leftarrow (-2b_1'^{(0)}, \dots, -2b_k'^{(0)}, \|\mathbf{b}'^{(0)}\|^2, 1)$ 
9:   repeat
10:     $\mathbf{b}'^{(1)} \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}^{(1)}}}()$ 
11:     $\mathbf{y}^{(1)} \leftarrow (-2b_1'^{(1)}, \dots, -2b_k'^{(1)}, \|\mathbf{b}'^{(1)}\|^2, 1)$ 
12:  until  $\mathbf{x}^{(0)}\mathbf{y}^{(0)T} = \mathbf{x}^{(1)}\mathbf{y}^{(1)T}$ 
13:   $\mathbf{c}_y^{(i)} \leftarrow \mathcal{O}_{\text{Enc}}(\mathbf{y}^{(0)}, \mathbf{y}^{(1)})$ 
14: end for
15:  $\tilde{b} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{B}^{(0)}}, \mathcal{O}_{\mathcal{B}^{(1)}}}(\text{pp}, \mathbf{c}_x, \{\mathbf{c}_y^{(i)}\}_{i=1}^t)$ 
16: return  $\tilde{b}$ 

```

---

The probability that Line 12 is satisfied is

$$\begin{aligned}
\Pr[\mathcal{D}_{\mathcal{B}^{(0)}}(1) = \mathcal{D}_{\mathcal{B}^{(1)}}(1)] &\geq \sum_{i=0}^{\tau} \Pr[\mathcal{D}_{\mathcal{B}^{(0)}}(1) = i]^2 \quad (\text{Assumption 1}) \\
&\geq \frac{1}{\tau+1} \cdot \left( \sum_{i=0}^{\tau} \Pr[\mathcal{D}_{\mathcal{B}^{(0)}}(1) = i] \right)^2 \\
&= \frac{1}{\tau+1} \cdot \left( \Pr \left[ \begin{array}{l} \mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}^{(0)}}}() \\ \mathbf{b}' \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}^{(0)}}}() : \|\mathbf{b} - \mathbf{b}'\| \leq \tau \end{array} \right] \right)^2 \\
&= \frac{\text{TP}(\mathcal{B}^{(0)})^2}{\tau+1} = \frac{\text{TP}^2}{\tau+1} \quad (\text{Assumption 1})
\end{aligned}$$

The expected number of repetitions is bounded above by  $\frac{\tau+1}{\text{TP}^2}$ . Moreover, the probability that it is satisfied within  $T$  repetitions is at least

$$1 - \left(1 - \frac{\text{TP}^2}{\tau+1}\right)^T \geq 1 - e^{-T \cdot \frac{\text{TP}^2}{\tau+1}}$$

We can reach a  $1 - \text{negl.}$  probability that the loop will end within  $T$  times by setting a polynomial-size  $T$ .

Now, we show that  $\mathcal{R}$  perfectly simulate an IND game for  $\mathcal{A}$ . If the challenge bit  $b$  of the fh-IND game is 0,  $\mathbf{c}_x$  and  $\mathbf{c}_y^{(i)}$  for all  $i \in [t]$  are generated from  $\mathcal{B}^{(0)}$  and have the same distributions as the inputs for an adversary in IND game. If the challenge bit  $b$  is 1, we show that distributions of  $\mathbf{c}_x, \{\mathbf{c}_y^{(i)}\}_{i=1}^t$  also follow the same distribution given Assumption 1.

Let  $b' \in \{0, 1\}$ , define distributions

$$\begin{aligned} \mathbf{X}^{(b')} &= \{\mathbf{b}^{(b')} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}^{(b')}}}() : \mathbf{x}^{(b')} \leftarrow (b_1^{(b')}, \dots, b_k^{(b')}, 1, \|\mathbf{b}^{(b')}\|^2)\} \\ \mathbf{Y}_i^{(b')} &= \{\mathbf{b}^{(b')} \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}^{(b')}}}() : \mathbf{y}^{(b')} \leftarrow (-2b_1^{(b')}, \dots, -2b_k^{(b')}, \|\mathbf{b}^{(b')}\|^2, 1)\} \\ \{\mathbf{Y}_i^{(b')}\}_{i \in [t]} &= (\mathbf{Y}_1^{(b')}, \dots, \mathbf{Y}_t^{(b')}) \quad (t \text{ identical and independent distributions}) \end{aligned}$$

Let  $\mathbf{Y}'_i$  be the distribution of  $\mathbf{y}^{(1)}$  derived after the loop in Line 12 in the  $i$ -th iteration. For any  $\{d_i\}_{i=1}^t, d_i > 0$ ,

$$\begin{aligned} \Pr \left[ \bigwedge_{i=1}^t \mathbf{X}^{(0)} \mathbf{Y}_i^{(0)T} = d_i^2 \right] &= \Pr [\mathcal{D}_{\mathcal{B}^{(0)}}(t) = (d_1, \dots, d_t)] \\ &= \Pr [\mathcal{D}_{\mathcal{B}^{(1)}}(t) = (d_1, \dots, d_t)] = \Pr \left[ \bigwedge_{i=1}^t \mathbf{X}^{(1)} \mathbf{Y}_i^{(1)T} = d_i^2 \right] \end{aligned}$$

Hence, for any  $\mathbf{x}$  and  $\{\mathbf{y}_i\}_{i=1}^t$ ,

$$\begin{aligned} &\Pr[\mathbf{X}^{(1)} = \mathbf{x}, \mathbf{Y}'_1 = \mathbf{y}_1, \dots, \mathbf{Y}'_t = \mathbf{y}_t] \\ &= \sum_{d_1, \dots, d_t} \left( \Pr \left[ \mathbf{X}^{(1)} = \mathbf{x}, \mathbf{Y}_1^{(1)} = \mathbf{y}_1, \dots, \mathbf{Y}_t^{(1)} = \mathbf{y}_t \mid \bigwedge_{i=1}^t \mathbf{X}^{(1)} \mathbf{Y}_i^{(1)T} = d_i^2 \right] \right. \\ &\quad \left. \times \Pr \left[ \bigwedge_{i=1}^t \mathbf{X}^{(0)} \mathbf{Y}_i^{(0)T} = d_i^2 \right] \right) \\ &= \sum_{d_1, \dots, d_t} \left( \Pr \left[ \mathbf{X}^{(1)} = \mathbf{x}, \mathbf{Y}_1^{(1)} = \mathbf{y}_1, \dots, \mathbf{Y}_t^{(1)} = \mathbf{y}_t \mid \bigwedge_{i=1}^t \mathbf{X}^{(1)} \mathbf{Y}_i^{(1)T} = d_i^2 \right] \right. \\ &\quad \left. \times \Pr \left[ \bigwedge_{i=1}^t \mathbf{X}^{(1)} \mathbf{Y}_i^{(1)T} = d_i^2 \right] \right) \\ &= \Pr[\mathbf{X}^{(1)} = \mathbf{x}, \mathbf{Y}_1^{(1)} = \mathbf{y}_1, \dots, \mathbf{Y}_t^{(1)} = \mathbf{y}_t] \end{aligned}$$

which implies  $\mathcal{R}$  also perfectly simulate an IND game for  $\mathcal{A}$  when the challenge bit  $b = 1$ .

In conclusion,

$$\mathbf{Adv}_{\text{FE}, \mathcal{R}}^{\text{fh-IND}} = \mathbf{Adv}_{\Pi, \mathbb{B}, \mathcal{A}}^{\text{IND}} = \text{negl.}$$

which holds for all adversaries  $\mathcal{A}$  in the IND game. This implies the IND security of  $\Pi$ . □

## 5 Security Analysis: Relational Hash-based Instantiation

**Definition 9** (Relational Hash (adapted from [MR14])). Let  $R$  be a relation over sets  $X, Y$ , and  $Z$ . A *relational hash* scheme  $\text{RH}$  for  $R$  consists of PPT algorithms  $\text{RH.KeyGen}$ ,  $\text{RH.HASH}_1$ ,  $\text{RH.HASH}_2$ , and  $\text{RH.Verify}$ :

- $\text{RH.KeyGen}(1^\lambda) \rightarrow \text{pk}$ : It outputs a public hash key  $\text{pk}$ .
- $\text{RH.Hash}_1(\text{pk}, \mathbf{x}) \rightarrow \mathbf{h}_\mathbf{x}$ : Given a hash key  $\text{pk}$  and  $\mathbf{x} \in X$ , it outputs a hash  $\mathbf{h}_\mathbf{x}$ .
- $\text{RH.Hash}_2(\text{pk}, \mathbf{y}) \rightarrow \mathbf{h}_\mathbf{y}$ : Given a hash key  $\text{pk}$  and  $\mathbf{y} \in Y$ , it outputs a hash  $\mathbf{h}_\mathbf{y}$ .
- $\text{RH.Verify}(\text{pk}, \mathbf{h}_\mathbf{x}, \mathbf{h}_\mathbf{y}, \mathbf{z}) \rightarrow r \in \{0, 1\}$ : Given a hash key  $\text{pk}$ , two hashes  $\mathbf{h}_\mathbf{x}$  and  $\mathbf{h}_\mathbf{y}$ , and  $\mathbf{z} \in Z$ , it verifies whether the relation among  $\mathbf{x}, \mathbf{y}$  and  $\mathbf{z}$  holds.

**Correctness** A relational hash scheme  $\text{RH}$  is *correct* if  $\forall \mathbf{x}, \mathbf{y}, \mathbf{z} \in X \times Y \times Z$ ,

$$\Pr \left[ \begin{array}{l} \text{pk} \leftarrow \text{RH.KeyGen}(1^\lambda) \\ \mathbf{h}_\mathbf{x} \leftarrow \text{RH.Hash}_1(\text{pk}, \mathbf{x}) : \text{RH.Verify}(\text{pk}, \mathbf{h}_\mathbf{x}, \mathbf{h}_\mathbf{y}, \mathbf{z}) = R(\mathbf{x}, \mathbf{y}, \mathbf{z}) \\ \mathbf{h}_\mathbf{y} \leftarrow \text{RH.Hash}_2(\text{pk}, \mathbf{y}) \end{array} \right] = 1 - \text{negl}.$$

Note that  $Z_\lambda$  is an auxiliary input. When the relation  $R$  is over two sets  $X \times Y$ , we ignore  $Z$  and write  $\text{RH.Verify}(\text{pk}, \mathbf{h}_\mathbf{x}, \mathbf{h}_\mathbf{y})$ .

## 5.1 Instantiation with a Relational Hash Scheme

Let  $\text{RH} = (\text{RH.KeyGen}, \text{RH.Hash}_1, \text{RH.Hash}_2, \text{RH.Verify})$  be a relational hash scheme for the relation  $R^\tau$  of Hamming distance proximity parametrized by a constant  $\tau$ .

$$R^\tau = \{(\mathbf{x}, \mathbf{y}) \mid \text{HD}(\mathbf{x}, \mathbf{y}) \leq \tau \wedge \mathbf{x}, \mathbf{y} \in \{0, 1\}^k\}$$

Note that here we ignore the third parameter  $Z$ . Let  $\text{getEnroll}^{\mathcal{O}_\mathcal{B}}()$  and  $\text{getProbe}^{\mathcal{O}_\mathcal{B}}()$  both output vectors in  $\{0, 1\}^k$  for all biometric distributions  $\mathcal{B} \in \mathbb{B}$ , and let

$$\text{BioCompare}(\mathbf{b}, \mathbf{b}') \rightarrow \begin{cases} 1 & \text{if } (\mathbf{b}, \mathbf{b}') \in R^\tau \\ 0 & \text{if } (\mathbf{b}, \mathbf{b}') \notin R^\tau \end{cases} \quad \text{and} \quad \text{Verify}(s) \rightarrow s.$$

Following [MR14], we can instantiate a biometric authentication scheme using  $\text{RH}$ . Let the biometric distribution  $\mathcal{B} \subseteq \{0, 1\}^k$ .

- $\text{Setup}(1^\lambda)$ : It calls  $\text{RH.KeyGen}(1^\lambda) \rightarrow \text{pk}$  and outputs  $\text{esk} \leftarrow \text{pk}$ ,  $\text{psk} \leftarrow \text{pk}$ , and  $\text{csk} \leftarrow \text{pk}$ .
- $\text{Enroll}(\text{esk}, \mathbf{b})$ : Let  $\mathbf{x} \leftarrow \mathbf{b}$ . It calls  $\text{RH.Hash}_1(\text{pk}, \mathbf{x}) \rightarrow \mathbf{h}_\mathbf{x}$  and outputs  $\mathbf{c}_\mathbf{x} \leftarrow \mathbf{h}_\mathbf{x}$ .
- $\text{Probe}(\text{psk}, \mathbf{b}')$ : Let  $\mathbf{y} \leftarrow \mathbf{b}'$ . It calls  $\text{RH.Hash}_2(\text{pk}, \mathbf{y}) \rightarrow \mathbf{h}_\mathbf{y}$  and outputs  $\mathbf{c}_\mathbf{y} \leftarrow \mathbf{h}_\mathbf{y}$ .
- $\text{Compare}(\text{csk}, \mathbf{c}_\mathbf{x}, \mathbf{c}_\mathbf{y})$ : It calls  $\text{RH.Verify}(\text{pk}, \mathbf{h}_\mathbf{x}, \mathbf{h}_\mathbf{y}) \rightarrow s$  and outputs the value  $s$ .

By the correctness of the relational hash scheme  $\text{RH}$ , we have (except for a negligible probability),

$$r = 1 \Leftrightarrow (\mathbf{x}, \mathbf{y}) = (\mathbf{b}, \mathbf{b}') \in R^\tau \Leftrightarrow \text{HD}(\mathbf{b}, \mathbf{b}') \leq \tau$$

The idea behind this construction is that users hold a personal device which runs **Setup**, **Enroll**, and **Probe** using templates coming from a biometric sensor. The message  $\mathbf{c}_x$  along with the comparison key  $\mathbf{csk} = \mathbf{pk}$ , which is assumed to be public, are sent to a server in order to enroll the user, and  $\mathbf{c}_y$  is sent on authentication.

Now, let  $\Pi$  be an authentication scheme instantiated by a relational hash scheme RH. For the UF and IND security of  $\Pi$ , we define the unforgeability [MR14] of RH.

**Definition 10** (Unforgeability). A relational hash scheme RH is called *unforgeable* for the distribution  $\mathcal{X}$  if for any adversary  $\mathcal{A}$ , the following probability is negligible.

$$\Pr \left[ \begin{array}{l} \mathbf{x} \leftarrow \mathcal{X} \\ \mathbf{pk} \leftarrow \text{RH.KeyGen}(1^\lambda) \\ \mathbf{h}_x \leftarrow \text{RH.Hash}_1(\mathbf{pk}, \mathbf{x}) \\ \tilde{\mathbf{z}} \leftarrow \mathcal{A}(\mathbf{pk}, \mathbf{h}_x) \end{array} : \text{RH.Verify}(\mathbf{pk}, \mathbf{h}_x, \tilde{\mathbf{z}}) = 1 \right] = \text{negl}.$$

## 5.2 UF Security of $\Pi$

We first consider option that includes  $\mathbf{c}_x$ .

**Theorem 9.** Let  $\text{option} = \{\mathbf{c}_x, \text{esk}, \text{psk}, \text{csk}\}$ . If RH is unforgeable for the distribution

$$\mathcal{X} = \{\mathcal{B} \leftarrow \mathbb{B} : \mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \mid \mathbb{B}\},$$

then  $\Pi$  is *option-UF*.

In [MR14], the authors construct an RH that is unforgeable for the uniform distribution over  $\{0, 1\}^k$ , under the hardness of some computational problems. Note that we need to provide knowledge of  $\mathbb{B}$  in the distribution  $\mathcal{X}$ .

*Proof.* Recall that the distribution of  $\mathbf{c}_x$  in the UF game of the instantiation of Section 5.1 is

$$\left\{ \begin{array}{l} \mathcal{B} \leftarrow \mathbb{B} \\ \mathbf{pk} \leftarrow \text{RH.KeyGen}(1^\lambda) : \mathbf{c}_x \leftarrow \text{RH.Hash}_1(\mathbf{pk}, \mathbf{x}) \\ \mathbf{x} = \mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \end{array} \right\}$$

Also recall that  $\text{Verify}(\text{Compare}(\mathbf{csk}, \mathbf{c}_x, \tilde{\mathbf{z}})) = \text{RH.Verify}(\mathbf{pk}, \mathbf{c}_x, \tilde{\mathbf{z}})$ . The option-UF security is thus guaranteed by the unforgeability of RH.  $\square$

**Remark** As we mentioned in Section 3.1, an adversary with  $\text{psk}$  can enjoy a winning rate of the false positive rate FP of  $\mathbb{B}$ . Theorem 9 thus implies that if FP is not negligible, there does not exist an RH that is unforgeable for the distribution  $\{\mathcal{B} \leftarrow \mathbb{B} : \mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \mid \mathbb{B}\}$ .

Note that since  $\text{esk}$ ,  $\text{psk}$ , and  $\text{csk}$  are all public in this instantiation, it is meaningless to discuss  $\mathcal{O}_{\text{Enroll}}$ ,  $\mathcal{O}_{\text{Probe}}$ , or  $\mathcal{O}_{\text{log}}$ . In addition, for *option* that includes  $\mathcal{O}_{\mathcal{B}}$  or  $\mathcal{O}'_{\text{Probe}}$ , as discussed in Section 3.1, we cannot achieve option-UF security since  $\text{psk}$  is public in this instantiation.

For *option* that includes  $\mathcal{O}'_{\text{Enroll}}$ , we notice that for the RH construction in [MR14], there exists an invalid  $\mathbf{pk}'$  such that  $\text{RH.Hash}_1(\mathbf{pk}', \mathbf{x})$  directly leaks  $\mathbf{x}$ . By returning  $\text{RH.Hash}_2(\mathbf{pk}, \mathbf{x})$ , one can break the  $\text{UF}_{\text{option}}$  game with probability 1.

### 5.3 IND Security of $\Pi$

For the IND security, we have a negative result for  $\Pi$ .

**Theorem 10.** *For any distribution family  $\mathbb{B}$  that  $TP - FP > \frac{1}{\text{poly}}$ , and for any relational hash scheme  $RH$ ,  $\Pi$  is not IND for any  $t \geq 0$ .*

*Proof.* Consider the adversary  $\mathcal{A}$  in Algorithm 12. When the challenge bit  $b = 0$ , the probability that  $\mathcal{A}$  wins is  $TP$ . When the challenge bit  $b = 1$ , the probability that  $\mathcal{A}$  wins is  $1 - FP$ . Now,

$$\text{Adv}_{\Pi, \mathbb{B}, \mathcal{A}}^{\text{IND}} = \left| \Pr[\text{IND}_{\Pi}(\mathcal{A}) \rightarrow 1] - \frac{1}{2} \right| = \left| \frac{1}{2}(TP + 1 - FP) - \frac{1}{2} \right| > \frac{1}{\text{poly}}.$$

---

**Algorithm 12**  $\mathcal{A}^{\mathcal{O}_{\mathcal{B}(0)}, \mathcal{O}_{\mathcal{B}(1)}}(\text{csk} = \text{pk}, \mathbf{c}_x, \{\mathbf{c}_y^{(i)}\}_{i=1}^t)$

---

```

1:  $\mathbf{y}^{(0)} = \mathbf{b}^{(0)} \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}(0)}}()$ 
2:  $\mathbf{h}_y^{(0)} \leftarrow RH.\text{Hash}_2(\text{pk}, \mathbf{y}^{(0)})$ 
3: if  $RH.\text{Verify}(\text{pk}, \mathbf{c}_x, \mathbf{h}_y^{(0)}) = 1$  then
4:   return 0
5: else
6:   return 1
7: end if

```

---

□

We note that this insecurity result holds whenever  $\text{psk}$  is public. When  $\text{esk}$  is public, one can also use  $\mathbf{c}_y^{(i)}$  to verify from which distribution the challenge ciphertexts are generated. We write this observation formally in the following theorem.

**Theorem 11.** *Given any distribution family  $\mathbb{B}$  that  $TP - FP > \frac{1}{\text{poly}}$ . If  $\text{psk}$  is public,  $\Pi$  is not IND for any  $t \geq 0$ . If  $\text{esk}$  is public,  $\Pi$  is not IND for any  $t \geq 1$ .*

#### 5.3.1 IND Security for a Particular Biometric Layer

Recall that in Section 3.2, we introduce as an example a particular biometric layer:

$$\text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \rightarrow \mathbf{b}^* + \mathcal{E}_{\text{Enroll}} \quad \text{and} \quad \text{getProbe}^{\mathcal{O}_{\mathcal{B}}}() \rightarrow \mathbf{b}^* + \mathcal{E}_{\text{Probe}}$$

where  $\mathbf{b}^* \in \{0, 1\}^k$  is a fixed vector only dependent on  $\mathcal{B}$ , and  $\mathcal{E}_{\text{Enroll}}, \mathcal{E}_{\text{Probe}} \subseteq \{0, 1\}^k$  are some *error distributions* independent of  $\mathcal{B}$ . With the same relational hash  $RH$  in Section 5.1, we can instantiate another authentication scheme using  $RH$ .

- **Setup**( $1^\lambda$ ): It runs  $RH.\text{KeyGen}(1^\lambda) \rightarrow \text{pk}$  and samples  $\mathbf{r} \leftarrow_{\$} \{0, 1\}^k$ . Then it outputs  $\text{esk} \leftarrow (\text{pk}, \mathbf{r})$ ,  $(\text{psk} \leftarrow \text{pk}, \mathbf{r})$ , and  $\text{csk} \leftarrow \text{pk}$ .
- **Enroll**( $\text{esk}, \mathbf{b}$ ): Let  $\mathbf{x} \leftarrow \mathbf{b}$ . It calls  $RH.\text{Hash}_1(\text{pk}, \mathbf{x} + \mathbf{r}) \rightarrow \mathbf{h}_x$  and outputs  $\mathbf{c}_x \leftarrow \mathbf{h}_x$ .

- $\text{Probe}(\text{psk}, \mathbf{b}')$ : Let  $\mathbf{y} \leftarrow \mathbf{b}$ . It calls  $\text{RH.Hash}_2(\text{pk}, \mathbf{y} + \mathbf{r}) \rightarrow \mathbf{h}_{\mathbf{y}}$  and outputs  $\mathbf{c}_{\mathbf{y}} \leftarrow \mathbf{h}_{\mathbf{y}}$ .
- $\text{Compare}(\text{csk}, \mathbf{c}_{\mathbf{x}}, \mathbf{c}_{\mathbf{y}})$ : It calls  $\text{RH.Verify}(\text{pk}, \mathbf{h}_{\mathbf{x}}, \mathbf{h}_{\mathbf{y}}) \rightarrow s$  and outputs the value  $s$ .

Correctness holds because

$$\text{Compare}(\text{csk}, \mathbf{c}_{\mathbf{x}}, \mathbf{c}_{\mathbf{y}}) = 1 \Leftrightarrow \text{HD}(\mathbf{x} + \mathbf{r}, \mathbf{y} + \mathbf{r}) \leq \tau \Leftrightarrow \text{HD}(\mathbf{x}, \mathbf{y}) \leq \tau = \text{BioCompare}(\mathbf{b}, \mathbf{b}').$$

With the same argument in Theorem 2, one can prove that this new scheme is now IND, albeit at the cost of requiring  $\text{esk}$  and  $\text{psk}$  to remain secret.



## A Construction in [Kim+16]

Let  $\mathbb{G}_1$  and  $\mathbb{G}_2$  be two groups of order a prime number  $q$  with generators  $g_1$  and  $g_2$ , respectively. Let  $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$  be a mapping to a target group  $\mathbb{G}_T$  also of order  $q$ .

**Definition 11** (Bilinear asymmetric group [Kim+16]). A tuple  $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, e)$  is a *bilinear asymmetric group* if the following hold.

- Group operations in  $\mathbb{G}_1, \mathbb{G}_2$ , and  $\mathbb{G}_T$  and mapping  $e$  are efficiently computable.
- $e$  is bilinear. That is, for  $x, y \in \mathbb{Z}_q$ ,  $e(g_1^x, g_2^y) = e(g_1, g_2)^{xy}$ .
- $e$  is non-degenerate. That is,  $e(g_1, g_2) \neq 1$ , the identity element of  $\mathbb{G}_T$ .

For a vector  $\mathbf{v} = (v_1, v_2, \dots, v_n) \in \mathbb{Z}_q^n$  and a group element  $g$  in group of order  $q$ , we write  $g^{\mathbf{v}}$  to denote the vector of group elements  $(g^{v_1}, g^{v_2}, \dots, g^{v_n})$ . Moreover, for  $k \in \mathbb{Z}_q$  and  $\mathbf{v}, \mathbf{w} \in \mathbb{Z}_q^n$ , we write  $(g^{\mathbf{v}})^k = g^{k \cdot \mathbf{v}}$  and  $g^{\mathbf{v}} \cdot g^{\mathbf{w}} = g^{\mathbf{v} + \mathbf{w}}$ . Finally, the pairing operation is extended to vectors.

$$e(g_1^{\mathbf{v}}, g_2^{\mathbf{w}}) = \prod_{i \in [n]} e(g_1^{v_i}, g_2^{w_i}) = e(g_1, g_2)^{\mathbf{v}\mathbf{w}^T}.$$

We now recall the fh-IPFE construction FE in [Kim+16].

- **FE.Setup**( $1^\lambda$ ): Sample an asymmetric bilinear group  $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, e)$  and choose generators  $g_1 \in \mathbb{G}_1$  and  $g_2 \in \mathbb{G}_2$ . Sample  $\mathbf{B} \in \mathbb{GL}_n(\mathbb{Z}_q)$  and find  $\mathbf{B}^* = \det(\mathbf{B}) \cdot (\mathbf{B}^{-1})^T$ . Finally, output the public parameter  $\mathbf{pp} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, e)$  and the master secret key  $\mathbf{msk} = (\mathbf{pp}, g_1, g_2, \mathbf{B}, \mathbf{B}^*)$ .
- **FE.KeyGen**( $\mathbf{msk}, \mathbf{pp}, \mathbf{x}$ ): Sample  $\alpha \leftarrow \mathbb{Z}_q$  and output

$$f_{\mathbf{x}} = (K_1, K_2) = (g_1^{\alpha \cdot \det(\mathbf{B})}, g_1^{\alpha \cdot \mathbf{x} \cdot \mathbf{B}})$$

- **FE.Enc**( $\mathbf{msk}, \mathbf{pp}, \mathbf{y}$ ): Sample  $\beta \leftarrow \mathbb{Z}_q$  and output

$$\mathbf{c}_{\mathbf{y}} = (C_1, C_2) = (g_2^\beta, g_2^{\beta \cdot \mathbf{y} \cdot \mathbf{B}^*})$$

- **FE.Dec**( $\mathbf{pp}, f_{\mathbf{x}}, \mathbf{c}_{\mathbf{y}}$ )  $\rightarrow z$ : Parse  $f_{\mathbf{x}} = (K_1, K_2)$  and  $\mathbf{c}_{\mathbf{y}} = (C_1, C_2)$  and compute

$$D_1 = e(K_1, C_1) \quad \text{and} \quad D_2 = e(K_2, C_2)$$

Solve the discrete logarithm to find  $z$  such that  $D_1^z = D_2$  and output  $z$ . If it fails to find such  $z$ , output  $\perp$ .

**Correctness** We have

$$D_1 = e(K_1, C_1) = e(g_1, g_2)^{\alpha \cdot \beta \cdot \det(\mathbf{B})}$$

and

$$D_2 = e(K_2, C_2) = e(g_1, g_2)^{\alpha \cdot \beta \cdot \mathbf{x} \cdot \mathbf{B} \cdot (\mathbf{B}^*)^T \cdot \mathbf{y}^T} = e(g_1, g_2)^{\alpha \cdot \beta \cdot \det(\mathbf{B}) \cdot \mathbf{x}\mathbf{y}^T}.$$

Therefore,  $(D_1)^{\mathbf{x}\mathbf{y}^T} = D_2$ .

**Remark** In this construction,  $q$  is exponential to  $\lambda$  to achieve security, and decryption relies on some priori knowledge of possible ranges of the inner product  $\mathbf{xy}^T$ . For example, for the instantiation in Section 4.1, one can enumerate  $z \in \{0, 1, \dots, \tau\}$  and return  $\perp$  when no valid  $z \leq \tau$  such that  $D_1^z = D_2$  is found.

## B $\gamma$ -RUF Security of FE

Let  $\mathbb{F} = \mathbb{Z}_q$ . We can extend the definition of the RUF security with an integer parameter  $\gamma$ .

---

**Algorithm 13**  $\text{RUF}_{\text{FE}}^{\mathcal{O}, \gamma}(\mathcal{A})$ 


---

```

1:  $\mathbf{r} \leftarrow \mathbb{F}^k$ 
2:  $\text{msk}, \text{pp} \leftarrow \text{FE.Setup}(1^\lambda)$ 
3:  $\mathbf{c} \leftarrow \text{FE.KeyGen}(\text{msk}, \text{pp}, \mathbf{r})$ 
4:  $\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}}(\text{pp}, \mathbf{c})$ 
5: if  $\tilde{\mathbf{z}}$  is equal to any output of  $\mathcal{O}'_{\text{Enc}}$  then
6:   return 0
7: end if
8:  $s \leftarrow \text{FE.Dec}(\text{pp}, \mathbf{c}, \tilde{\mathbf{z}})$ 
9: return  $1_{s \leq \gamma}$ 

```

---

Here, the game runs  $1_{s \leq \gamma}$  by first viewing the field element  $s \in \mathbb{Z}_q$  as a positive integer in  $\{0, 1, \dots, q-1\}$  and comparing it with  $\gamma$ .

The oracle  $\mathcal{O}$  can be nothing or include  $\mathcal{O}'_{\text{KeyGen}}(\cdot)$  and  $\mathcal{O}'_{\text{Enc}}(\cdot)$  based on the threat model as in Section 4.3.

**Definition 12** ( $\gamma$ -RUF Security). An fh-IPFE scheme FE is called  $\{\mathcal{O}, \gamma\}$ -RUF secure if for any adversary  $\mathcal{A}$ , the advantage of  $\mathcal{A}$  in the  $\text{RUF}_{\text{FE}}^{\mathcal{O}, \gamma}$  game in Algorithm 13 is

$$\text{Adv}_{\text{FE}, \mathcal{A}}^{\text{RUF}, \mathcal{O}, \gamma} := \Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}, \gamma}(\mathcal{A}) \rightarrow 1] = \text{negl}.$$

Note that if FE is  $\mathcal{O}$ -RUF secure, it is  $\{\mathcal{O}, \gamma\}$ -RUF secure for any integer  $\gamma$ .

With the extension with  $\gamma$ , we can rewrite our results in Section 4.

**Theorem 12** (Theorem 5). *Let  $\text{option} = \{\mathbf{c}_x, \text{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\text{Enroll}}\}$ . For any distribution family  $\mathbb{B}$ , if FE is fh-IND and  $\{\mathcal{O}'_{\text{KeyGen}}, \gamma\}$ -RUF for a  $\gamma \geq \tau^2$ , then  $\Pi$  is **option-UF**.*

**Theorem 13** (Theorem 6). *Let  $\text{option} = \{\mathbf{c}_x, \text{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\text{Probe}}\}$ . For any distribution family  $\mathbb{B}$ , if FE is fh-IND and  $\{\mathcal{O}'_{\text{Enc}}, \gamma\}$ -RUF for a  $\gamma \geq \tau^2$ , then  $\Pi$  is **option-UF**.*

### B.1 Achievability of $\gamma$ -RUF Security

**Assumption 2.** Assume that  $\text{FE.Dec}(\text{pp}, \mathbf{c}, \mathbf{z})$  only returns when  $\mathbf{z}$  corresponds to a *nonzero* vector  $\mathbf{v} \in \mathbb{F}^k$ . More precisely, assume that for any  $\mathbf{z}$ , there can only be two possibilities.

- Either there exists a vector  $\mathbf{v} \in \mathbb{F}^k \setminus \{\mathbf{0}\}$  such that for any  $\mathbf{x} \in \mathbb{F}^k, \mathbf{c} \leftarrow \text{FE.KeyGen}(\text{msk}, \text{pp}, \mathbf{x})$ ,

$$\text{FE.Dec}(\text{pp}, \mathbf{c}, \mathbf{z}) = \mathbf{x}\mathbf{v}^T.$$

- Or for any  $\mathbf{x} \in \mathbb{F}^k$  and  $\mathbf{c} \leftarrow \text{FE.KeyGen}(\text{msk}, \text{pp}, \mathbf{x})$ ,  $\text{FE.Dec}(\text{pp}, \mathbf{c}, \mathbf{z}) \rightarrow \perp$ .

Note that this implies FE rejects zero vector  $\mathbf{0}$  as the input of FE.Enc.

**Theorem 14.** *Given Assumption 2. If FE is fh-IND, then FE is  $\{\mathcal{O}'_{\text{KeyGen}}, \gamma\}$ -RUF for any  $\gamma \leq (1 - \frac{1}{\text{poly}}) \cdot \|\mathbb{F}\|$ .*

*Proof.* Given an adversary  $\mathcal{A}$  in the  $\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{KeyGen}}, \gamma}$  game for a  $\gamma \leq (1 - \frac{1}{P(\lambda)}) \cdot \|\mathbb{F}\|$ , where  $P(\lambda)$  is any polynomial. Let  $t$  be an integer, consider the reduction adversary  $\mathcal{R}$  in Algorithm 14 which plays the fh-IND game.  $\mathcal{R}$  simulates  $\mathcal{O}'_{\text{KeyGen}}(\mathbf{x}')$  by  $\mathcal{O}_{\text{KeyGen}}(\mathbf{x}', \mathbf{x}')$ . If there exists an  $s_i \neq \perp$  in Line 7, by Assumption 2, let  $\tilde{\mathbf{z}}$  correspond to a nonzero vector  $\tilde{\mathbf{v}}$ .

---

**Algorithm 14**  $\mathcal{R}^{\mathcal{O}_{\text{KeyGen}}, \mathcal{O}_{\text{Enc}}}(\text{pp})$

---

```

1:  $\mathbf{r}^{(0)}, \mathbf{r}^{(1)} \leftarrow_{\$} \mathbb{F}^k$ 
2:  $\mathbf{c} \leftarrow \mathcal{O}_{\text{KeyGen}}(\mathbf{r}^{(0)}, \mathbf{r}^{(1)})$ 
3:  $\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}'_{\text{KeyGen}}}(\text{pp}, \mathbf{c})$ 
4: for  $i = 1$  to  $t$  do
5:    $\mathbf{r}_i \leftarrow_{\$} \mathbb{F}^k$ 
6:    $\mathbf{c}_i \leftarrow \mathcal{O}_{\text{KeyGen}}(\mathbf{r}^{(0)}, \mathbf{r}_i)$ 
7:    $s_i \leftarrow \text{FE.Dec}(\text{pp}, \mathbf{c}_i, \tilde{\mathbf{z}})$ 
8: end for
9: if  $\bigwedge_{i=1}^t s_i \leq \gamma$  then
10:   return  $\tilde{b} = 0$ 
11: else
12:   return  $\tilde{b} \leftarrow_{\$} \{0, 1\}$ 
13: end if
```

---

If the challenge bit  $b = 0$ , then by Assumption 2, any  $s_i \neq \perp$  in Line 7 implies all  $s_i \neq \perp$  and  $s_i = s_j$  for any  $i, j$ . Therefore, the probability that all  $s_i \leq \gamma$  in Line 9 is

$$\begin{aligned}
\Pr \left[ \bigwedge_{i=1}^t s_i \leq \gamma \mid b = 0 \right] &= \Pr[s_1 \neq \perp \mid b = 0] \cdot \Pr[s_1 \leq \gamma \mid b = 0 \wedge s_1 \neq \perp] \\
&= \Pr[s_1 \neq \perp \mid b = 0] \cdot \Pr[\mathbf{r}^{(0)} \tilde{\mathbf{v}}^T \leq \gamma \mid b = 0 \wedge s_1 \neq \perp] \\
&= \Pr[s_1 \neq \perp \mid b = 0] \cdot \Pr[\text{FE.Dec}(\text{pp}, \mathbf{c}, \tilde{\mathbf{z}}) \leq \gamma \mid b = 0 \wedge s_1 \neq \perp] \\
&= \Pr[s_1 \neq \perp \mid b = 0] \cdot \Pr[\text{RUF}_{\text{KeyGen}}^{\mathcal{O}'_{\text{KeyGen}}, \gamma}(\mathcal{A}) \rightarrow 1 \mid b = 0 \wedge s_1 \neq \perp] \\
&= \Pr[\text{RUF}_{\text{KeyGen}}^{\mathcal{O}'_{\text{KeyGen}}, \gamma}(\mathcal{A}) \rightarrow 1]
\end{aligned}$$

If the challenge bit  $b = 1$ , for any  $i \in [t]$ ,

$$\begin{aligned}
\Pr[s_i \leq \gamma \mid b = 1] &= \Pr[s_i \neq \perp \mid b = 1] \cdot \Pr[s_i \leq \gamma \mid b = 1 \wedge s_i \neq \perp] \\
&= \Pr[s_i \neq \perp \mid b = 1] \cdot \Pr[\mathbf{r}_i \tilde{\mathbf{v}}^T \leq \gamma \mid b = 1 \wedge s_i \neq \perp]
\end{aligned}$$

Note that  $\mathbf{r}_i$  is independent of  $\tilde{\mathbf{z}}$  and thus independent of  $\tilde{\mathbf{v}}$ . Hence,  $\Pr[\mathbf{r}_i \tilde{\mathbf{v}}^T \leq \gamma \mid b = 1 \wedge s_i \neq \perp] = \frac{\gamma}{\|\mathbb{F}\|}$  and

$$\Pr \left[ \bigwedge_{i=1}^t s_i \leq \gamma \mid b = 1 \right] = \Pr \left[ \bigwedge_{i=1}^t s_i \neq \perp \mid b = 1 \right] \cdot \left( \frac{\gamma}{\|\mathbb{F}\|} \right)^t \leq \left( \frac{\gamma}{\|\mathbb{F}\|} \right)^t$$

In conclusion,

$$\begin{aligned} \Pr[\text{fh-IND}(\mathcal{R}) \rightarrow 1] &= \frac{1}{2} + \frac{1}{4} \left( \Pr \left[ \bigwedge_{i=1}^t s_i \leq \gamma \mid b = 0 \right] - \Pr \left[ \bigwedge_{i=1}^t s_i \leq \gamma \mid b = 1 \right] \right) \\ &\geq \frac{1}{2} + \frac{1}{4} \left( \Pr[\text{RUF}^{\mathcal{O}'_{\text{KeyGen}, \gamma}}(\mathcal{A}) \rightarrow 1] - \left( \frac{\gamma}{\|\mathbb{F}\|} \right)^t \right) \\ &\geq \frac{1}{2} + \frac{1}{4} \left( \Pr[\text{RUF}^{\mathcal{O}'_{\text{KeyGen}, \gamma}}(\mathcal{A}) \rightarrow 1] - e^{-t \cdot (1 - \frac{\gamma}{\|\mathbb{F}\|})} \right) \\ &\geq \frac{1}{2} + \frac{1}{4} \left( \Pr[\text{RUF}^{\mathcal{O}'_{\text{KeyGen}, \gamma}}(\mathcal{A}) \rightarrow 1] - e^{-\frac{t}{P(\lambda)}} \right) \end{aligned}$$

Take  $t$  be any integer larger than  $P(\lambda) \cdot \lambda$ . Since  $\mathbf{Adv}_{\text{FE}, \mathcal{R}}^{\text{fh-IND}} = |\Pr[\text{fh-IND}(\mathcal{R}) \rightarrow 1] - \frac{1}{2}|$  and  $e^{-\frac{1}{P(\lambda)}} < e^{-\lambda}$  are negligible,

$$\Pr[\text{RUF}^{\mathcal{O}'_{\text{KeyGen}, \gamma}}(\mathcal{A}) \rightarrow 1] \leq e^{-\frac{t}{P(\lambda)}} + 4 \cdot \mathbf{Adv}_{\text{FE}, \mathcal{R}}^{\text{fh-IND}} = \text{negl}.$$

□

## C One-Way Game

---

**Algorithm 15**  $\text{OW}_{\Pi, \mathbb{B}}(\mathcal{A})$ 


---

```

1:  $\mathcal{B} \leftarrow_{\$} \mathbb{B}, \quad \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$ 
2:  $\text{esk}, \text{psk}, \text{csk} \leftarrow \text{Setup}(1^\lambda)$ 
3:  $\mathbf{b} \leftarrow \text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ 
4:  $\mathbf{c}_{\mathbf{x}} \leftarrow \text{Enroll}(\text{esk}, \mathbf{b})$ 
5:  $\tilde{\mathbf{b}} \leftarrow \mathcal{A}(\text{option})$ 
6:  $s \leftarrow \text{BioCompare}(\tilde{\mathbf{b}}, \mathbf{b})$ 
7: return  $\text{Verify}(s)$ 

```

---

option can be  $\mathbf{c}_{\mathbf{x}}, \text{esk}, \text{psk}, \text{csk}$  or oracles  $\mathcal{O}_{\text{Enroll}}, \mathcal{O}_{\text{Probe}}, \mathcal{O}'_{\text{Enroll}}, \mathcal{O}'_{\text{Probe}}$ .

In the reusable fuzzy extractor paper [Boy04], if public strings related to personal information are generated many times, the sensitive information might leak. In our setting, it is a little similar to having multiple  $\mathbf{c}_{\mathbf{x}}^{(i)} \leftarrow \text{Enroll}(\text{esk}_i, \mathbf{b})$ .

To model reusability, I think we can consider either providing  $\mathbf{c}_{\mathbf{x}}^{(i)} \leftarrow \text{Enroll}(\text{esk}_i, \mathbf{b})$  for different  $\text{esk}_i$ , or just providing  $\mathcal{O}'_{\text{Enroll}}$ .

## D Fixing the Proof of Theorem 6

In Theorem 6, we simulate the oracle  $\mathcal{O}_{\text{Probe}}$  in the adversary  $\mathcal{A}'$  in Algorithm 10 in the  $\text{RUF}^{\mathcal{O}'_{\text{Enc}}}$  game by the following steps:

1. Sample  $k + 2$  independent vectors  $\mathbf{e}^{(1)}, \dots, \mathbf{e}^{(k+2)}$ .
2. For  $i \in [k + 2]$ ,  $\mathbf{c}^{(i)} \leftarrow \mathcal{O}'_{\text{Enc}}(\mathbf{e}^{(i)})$ .
3. For  $i \in [k + 2]$ ,  $d_i \leftarrow \text{FE.Dec}(\text{pp}, \mathbf{c}, \mathbf{c}^{(i)})$ , where  $\mathbf{c}$  is  $\text{FE.KeyGen}(\text{msk}, \text{pp}, \mathbf{r})$ .
4. Find the vector  $\mathbf{r}$  by solving the linear system  $\{\mathbf{r}\mathbf{e}^{(i)T} = d_i\}_{i=1}^{k+2}$ .
5. On query  $\mathcal{O}_{\text{Probe}}(\mathbf{b}')$ , first encode  $\mathbf{b}'$  into  $\mathbf{y}'$  and find  $d \leftarrow \mathbf{x}^* \mathbf{y}'^T$ . Then find a vector  $\mathbf{y}''$  such that  $\mathbf{r}\mathbf{y}''^T = d$ . Return  $\mathcal{O}'_{\text{Enc}}(\mathbf{y}'')$ .

However,  $\text{FE.Dec}$  of constructions in [DDM15; TAO16; Kim+16] rely on some *prior knowledge* of the inner product. Their basic idea is to find  $d$  given  $g$  and  $g^d$ , where  $g$  is in a group of exponential size. Therefore,  $\text{FE.Dec}(\text{pp}, \mathbf{c}, \mathbf{c}^{(i)})$  will probably return  $\perp$ , representing that  $d$  is too large to find.

One way to fix this is by modifying the RUF game. Instead of sampling  $\mathbf{r} \leftarrow_{\$} \mathbb{F}^k$ , one samples  $\mathbf{r} \leftarrow_{\$} \{0, 1\}^{k+2}$ .

1. Pick  $k + 2$  random one-hot vectors  $\mathbf{e}^{(i)} \leftarrow (0, \dots, \overset{\text{ith}}{u_i}, \dots, 0)$ , where  $u \leftarrow_{\$} \mathbb{F}$ .
2. For  $i \in [k + 2]$ ,  $\mathbf{c}^{(i)} \leftarrow \mathcal{O}'_{\text{Enc}}(\mathbf{e}^{(i)})$ .
3. For  $i \in [k + 2]$ ,  $d_i \leftarrow \text{FE.Dec}(\text{pp}, \mathbf{c}, \mathbf{c}^{(i)})$ . Note that  $d_i$  can only be  $u_i$  or 0.

4. Find the vector  $\mathbf{r} \in \{0, 1\}^{k+2}$ .
5. On query  $\mathcal{O}_{\text{Probe}}(\mathbf{b}')$ , first encode  $\mathbf{b}'$  into  $\mathbf{y}'$  and find  $d \leftarrow \mathbf{x}^* \mathbf{y}'^T$ . Then find a vector  $\mathbf{y}''$  such that  $\mathbf{r} \mathbf{y}''^T = d$ . Return  $\mathcal{O}'_{\text{Enc}}(\mathbf{y}'')$ .

Another way is to provide  $\mathbf{r}$  in the RUF game.

We can say these fh-IPFE constructions are not *ideal* in the sense that FE.Dec aborts without further restrictions on input vectors. Constructions in [Che+21] do not need discrete logarithm in FE.Dec. Unfortunately, it is also not fh-IND secure.

One direction now is to find an fh-IPFE such that

- $q = \text{poly}(\lambda)$ . Otherwise, the scheme is very slow.
- FE.Dec does not need prior knowledge of inputs. Otherwise, we need to change RUF game.
- Full fh-IND security.

## E Achievability of $\mathcal{O}'_{\text{Enc}}$ -RUF Security

**Assumption 3.** Let FE be an fh-IPFE scheme and  $(\text{msk}, \text{pp}) \leftarrow \text{FE.Setup}(1^\lambda)$ . Assume that given  $\text{pp}$ , there exist PPT algorithms RandKeyGen and RandEnc that can generate  $\text{FE.KeyGen}(\text{msk}, \mathbf{r})$  and  $\text{FE.Enc}(\text{msk}, \mathbf{r})$  for some random vector  $\mathbf{r} \in \mathbb{F}^k$ , respectively.

Note that constructions in [TAO16; Kim+16] satisfy Assumption 3.

**Theorem 15.** *Given Assumption 3. If FE is fh-IND and  $\emptyset$ -RUF, then FE is also  $\mathcal{O}'_{\text{Enc}}$ -RUF.*

*Proof.* Given an adversary  $\mathcal{A}$  in the  $\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}$  game, consider the reduction adversary  $\mathcal{R}$  in Algorithm 16 which plays the fh-IND game.  $\mathcal{R}$  simulates  $\mathcal{O}'_{\text{Enc}}(\mathbf{y}')$  by first sampling a  $\mathbf{r}' \leftarrow \mathbb{F}^k$  and returning  $\mathcal{O}_{\text{Enc}}(\mathbf{y}', \mathbf{r}')$ .

---

### Algorithm 16 $\mathcal{R}^{\mathcal{O}_{\text{KeyGen}}, \mathcal{O}_{\text{Enc}}}(\text{pp})$

---

```

1:  $\mathbf{c} \leftarrow \text{RandKeyGen}(\text{pp})$ .
2:  $\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}'_{\text{Enc}}}(\text{pp}, \mathbf{c})$ 
3:  $s \leftarrow \text{FE.Dec}(\text{pp}, \mathbf{c}, \tilde{\mathbf{z}})$ 
4: if  $s \neq \perp$  then
5:   return  $\tilde{b} = 0$ 
6: else
7:   return  $\tilde{b} = 1$ 
8: end if
```

---



---

### Algorithm 17 $\mathcal{A}'(\text{pp})$

---

```

1:  $\mathbf{c} \leftarrow \text{RandKeyGen}(\text{pp})$ .
2:  $\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}'_{\text{Enc}}}(\text{pp}, \mathbf{c})$ 
3: return  $\tilde{\mathbf{z}}$ 
```

---

By Assumption 3,  $\mathbf{c}$  looks like an honest key of some random vector  $\mathbf{r}$ . If the challenge bit  $b = 0$ ,  $\mathcal{R}$  perfectly simulates an  $\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}$  game for  $\mathcal{A}$  and  $\Pr[\tilde{b} = 0 \mid$

$b = 0] = \Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}(\mathcal{A}) \rightarrow 1]$ . On the other hand, if the challenge bit  $b = 1$ , then  $\mathcal{R}$  simulates an  $\text{RUF}_{\text{FE}}^{\emptyset}$  adversary  $\mathcal{A}'$  in Algorithm 17.  $\mathcal{A}'$  runs  $\mathcal{A}$  and simulates  $\mathcal{O}'_{\text{Enc}}(\mathbf{y}')$  by simply returning  $\text{RandEnc}(\text{pp})$ . Therefore,  $\Pr[\tilde{b} = 0 \mid b = 1] = \Pr[\text{RUF}_{\text{FE}}^{\emptyset}(\mathcal{A}') \rightarrow 1]$ .

In conclusion,

$$\begin{aligned} \Pr[\text{fh-IND}(\mathcal{R}) \rightarrow 1] &= \Pr[b = 0] \cdot \Pr[\tilde{b} = 0 \mid b = 0] + \Pr[b = 1] \cdot \Pr[\tilde{b} = 1 \mid b = 1] \\ &= \frac{1}{2} \left( \Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}(\mathcal{A}) \rightarrow 1] + 1 - \Pr[\text{RUF}_{\text{FE}}^{\emptyset}(\mathcal{A}') \rightarrow 1] \right) \\ &= \frac{1}{2} + \frac{1}{2} \left( \Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}(\mathcal{A}) \rightarrow 1] - \Pr[\text{RUF}_{\text{FE}}^{\emptyset}(\mathcal{A}') \rightarrow 1] \right) \end{aligned}$$

Since  $\mathbf{Adv}_{\text{FE}, \mathcal{R}}^{\text{fh-IND}} = \left| \Pr[\text{fh-IND}(\mathcal{R}) \rightarrow 1] - \frac{1}{2} \right|$  and  $\mathbf{Adv}_{\text{FE}, \mathcal{A}'}^{\text{RUF}, \emptyset} = \Pr[\text{RUF}_{\text{FE}}^{\emptyset}(\mathcal{A}') \rightarrow 1]$  is negligible,

$$\Pr[\text{RUF}_{\text{FE}}^{\mathcal{O}'_{\text{Enc}}}(\mathcal{A}) \rightarrow 1] = \mathbf{Adv}_{\text{FE}, \mathcal{R}}^{\text{fh-IND}} + \mathbf{Adv}_{\text{FE}, \mathcal{A}'}^{\text{RUF}, \emptyset} = \text{negl}.$$

□

With a similar proof, one can also show that

**Theorem 16.** *Given Assumption 3. If FE is fh-IND and  $\{\emptyset, \gamma\}$ -RUF, then FE is also  $\{\mathcal{O}'_{\text{Enc}}, \gamma\}$ -RUF.*

Since  $\{\mathcal{O}'_{\text{KeyGen}}, \gamma\}$ -RUF implies  $\{\emptyset, \gamma\}$ -RUF, with Theorem 14, we have the following corollary.

**Corollary 1.** *Given Assumption 2 and 3. If FE is fh-IND, then FE is both  $\{\mathcal{O}'_{\text{KeyGen}}, \gamma\}$ -RUF and  $\{\mathcal{O}'_{\text{Enc}}, \gamma\}$ -RUF for any  $\gamma \leq (1 - \frac{1}{\text{poly}}) \cdot \|\mathbb{F}\|$ .*



## References

- [Boy04] Xavier Boyen. “Reusable cryptographic fuzzy extractors”. In: *Proceedings of the 11th ACM Conference on Computer and Communications Security*. CCS '04. Washington DC, USA: Association for Computing Machinery, 2004, pp. 82–91. ISBN: 1581139616. DOI: [10.1145/1030083.1030096](https://doi.org/10.1145/1030083.1030096). URL: <https://doi.org/10.1145/1030083.1030096>.
- [BCP12] Julien Bringer, Herve Chabanne, and Alain Patey. *SHADE: Secure HAMming Distance computation from oblivious transfer*. Cryptology ePrint Archive, Paper 2012/586. 2012. URL: <https://eprint.iacr.org/2012/586>.
- [Yas+13] Masaya Yasuda et al. “Packed Homomorphic Encryption Based on Ideal Lattices and Its Application to Biometrics”. In: *Security Engineering and Intelligence Informatics*. Ed. by Alfredo Cuzzocrea et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 55–74. ISBN: 978-3-642-40588-4.
- [MR14] Avradip Mandal and Arnab Roy. *Relational Hash*. Cryptology ePrint Archive, Paper 2014/394. 2014. URL: <https://eprint.iacr.org/2014/394>.
- [DDM15] Pratish Datta, Ratna Dutta, and Sourav Mukhopadhyay. *Functional Encryption for Inner Product with Full Function Privacy*. Cryptology ePrint Archive, Paper 2015/1255. 2015. URL: <https://eprint.iacr.org/2015/1255>.
- [Kim+16] Sam Kim et al. *Function-Hiding Inner Product Encryption is Practical*. Cryptology ePrint Archive, Paper 2016/440. 2016. URL: <https://eprint.iacr.org/2016/440>.
- [TAO16] Junichi Tomida, Masayuki Abe, and Tatsuaki Okamoto. “Efficient Functional Encryption for Inner-Product Values with Full-Hiding Security”. In: *Information Security*. Ed. by Matt Bishop and Anderson C A Nascimento. Cham: Springer International Publishing, 2016, pp. 408–425. ISBN: 978-3-319-45871-7.
- [Li+17] Nan Li et al. “Fuzzy Extractors for Biometric Identification”. In: *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. 2017, pp. 667–677. DOI: [10.1109/ICDCS.2017.107](https://doi.org/10.1109/ICDCS.2017.107).
- [Lee+18] Joohee Lee et al. *Instant Privacy-Preserving Biometric Authentication for Hamming Distance*. Cryptology ePrint Archive, Paper 2018/1214. 2018. URL: <https://eprint.iacr.org/2018/1214>.
- [Che+21] Jung Hee Cheon et al. “Lattice-Based Secure Biometric Authentication for Hamming Distance”. In: *Information Security and Privacy*. Ed. by Joonsang Baek and Sushmita Ruj. Cham: Springer International Publishing, 2021, pp. 653–672. ISBN: 978-3-030-90567-5.
- [PM21] Gaëtan Pradel and Chris Mitchell. *Privacy-Preserving Biometric Matching Using Homomorphic Encryption*. 2021. arXiv: [2111.12372](https://arxiv.org/abs/2111.12372) [cs.CR]. URL: <https://arxiv.org/abs/2111.12372>.

- [Cac+22] Chloe Cachet et al. “Proximity Searchable Encryption for the Iris Biometric”. In: *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*. ASIA CCS ’22. Nagasaki, Japan: Association for Computing Machinery, 2022, pp. 1004–1018. ISBN: 9781450391405. DOI: [10.1145/3488932.3497754](https://doi.org/10.1145/3488932.3497754). URL: <https://doi.org/10.1145/3488932.3497754>.
- [PP22] Paola de Perthuis and David Pointcheval. *Two-Client Inner-Product Functional Encryption, with an Application to Money-Laundering Detection*. Cryptology ePrint Archive, Paper 2022/441. 2022. DOI: [10.1145/3548606.3559374](https://eprint.iacr.org/2022/441). URL: <https://eprint.iacr.org/2022/441>.
- [EM23] Johannes Ernst and Aikaterini Mitrokotsa. *A Framework for UC Secure Privacy Preserving Biometric Authentication using Efficient Functional Encryption*. Cryptology ePrint Archive, Paper 2023/481. 2023. URL: <https://eprint.iacr.org/2023/481>.