Biometrics Authentication: Formalization and Instantiation

Keng-Yu Chen

December 27, 2024

This project formalizes the biometric authentication scheme, including its structure, usage, and security analysis with a security game model.

1 Preliminaries

In this project, we assume

- λ is the security parameter.
- $[m]$ denotes the set of integers $\{1, 2, \cdots, m\}$.
- $\mathbb{Z}_q$ 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 $f(n) = \mathsf{negl}$, and we may also use negl to represent an arbitrary negligible function.
- poly is the class of polynomial funcions. We may also use poly to represent an arbitrary polynomial function.
- We write sampling a value r from a distribution $\mathcal{D}$ as $r \leftarrow_\$ \mathcal{D}$. If S is a finite set, then $r \leftarrow_\$ S$ 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.

We introduce three types of inner product functional encryption schemes: function hiding functional encryption, two-input functional encryption, and two-client functional encryption. We will instantiate our biometric authentication scheme using these primitives.

Definition 1 (Function Hiding Inner Product Functional Encryption)**.** 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 $\mathsf{FE.Setup}$, $\mathsf{FE.KeyGen}$, $\mathsf{FE.Enc}$, and $\mathsf{FE.Dec}$:

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

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

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

Instantiation using an fh-IPFE scheme is given in Section 2.3.

Definition 2 (Two-Input Inner Product Functional Encryption (adapted from [PP22]))**.** A *two-input inner product functional encryption* (2i-IPFE) scheme FE for a field $\mathbb{F}$ and input length k is composed of PPT algorithms $\mathsf{FE.Setup}$, $\mathsf{FE.KeyGen}$, $\mathsf{FE.Enc}$, and $\mathsf{FE.Dec}$:

- $\mathsf{FE.Setup}(1^\lambda) \to \mathsf{sk}, \mathsf{ek}_1, \mathsf{ek}_2$: It outputs a secret key sk and two encryption keys $\mathsf{ek}_1, \mathsf{ek}_2$.
- $\mathsf{FE.KeyGen}(\mathsf{sk}, \mathbf{A}) \to \mathsf{dk}_{\mathbf{A}}$: It generates the functional decryption key $\mathsf{dk}_{\mathbf{A}}$ for a diagonal matrix $\mathbf{A} \in \mathbb{F}^{k\times k}$,
- $\mathsf{FE.Enc}(\mathsf{ek}_i, \mathbf{x}) \to \mathbf{c}_{\mathbf{x}}$: Given an encryption key, either ek_1 or ek_2, it encrypts the input vector $\mathbf{x} \in \mathbb{F}^k$ to the ciphertext $\mathbf{c}_{\mathbf{x}}$.
- $\mathsf{FE.Dec}(\mathsf{dk}_{\mathbf{A}}, \mathbf{c}_{\mathbf{x}}, \mathbf{c}_{\mathbf{y}}) \to z$: It outputs a value $z \in \mathbb{F}$.

Correctness: A 2i-IPFE scheme FE is *correct* if $\forall(\mathsf{sk}, \mathsf{ek}_1, \mathsf{ek}_2) \leftarrow \mathsf{FE.Setup}(1^\lambda)$, $\mathbf{A} \in \mathbb{F}^{k\times k}$, and $\mathbf{x}, \mathbf{y} \in \mathbb{F}^k$, we have

$$\mathsf{FE.Dec}(\mathsf{FE.KeyGen}(\mathsf{sk}, \mathbf{A}), \mathsf{FE.Enc}(\mathsf{ek}_1, \mathbf{x}), \mathsf{FE.Enc}(\mathsf{ek}_2, \mathbf{y})) = \mathbf{x}\mathbf{A}\mathbf{y}^T \in \mathbb{F}.$$

Instantiation using a 2i-IPFE is given in Section 2.4.

Definition 3 (Two-Client Inner Product Functional Encryption (adapted from [PP22]))**.** A *two-client inner product functional encryption* (2c-IPFE) scheme FE for a field $\mathbb{F}$ and input length k is composed of PPT algorithms $\mathsf{FE.Setup}$, $\mathsf{FE.KeyGen}$, $\mathsf{FE.Enc}$, and $\mathsf{FE.Dec}$:

- $\mathsf{FE.Setup}(1^\lambda) \to \mathsf{sk}, \mathsf{ek}_1, \mathsf{ek}_2$: It outputs a secret key sk and two encryption keys $\mathsf{ek}_1, \mathsf{ek}_2$.
- $\mathsf{FE.KeyGen}(\mathsf{sk}, \mathbf{A}) \to \mathsf{dk}_{\mathbf{A}}$: It generates the functional decryption key $\mathsf{dk}_{\mathbf{A}}$ for a diagonal matrix $\mathbf{A} \in \mathbb{F}^{k\times k}$,

- $\mathsf{FE.Enc}(\ell, \mathsf{ek}_i, \mathbf{x}) \to \mathbf{c}_\mathbf{x}$: Given a label ℓ and an encryption key, either ek_1 or ek_2, it encrypts the input vector $\mathbf{x} \in \mathbb{F}^k$ to the ciphertext $\mathbf{c}_\mathbf{x}$.
- $\mathsf{FE.Dec}(\mathsf{dk}_\mathbf{A}, \mathbf{c}_\mathbf{x}, \mathbf{c}_\mathbf{y}) \to z$: It outputs a value $z \in \mathbb{F}$.

Correctness: A 2c-IPFE scheme FE is *correct* if $\forall (\mathsf{sk}, \mathsf{ek}_1, \mathsf{ek}_2) \leftarrow \mathsf{FE.Setup}(1^\lambda)$, $\mathbf{A} \in \mathbb{F}^{k \times k}$, label ℓ, and $\mathbf{x}, \mathbf{y} \in \mathbb{F}^k$, we have

$$\mathsf{FE.Dec}(\mathsf{FE.KeyGen}(\mathsf{sk}, \mathbf{A}), \mathsf{FE.Enc}(\ell, \mathsf{ek}_1, \mathbf{x}), \mathsf{FE.Enc}(\ell, \mathsf{ek}_2, \mathbf{y})) = \mathbf{x}\mathbf{A}\mathbf{y}^T \in \mathbb{F}.$$

Instantiation using a 2c-IPFE is given in Section 2.5.
We also consider an instantiation using a relational hash scheme.

Definition 4 (Relational Hash (adapted from [MR14]))**.** Let R_λ be a relation over sets X_λ, Y_λ, and Z_λ. A *relational hash* scheme RH for R_λ consists of PPT algorithms $\mathsf{RH.KeyGen}$, $\mathsf{RH.HASH}_1$, $\mathsf{RH.HASH}_2$, and $\mathsf{RH.Verify}$:

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

Correctness: A relational hash scheme RH is *correct* if $\forall \mathbf{x}, \mathbf{y}, \mathbf{z} \in X_\lambda \times Y_\lambda \times Z_\lambda$,

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

Note that Z_λ is an auxiliary input. When the relation R is over two sets $X \times Y$, we ignore Z and write $\mathsf{RH.Verify}(\mathsf{pk}, \mathbf{h}_\mathbf{x}, \mathbf{h}_\mathbf{y})$.
Instantiation using a relational hash is given in Section 2.6.

2 Formalization

In general, an authentication shceme Π associated with a family of biometric distributions $\mathbb{B}$ is composed of the following algorithms.

- $\mathsf{Setup}(1^\lambda) \to \mathsf{esk}, \mathsf{psk}, \mathsf{csk}$: It outputs the enrollment secret key esk, probe secret key psk, and compare secret key csk.
- $\mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}() \to \mathbf{x}$: Given an oracle $\mathcal{O}_\mathcal{B}$, which samples biometric data from the distribution $\mathcal{B} \in \mathbb{B}$, it encodes biometric samples as $\mathbf{x}$, the input format for enrollment. We write $\mathsf{encodeEnroll}(\mathbf{b}) \to \mathbf{x}$ when $\mathsf{encodeEnroll}$ only has one biometric template vector $\mathbf{b}$ to generate $\mathbf{x}$.

- $\mathsf{Enroll}(\mathsf{esk}, \mathbf{x}) \to \mathbf{c_x}$: It outputs the enrollment message $\mathbf{c_x}$ from $\mathbf{x}$.
- $\mathsf{encodeProbe}^{\mathcal{O}_\mathcal{B}}() \to \mathbf{y}$: Given an oracle $\mathcal{O}_\mathcal{B}$, which samples biometric data from the distribution $\mathcal{B} \in \mathbb{B}$, it encodes biometric samples as $\mathbf{y}$, the input format for probe. We write $\mathsf{encodeProbe}(\mathbf{b}') \to \mathbf{y}$ when $\mathsf{encodeProbe}$ only has one biometric template vector $\mathbf{b}'$ to generate $\mathbf{y}$.
- $\mathsf{Probe}(\mathsf{psk}, \mathbf{y}) \to \mathbf{c_y}$: It outputs the probe message $\mathbf{c_y}$ from $\mathbf{y}$.
- $\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_y}) \to s$: It compares the enrollment message $\mathbf{c_x}$ and probe message $\mathbf{c_y}$ and outputs a score s.
- $\mathsf{Verify}(s) \to r \in \{0, 1\}$: 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 consider a deterministic algorithm $\mathsf{BioCompare}$ for authentication correctness.

- $\mathsf{BioCompare}(\mathbf{x}, \mathbf{y}) \to s$: Given two encoded vectors $\mathbf{x}$ and $\mathbf{y}$, it outputs a score s.

Correctness: An authenticaion scheme Π is *correct* if for any biometric distributions $\mathcal{B}$ and $\mathcal{B}'$, let $\mathsf{esk}, \mathsf{psk}, \mathsf{csk} \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}()$, $\mathbf{y} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}'}}()$, , $\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$, $\mathbf{c_y} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{y})$. Then

$$\Pr\left[\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_y}) = \mathsf{BioCompare}(\mathbf{x}, \mathbf{y})\right] = 1 - \mathsf{negl}.$$

We discuss two usage models that employs the authentication scheme Π.

2.1 Usage Model – Device-of-User

In the model described in Figure 1 (an overview), Figure 2 (on enrollment), and Figure 3 (on authentication), users authenticate themselves to a server through their own devices and biometric scanners that are shared among different users. A key distribution service distributes keys for them. In practice, this model applies to the situation when the users access an online service run by the server.

- User: The user who enrolls its biometric data and authenticates itself to the server. We assume the user's biometric distribution is $\mathcal{B} \in \mathbb{B}$.
- $\mathsf{Scanner}$: A machine to extract the user's biometric data by querying the oracle $\mathcal{O}_\mathcal{B}$.
- Device: A device belonging to the user. In practice, it can be a desktop or a mobile phone. It processes the Enroll and Probe functions for User with keys esk and psk. It queries $\mathcal{O}_\mathcal{B}$ for biometric data through the $\mathsf{Scanner}$.
- KDS: A key distribution service. It runs Setup to generate keys and distribute them to Device and Server.

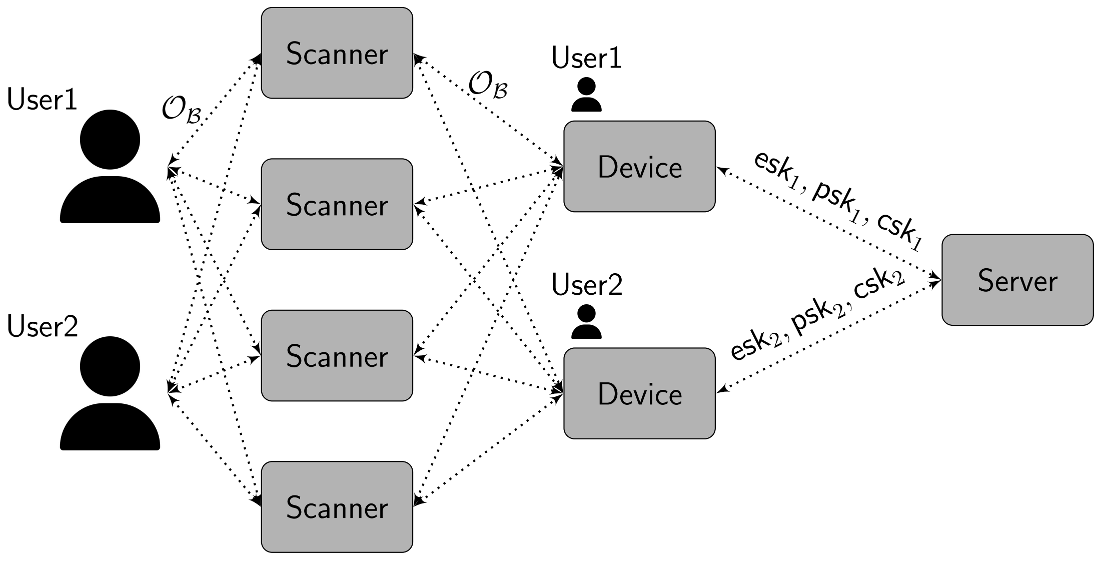
User1
User2
$\mathcal{O}_{\mathcal{B}}$
$\mathcal{O}_{\mathcal{B}}$
Scanner
Scanner
Scanner
Scanner
User1
User2
Device
Device
$\mathsf{esk}_1, \mathsf{psk}_1, \mathsf{csk}_1$
$\mathsf{esk}_2, \mathsf{psk}_2, \mathsf{csk}_2$
Server

Figure 1: An Overview of the Device-of-User Usage Model

- Server: The server responsible for authenticating the user. It stores the comparison key csk and the user's enrollment message $\mathbf{c_x}$. On authentication, it compares the probe message with the registered enrollment message and returns the result.

The Device-of-User model, when instantiated by an fh-IPFE scheme (Section 2.3), is analogous to the use case presented in [EM23]. In their model, a user possesses a personal device, such as a smartphone or laptop, and a secure hardware device that runs an initial setup and stores all the keys, which corresponds to our KDS. On enrollemnt and authentication, the user inputs biometric templates onto the device, which corresponds to our Scanner. Subsequently, the device transmits the template to the secure hardware for the enrollment or probing processes, which are equivalent to our Device. In addition, they incorporate a two-factor authentication mechanism. The secure hardware also executes a digital signature scheme and sign the probe message on authentication.

2.2 Usage Model – Device-of-Domain

In the model described in Figure 4 (an overview), Figure 5 (on enrollment), and Figure 6 (on authentication), users first enroll themselves at an enrollment station and then authenticate themselves to a server through devices that belong to a domain. A key distribution service distributes enrollment keys to the enrollment station, probe keys to the domain, and comparison keys to the server. In practice, a domain can be a department in an organization, and this models applies to the situation when a user wants to access a public service of a department, such as a restricted area or instruments.

- User: The user who enrolls its biometric data at an enrollment station and authenticates itself to the server. We assume the user's biometric distribution is $\mathcal{B} \in \mathbb{B}$.

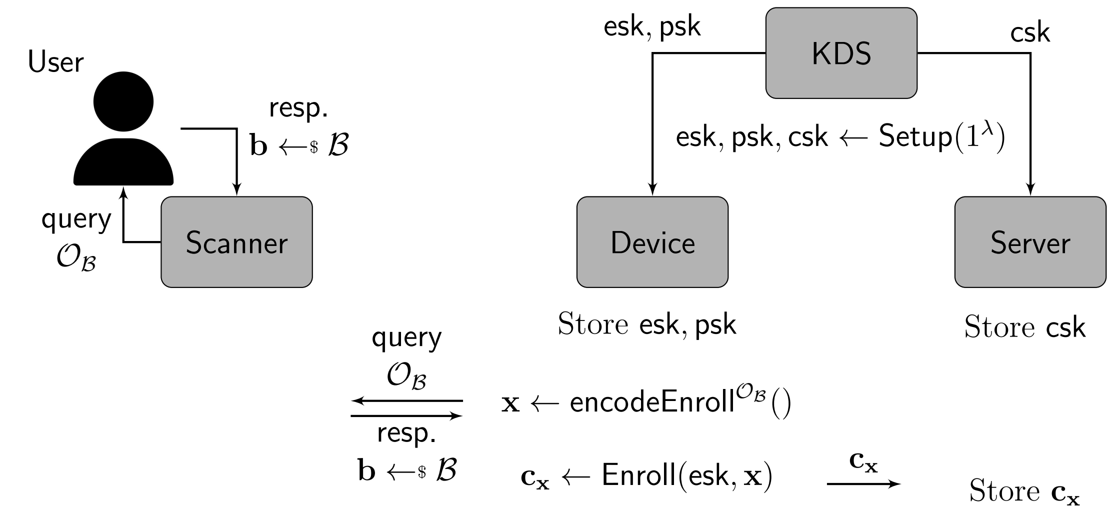
User
resp.
$\mathbf{b} \leftarrow_\$ \mathcal{B}$
query
$\mathcal{O}_\mathcal{B}$
Scanner
esk, psk
KDS
csk
esk, psk, csk ← Setup(1^λ)
Device
Server
Store esk, psk
Store csk
query
$\mathcal{O}_\mathcal{B}$
$\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}()$
resp.
$\mathbf{b} \leftarrow_\$ \mathcal{B}$
$\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$
$\mathbf{c_x}$
Store $\mathbf{c_x}$

Figure 2: Device-of-User Usage Model on Enrollment

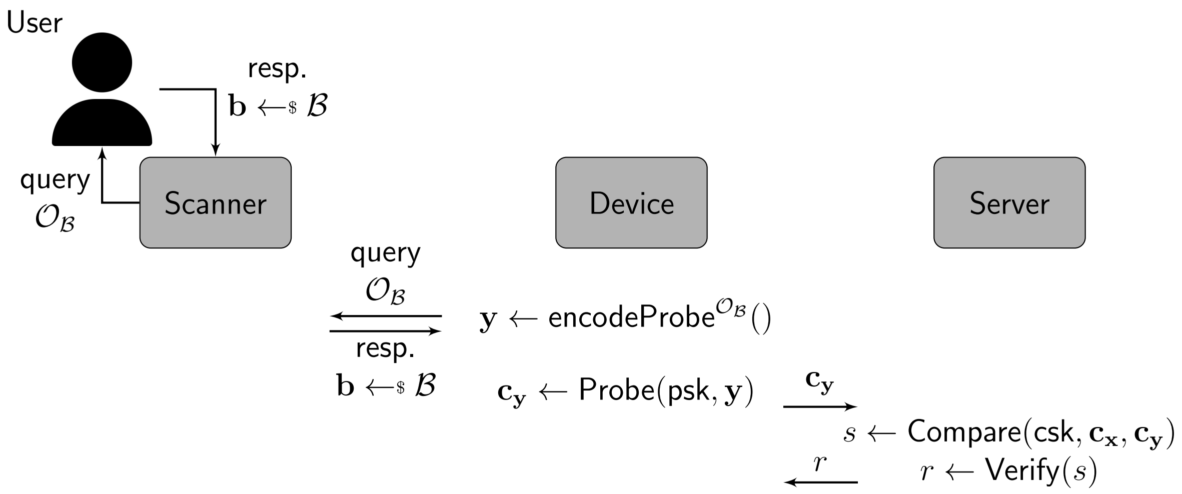
User
resp.
$\mathbf{b} \leftarrow_\$ \mathcal{B}$
query
$\mathcal{O}_\mathcal{B}$
Scanner
Device
Server
query
$\mathcal{O}_\mathcal{B}$
$\mathbf{y} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_\mathcal{B}}()$
resp.
$\mathbf{b} \leftarrow_\$ \mathcal{B}$
$\mathbf{c_y} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{y})$
$\mathbf{c_y}$
$s \leftarrow \mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_y})$
r
$r \leftarrow \mathsf{Verify}(s)$

Figure 3: Device-of-User Usage Model on Authentication

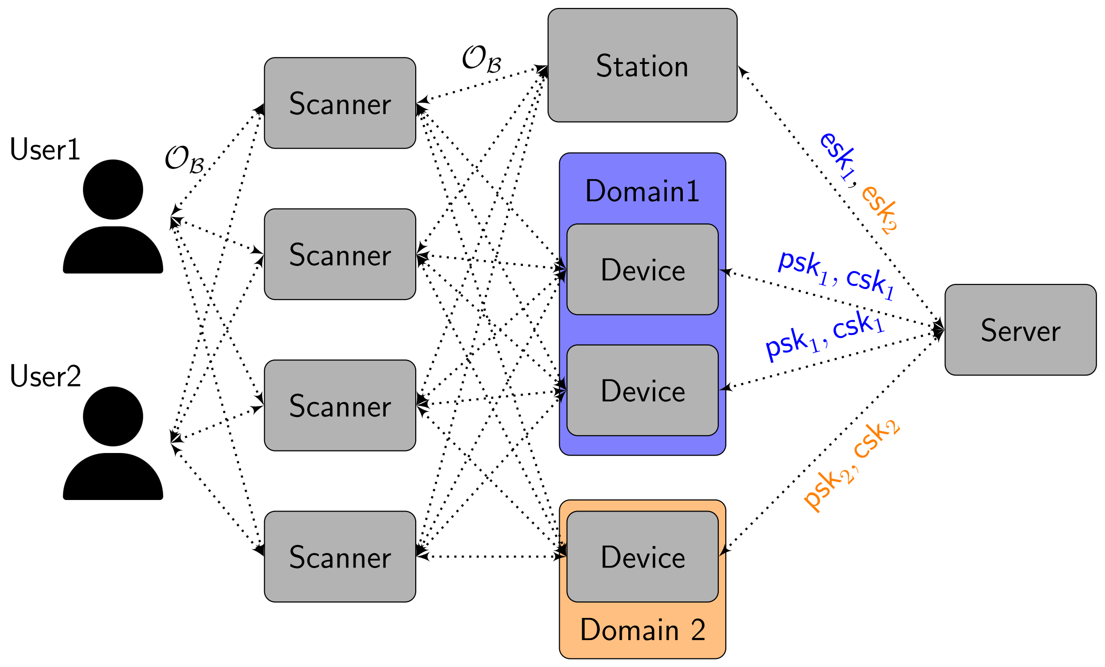

Figure 4: An overview of the Device-of-Domain Usage Model

- Domain: A domain that owns several devices, all of which share one enrollment key esk, one probe key psk and one comparison key csk. Only the probe key is stored at each device of a domain. The enrollment key is stored at the enrollment station, and the comparison key is stored at the server. In practice, a domain can be a department, and users enroll and authenticate themselves before accessing a restricted service of this department.

- Scanner: A machine to extract the user's biometric data by querying the oracle $\mathcal{O}_{\mathcal{B}}$.

- Station: An enrollment station responsible for collecting the user's biometric data to enroll them for a domain on the server.

- Device: A device belonging to a domain. In practice, it can be a device checking identities for a restricted area or an instrument. It owns a probe key psk and processes the Probe function for enrolled users of this domain.

- KDS: A key distribution service. It runs Setup to generate keys and distribute them to Station, Domain, and Server.

- Server: The server responsible for authenticating the user. It stores the comparison key csk for each domain and the user's enrollment message $\mathbf{c_x}$. On authentication, it compares the probe message with the registered enrollment message and returns the result.

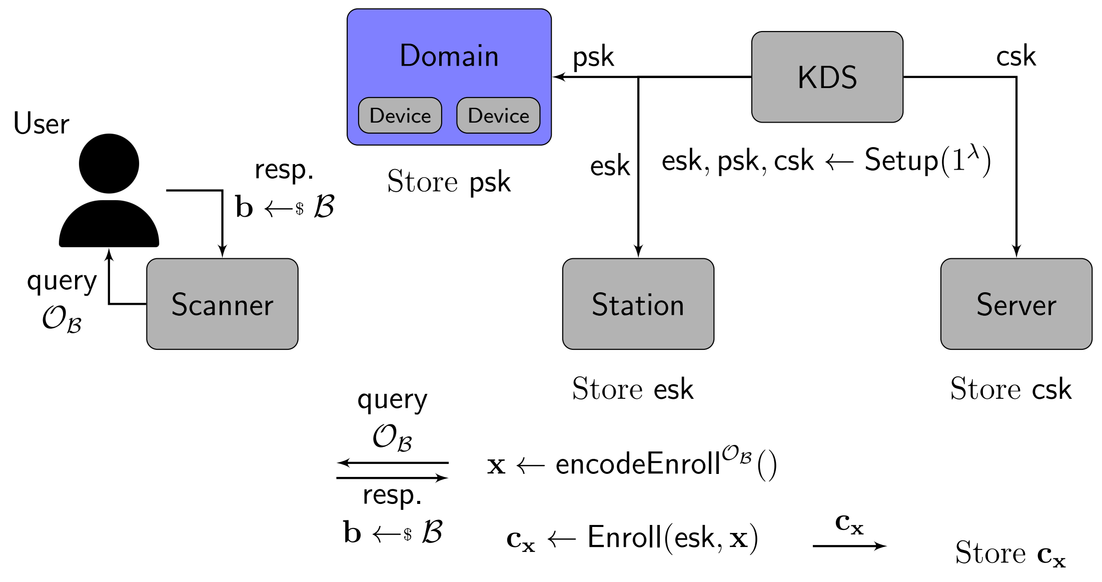
Domain
Device
Device
psk
KDS
csk
User
resp.
$\mathbf{b} \leftarrow_{\$} \mathcal{B}$
Store psk
esk
esk, psk, csk $\leftarrow$ Setup(1^λ)
query
$\mathcal{O}_\mathcal{B}$
Scanner
Station
Server
Store esk
Store csk
query
$\mathcal{O}_\mathcal{B}$
$\mathbf{x} \leftarrow$ encodeEnroll$^{\mathcal{O}_\mathcal{B}}()$
resp.
$\mathbf{b} \leftarrow_{\$} \mathcal{B}$
$\mathbf{c_x} \leftarrow$ Enroll(esk, $\mathbf{x}$)
$\mathbf{c_x}$
Store $\mathbf{c_x}$

Figure 5: Device-of-Domain Usage Model on Enrollment

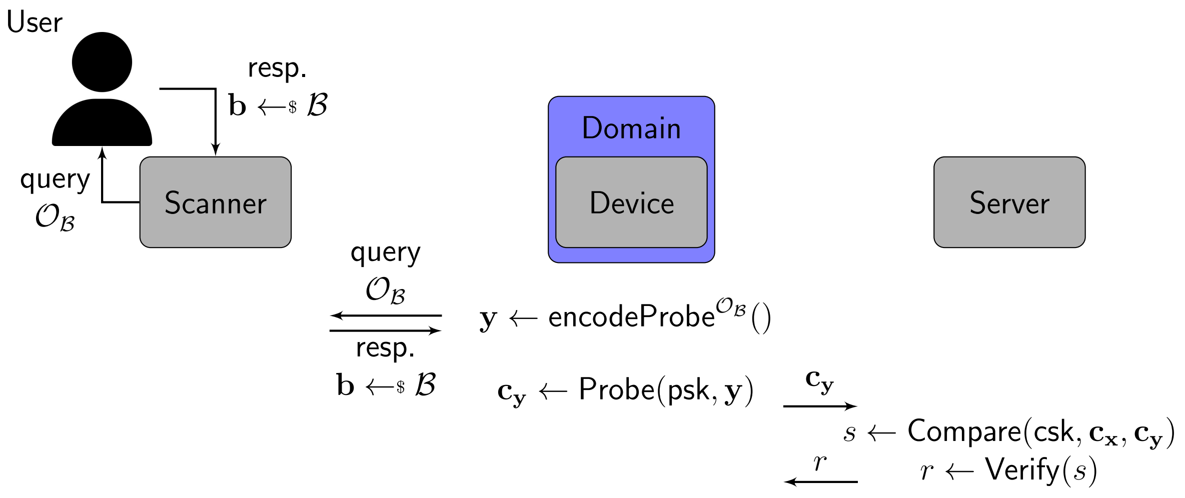
User
resp.
$\mathbf{b} \leftarrow_{\$} \mathcal{B}$
Domain
Device
Server
query
$\mathcal{O}_\mathcal{B}$
Scanner
query
$\mathcal{O}_\mathcal{B}$
$\mathbf{y} \leftarrow$ encodeProbe$^{\mathcal{O}_\mathcal{B}}()$
resp.
$\mathbf{b} \leftarrow_{\$} \mathcal{B}$
$\mathbf{c_y} \leftarrow$ Probe(psk, $\mathbf{y}$)
$\mathbf{c_y}$
$s \leftarrow$ Compare(csk, $\mathbf{c_x}$, $\mathbf{c_y}$)
r
$r \leftarrow$ Verify(s)

Figure 6: Device-of-Domain Usage Model on Authentication

2.3 Instantiation with an fh-IPFE Scheme

Let $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.KeyGen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ be an fh-IPFE scheme we defined in Definition 1. Following [EM23], we can instantiate a biometric authentication scheme using FE with the distance metric the Euclidean distance. Let the biometric distribution $\mathcal{B} \subseteq [m]^k$, and 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.

- $\mathsf{Setup}(1^\lambda)$: It calls $\mathsf{FE.Setup}(1^\lambda) \to \mathsf{msk}, \mathsf{pp}$ and outputs $\mathsf{esk} \leftarrow (\mathsf{msk}, \mathsf{pp})$, $\mathsf{psk} \leftarrow (\mathsf{msk}, \mathsf{pp})$ and $\mathsf{csk} \leftarrow \mathsf{pp}$.
- $\mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}()$: For a template vector $\mathbf{b} = (b_1, b_2, \cdots, b_k)$ sampled from $\mathcal{O}_\mathcal{B}$, the function encodes it as $\mathbf{x} = (x_1, x_2, \cdots, x_{k+2}) = (b_1, b_2, \cdots, b_k, 1, \|\mathbf{b}\|^2)$.
- $\mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$: It calls $\mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{x}) \to f_\mathbf{x}$ and outputs $\mathbf{c_x} \leftarrow f_\mathbf{x}$.
- $\mathsf{encodeProbe}^{\mathcal{O}_\mathcal{B}}()$: For a template vector $\mathbf{b}' = (b'_1, b'_2, \cdots, b'_k)$ sampled from $\mathcal{O}_\mathcal{B}$, the function encodes it as $\mathbf{y} = (y_1, y_2, \cdots, y_{k+2}) = (-2b'_1, -2b'_2, \cdots, -2b'_k, \|\mathbf{b}'\|^2, 1)$.
- $\mathsf{Probe}(\mathsf{psk}, \mathbf{y})$: It calls $\mathsf{FE.Enc}(\mathsf{msk}, \mathsf{pp}, \mathbf{y}) \to \mathbf{c_y}$ and outputs $\mathbf{c_y}$.
- $\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_y})$: It calls $\mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c_x}, \mathbf{c_y}) \to s$ and outputs the value s.
- $\mathsf{Verify}(s)$: If $\sqrt{s} \leq \tau$, a pre-defined threshold for comparing the closeness of two templates, then it outputs $r = 1$; otherwise, it outputs $r = 0$.

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

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

which is the square of the Euclidean distance between two templates $\mathbf{b}$ and $\mathbf{b}'$. 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 csk is public, and the enrollment secret key esk and probe secret key psk are the same. Anyone with access to the enrollment message $\mathbf{c_x}$ and either one of esk, psk, or a probe oracle $\mathsf{Probe}(\mathsf{psk}, \cdot)$ can probe some $\mathbf{y}' \in \mathbb{Z}_q^{k+2}$ and find $\mathbf{x}\mathbf{y}'^T$ to get partial or full information about $\mathbf{x}$. Even if the adversary can only sample random ciphertexts $\mathbf{c_y}$ without knowing $\mathbf{y}$, if the field size q is not large enough, one can find a forged $\mathbf{c}_{\mathbf{y}^*}$ such that $\mathbf{x}\mathbf{y}^{*T} \leq \tau$ to impersonate the user by sampling many times offline.

Therefore, Server must store $\mathbf{c_x}$ securely, to avoid such an attack from an adversary who can access the probe oracle; Device must protect its probe function, to avoid such an attack from a malicious Server.

In the Device-of-Domain model, we assume the probe oracle is public, just as everyone can try accessing a public service. A malicious $\mathsf{Station}$ or Server, who has the enrollment message $\mathbf{c_x}$, can utilize this attack to retrieve information about User.

2.4 Instantiation with a 2i-IPFE Scheme

Let $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.KeyGen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ be a 2i-IPFE scheme we defined in Definition 2. Following the scheme in Section 2.3, we can instantiate a biometric authentication scheme using FE.

- $\mathsf{Setup}(1^\lambda)$: It calls $\mathsf{FE.Setup}(1^\lambda) \to \mathsf{sk}, \mathsf{ek}_1, \mathsf{ek}_2$, $\mathsf{FE.KeyGen}(sk, \mathbf{I}_{k+2}) \to \mathsf{dk}_\mathbf{I}$, where $\mathbf{I}_{k+2}$ is an identity matrix of size $(k+2) \times (k+2)$. It outputs $\mathsf{esk} \leftarrow \mathsf{ek}_1$, $\mathsf{psk} \leftarrow \mathsf{ek}_2$, and $\mathsf{csk} \leftarrow \mathsf{dk}_\mathbf{I}$
- $\mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}(), \mathsf{encodeProbe}^{\mathcal{O}_\mathcal{B}}()$: The same as the scheme in 2.3.
- $\mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$: It calls $\mathsf{FE.Enc}(\mathsf{ek}_1, \mathbf{x}) \to \mathbf{c_x}$ and outputs $\mathbf{c_x}$.
- $\mathsf{Probe}(\mathsf{psk}, \mathbf{y})$: It calls $\mathsf{FE.Enc}(\mathsf{ek}_2, \mathbf{y}) \to \mathbf{c_y}$ and outputs $\mathbf{c_y}$.
- $\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_y})$: It calls $\mathsf{FE.Dec}(\mathsf{dk}_\mathbf{I}, \mathbf{c_x}, \mathbf{c_y}) \to s$ and outputs the value s.
- $\mathsf{Verify}(s)$: If $\sqrt{s} \le \tau$, a pre-defined threshold for comparing the closeness of two templates, then it outputs $r = 1$; otherwise, it outputs $r = 0$.

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

$$s = \mathsf{FE.Dec}(\mathsf{dk}_\mathbf{I}, \mathbf{c_x}, \mathbf{c_y}) = \mathbf{x}\mathbf{I}_{k+2}\mathbf{y}^T = \mathbf{x}\mathbf{y}^T = \|\mathbf{b} - \mathbf{b}'\|^2.$$

just as the scheme in Section 2.3

Unlike the previous scheme, instantiated with a 2i-IPFE scheme in this way, the comparison secret key csk is now secret, and the enrollment secret key esk and probe secret key psk are distinct. Without csk, one cannot compare an enrollment message $\mathbf{c_x}$ and a probe message $\mathbf{c_y}$. We can also transmit $\mathbf{c_x}$ in a public channel and store it in a public storage, under necessary security requirements of the 2i-IPFE scheme, such as indistinguishability of $\mathbf{c_x}$.

In the Device-of-Domain model, the indistinguishability of $\mathbf{c_x}$ is against an adversary who has a probe oracle $\mathsf{Probe}(\mathsf{psk}, \cdot)$. If Server is malicious, then it can use csk to distinguish $\mathbf{c_x}$ enrolled by different samples. Therefore, we must limit the adversary's ability. For example, we can require the adversary to distinguish biometric vectors sampled from distributions in a pre-defined pool, and the adversary can only probe vectors randomly sampled from a distribution in the pool. We can also limit the rate of the probe oracle.

2.5 Instantiation with a 2c-IPFE Scheme

Note that if labels remain constant, a 2c-IPFE scheme is reduced to a 2i-IPFE scheme. Therefore, we can consider utilizing the label to represent each domain in the Device-of-Domain model. Let $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.KeyGen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ be a 2c-IPFE scheme we defined in Definition 3. Following the scheme in Section 2.4, we can instantiate a biometric authentication scheme using FE.

- $\mathsf{Setup}(1^\lambda)$: It calls $\mathsf{FE.Setup}(1^\lambda) \to \mathsf{sk}, \mathsf{ek}_1, \mathsf{ek}_2$, $\mathsf{FE.KeyGen}(sk, \mathbf{I}_{k+2}) \to \mathsf{dk}_\mathbf{I}$, where $\mathbf{I}_{k+2}$ is an identity matrix of size $(k+2) \times (k+2)$. For keys used for Domain ℓ, it outputs $\mathsf{esk} \leftarrow (\ell, \mathsf{ek}_1)$, $\mathsf{psk} \leftarrow (\ell, \mathsf{ek}_2)$, and $\mathsf{csk} \leftarrow \mathsf{dk}_\mathbf{I}$.

  Note that when the previous 2i-IPFE-based scheme in Section 2.4 is applied to a Device-of-Domain model, we assume that Setup is run once for each domain to generate different $\mathsf{esk}, \mathsf{psk}, \mathsf{csk}$. In the scheme in this section, however, Setup is run only once for all the domains, and each domain shares the same csk and the same $\mathsf{esk}, \mathsf{psk}$ except different labels.

- $\mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}(), \mathsf{encodeProbe}^{\mathcal{O}_\mathcal{B}}()$: The same as the scheme in 2.4.

- $\mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$: It calls $\mathsf{FE.Enc}(\ell, \mathsf{ek}_1, \mathbf{x}) \to \mathbf{c_x}$ and outputs $\mathbf{c_x}$.

- $\mathsf{Probe}(\mathsf{psk}, \mathbf{y})$: It calls $\mathsf{FE.Enc}(\ell, \mathsf{ek}_2, \mathbf{y}) \to \mathbf{c_y}$ and outputs $\mathbf{c_y}$.

- $\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_y})$: It calls $\mathsf{FE.Dec}(\mathsf{dk}_\mathbf{I}, \mathbf{c_x}, \mathbf{c_y}) \to s$ and outputs the value s.

- $\mathsf{Verify}(s)$: If $\sqrt{s} \le \tau$, a pre-defined threshold for comparing the closeness of two templates, then it outputs $r = 1$; otherwise, it outputs $r = 0$.

By the correctness of the functional encryption scheme FE, if the labels of $\mathbf{c_x}$ and $\mathbf{c_y}$ are the same (they are of the same domain), we have

$$s = \mathsf{FE.Dec}(\mathsf{dk}_\mathbf{I}, \mathbf{c_x}, \mathbf{c_y}) = \mathbf{x}\mathbf{I}_{k+2}\mathbf{y}^T = \|\mathbf{b} - \mathbf{b}'\|^2.$$

just as the scheme in Section 2.4

When the Device-of-Domain model is instantiated with a 2c-IPFE scheme in this way, the enrollment secret key esk and probe secret key psk are now shared among all the devices, regardless of their domains. Therefore, to let a malicious or broken Domain not threaten other honest ones, one needs to make sure given esk or psk, $\mathbf{c_x}$ still does not leak information about $\mathbf{x}$. This is different from the scheme in Section 2.4, where we only need seurity against an adversary who has a probe oracle $\mathsf{Probe}(\mathsf{psk}, \cdot)$.

If Server and Domain are both malicious, then the adversary can use csk to distinguish $\mathbf{c_x}$ and even recover $\mathbf{x}$. Therefore, we assume at most one party of them can be malicious at the same time. Note that this is the same as the 2i-IPFE-based scheme, where only one of Server and Domain can be malicious.

2.6 Instantiation with a Relational Hash Scheme

Let $\mathsf{RH} = (\mathsf{RH.KeyGen}, \mathsf{RH.Hash}_1, \mathsf{RH.Hash}_2, \mathsf{RH.Verify})$ be a relational hash scheme we defined in Definition 4 for the relation R^τ of Hamming distance proximity parametrized by a constant τ.

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

Note that here we ignore the third parameter Z. Following [MR14], we can instantiate a biometric authentication scheme using RH. Let the biometric distribution $\mathcal{B} \subseteq \{0,1\}^k$.

- $\mathsf{Setup}(1^\lambda)$: It calls $\mathsf{RH.KeyGen}(1^\lambda) \to \mathsf{pk}$ and outputs $\mathsf{esk} \leftarrow \mathsf{pk}$, $\mathsf{psk} \leftarrow \mathsf{pk}$, and $\mathsf{csk} \leftarrow \mathsf{pk}$.
- $\mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}()$: For a template vector $\mathbf{b}$ sampled from $\mathcal{O}_\mathcal{B}$, it direclty outputs $\mathbf{x} \leftarrow \mathbf{b}$.
- $\mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$: It calls $\mathsf{RH.Hash}_1(\mathsf{pk}, \mathbf{x}) \to \mathbf{h_x}$ and outputs $\mathbf{c_x} \leftarrow \mathbf{h_x}$.
- $\mathsf{encodeProbe}^{\mathcal{O}_\mathcal{B}}()$: For a template vector $\mathbf{b}'$ sampled from $\mathcal{O}_\mathcal{B}$, it directy outputs $\mathbf{y} \leftarrow \mathbf{b}'$.
- $\mathsf{Probe}(\mathsf{psk}, \mathbf{y})$: It calls $\mathsf{RH.Hash}_2(\mathsf{pk}, \mathbf{y}) \to \mathbf{h_y}$ and outputs $\mathbf{c_y} \leftarrow \mathbf{h_y}$.
- $\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_y})$: It calls $\mathsf{RH.Verify}(\mathsf{pk}, \mathbf{h_x}, \mathbf{h_y}) \to s$ and outputs the value s.
- $\mathsf{Verify}(s)$: It direclty returns $r \leftarrow s$.

By the correctness of the relational hash scheme RH, we have (except for a negligible probability),

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

3 Security Games

To rigorously analyze the security of an authentication scheme, we simulate biometric distributions of users by assuming the existence of a family $\mathbb{B}$ of distributions. We require that all distributions in $\mathbb{B}$ are efficiently samplable and has an excessively large size for a PPT adversary to enumerate. We then provide interfaces for all algorithms to interact with $\mathbb{B}$.

- $\mathsf{BioSamp}()$: 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 $\mathcal{B} \leftarrow \mathsf{BioSamp}()$ as $\mathcal{B} \leftarrow_\$ \mathbb{B}$.
- $\mathsf{BioDelete}(\mathcal{B})$: Delete $\mathcal{B}$ from $\mathbb{B}$. Consequently, no further access to $\mathsf{BioSamp}$ can derive $\mathcal{B}$. For simplicity, we write $\mathsf{BioDelete}(\mathcal{B})$ as $\mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$.
- $\mathsf{TempSamp}(\mathcal{B})$: Let $\mathcal{B}$ be a biometric distribution in $\mathbb{B}$. This algorithm samples a biometric template from $\mathcal{B}$. For simplicity, we write $\mathbf{b} \leftarrow \mathsf{TempSamp}(\mathcal{B})$ as $\mathbf{b} \leftarrow_\$ \mathcal{B}$.

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 $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ is defined in Algorithm 1.

The auxiliary information option can be nothing or include $\mathbf{c_x}, \mathsf{esk}, \mathsf{psk}, \mathsf{csk}$ or the following oracles:

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

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

- $\mathcal{O}_\mathcal{B}$: It outputs a biometric sample $\mathbf{b} \leftarrow_{\$} \mathcal{B}$. This oracle and psk should not be given at the same time.
- $\mathcal{O}_{\mathsf{Enroll}}(\mathsf{esk}, \cdot)$: On input $\mathbf{x}'$, it outputs the enrollment message $\mathsf{Enroll}(\mathsf{esk}, \mathbf{x}')$.
- $\mathcal{O}_{\mathsf{Probe}}(\mathsf{psk}, \cdot)$: On input $\mathbf{y}'$, it outputs the probe message $\mathsf{Probe}(\mathsf{psk}, \mathbf{y}')$. If this oracle is given, we require the adversary to return a $\tilde{\mathbf{z}}$ that is not equal to any previous answer of $\mathcal{O}_{\mathsf{Probe}}$.
- $\mathcal{O}_{\mathsf{log}}(\mathsf{csk}, \mathbf{c_x}, \cdot)$: On input $\mathbf{z}$, it first computes $\mathbf{c_z} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathsf{encodeProbe}(\mathbf{z}))$ and outputs $\mathsf{Verify}(\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_z}))$.
- $\mathcal{O}'_{\mathsf{Enroll}}(\cdot)$: On input esk', it first samples $\mathbf{x}' \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}()$ and outputs $\mathsf{Enroll}(\mathsf{esk}', \mathbf{x}')$.
- $\mathcal{O}'_{\mathsf{Probe}}(\cdot)$: On input psk', it first samples $\mathbf{y}' \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_\mathcal{B}}()$ and outputs $\mathsf{Probe}(\mathsf{psk}', \mathbf{y}')$. This oracle and psk should not be given at the same time.

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

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

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

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

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

3.2 Choice of option and True/False Positive Rates

In this section, we detail possibilities of the auxiliary information option in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game and rule out trivial attacks.

For a biometric distribution $\mathcal{B} \in \mathbb{B}$ and $\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}()$, define the *true positive rates* TP.

$$\begin{aligned}
\mathsf{TP}(\mathcal{B},\mathbf{x}) &:= \Pr[\mathbf{y} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_\mathcal{B}}() : \mathsf{Verify}(\mathsf{BioCompare}(\mathbf{x},\mathbf{y})) = 1] \\
\mathsf{TP}(\mathcal{B}) &:= \Pr\begin{bmatrix} \mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}() \\ \mathbf{y} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_\mathcal{B}}() \end{bmatrix} : \mathsf{Verify}(\mathsf{BioCompare}(\mathbf{x},\mathbf{y})) = 1 \\
&= \mathbb{E}_{\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}()}[\mathsf{TP}(\mathcal{B},\mathbf{x})] \\
\mathsf{TP} &:= \Pr\begin{bmatrix} \mathcal{B} \leftarrow_\$ \mathbb{B} \\ \mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}() : \mathsf{Verify}(\mathsf{BioCompare}(\mathbf{x},\mathbf{y})) = 1 \\ \mathbf{y} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_\mathcal{B}}() \end{bmatrix} \\
&= \mathbb{E}_{\mathcal{B} \leftarrow_\$ \mathbb{B}}[\mathsf{TP}(\mathcal{B})]
\end{aligned}$$

For a biometric distribution $\mathcal{B} \in \mathbb{B}, \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$ and $\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}()$, we also define the *false positive rates* FP.

$$\begin{aligned}
\mathsf{FP}(\mathbf{x}) &:= \Pr\begin{bmatrix} \mathcal{B}' \leftarrow_\$ \mathbb{B} \\ \mathbf{y}' \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}'}}() \end{bmatrix} : \mathsf{Verify}(\mathsf{BioCompare}(\mathbf{x},\mathbf{y})) = 1 \\
\mathsf{FP}(\mathcal{B}) &:= \Pr\begin{bmatrix} \mathcal{B}' \leftarrow_\$ \mathbb{B} \\ \mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}() \ : \mathsf{Verify}(\mathsf{BioCompare}(\mathbf{x},\mathbf{y})) = 1 \\ \mathbf{y} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}'}}() \end{bmatrix} \\
&= \mathbb{E}_{\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}()}[\mathsf{FP}(\mathbf{x})] \\
\mathsf{FP} &:= \Pr\begin{bmatrix} \mathcal{B}, \mathcal{B}' \leftarrow_\$ \mathbb{B} \\ \mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}() \ : \mathsf{Verify}(\mathsf{BioCompare}(\mathbf{x},\mathbf{y})) = 1 \\ \mathbf{y} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}'}}() \end{bmatrix} \\
&= \mathbb{E}_{\mathcal{B} \leftarrow_\$ \mathbb{B}}[\mathsf{FP}(\mathcal{B})]
\end{aligned}$$

Ideally, we hope TP to be close to 1 and FP to be negligible for any $\mathcal{B}$. However, due to the nature of biometrics, FP can be non-negligible. In the design of UF game, we need 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 $\mathcal{O}_\mathsf{Probe}$, 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 from returning what $\mathcal{O}_\mathsf{Probe}$ returns.

If option has only psk or $\mathcal{O}_\mathsf{Probe}$, the UF adversary $\mathcal{A}$ in Algorithm 2 can still enjoy a winning rate FP, if we do not place any 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 $\mathcal{O}_\mathsf{Probe}$ is given.

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

```
1: $\mathcal{B}' \leftarrow_{\$} \mathbb{B}$
2: $\mathbf{y} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}'}}()$
3: $\mathbf{c_y} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{y})$        $\triangleright$ or $\mathbf{c_y} \leftarrow \mathcal{O}_{\mathsf{Probe}}(\mathbf{y})$
4: return $\mathbf{c_y}$
```

3.3 Indistinguishable against Malicious Server (IND-MSV)

In the game of indistinguishability against a malicious server, we model the ability of an authentication server who tries to identify the user. The adversary $\mathcal{A}$ is given oracles to two biometric distributions $\mathcal{B}^{(0)}$ and $\mathcal{B}^{(1)}$, the comparison key 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 $\mathsf{IND\text{-}MSV}_{\Pi,\mathbb{B}}(\mathcal{A})$

```
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: $\mathsf{esk}, \mathsf{psk}, \mathsf{csk} \leftarrow \mathsf{Setup}(1^{\lambda})$
5: $\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}^{(b)}}}()$
6: $\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$
7: for $i = 1$ to $t$ do
8:     $\mathbf{y}^{(i)} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}^{(b)}}}()$
9:     $\mathbf{c_y}^{(i)} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{y}^{(i)})$
10: end for
11: $\tilde{b} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{B}^{(0)}}, \mathcal{O}_{\mathcal{B}^{(1)}}}(\mathsf{csk}, \mathbf{c_x}, \{\mathbf{c_y}^{(i)}\}_{i=1}^{t})$
12: return $1_{\tilde{b}=b}$
```

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

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

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

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

4 Security Analysis: fh-IPFE-based Instantiation

Let Π be an authentication scheme instantiated by an fh-IPFE scheme FE as in Section 2.3. We discuss the UF and IND-MSV security of Π in this section. For this, we first define two security notions of FE.

Given an fh-IPFE scheme FE, we define the $\mathsf{fh\text{-}IND}$ game [EM23] in Algorithm 4.

Algorithm 4 $\mathsf{fh\text{-}IND}_{\mathsf{FE}}(\mathcal{A})$

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

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

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

Definition 5 (Admissible Adversary)**.** Let $\mathcal{A}$ be an adversary in an $\mathsf{fh\text{-}IND}$ game, and let $(\mathbf{x}_1^{(0)}, \mathbf{x}_1^{(1)}), \cdots, (\mathbf{x}_{Q_K}^{(0)}, \mathbf{x}_{Q_K}^{(1)})$ be its queries to $\mathcal{O}_{\mathsf{KeyGen}}$ and $(\mathbf{y}_1^{(0)}, \mathbf{y}_1^{(1)}), \cdots, (\mathbf{y}_{Q_E}^{(0)}, \mathbf{y}_{Q_E}^{(1)})$ be its queries to $\mathcal{O}_{\mathsf{Enc}}$. 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 $\mathsf{fh\text{-}IND}$ game in Algorithm 4 is

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

We note that the constructions in [DDM15; TAO16; Kim+16] are fh-IND secure.

We also define the RUF game in Algorithm 5 for a real number γ.

- $\mathcal{O}'_{\mathsf{KeyGen}}(\cdot)$: On input $\mathbf{x}'$, it outputs $\mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{x}')$.
- $\mathcal{O}'_{\mathsf{Enc}}(\cdot)$: On input $\mathbf{y}'$, it outputs $\mathsf{FE.Enc}(\mathsf{msk}, \mathsf{pp}, \mathbf{y}')$. 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 FE is called RUF secure for a real number γ if for any adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in the RUF game in Algorithm 5 is

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

Algorithm 5 $\mathsf{RUF}^{\gamma}_{\mathsf{FE}}(\mathcal{A})$

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

We note that by adding a sEUF-CMA signature scheme $\mathsf{Sig} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$, an fh-IPFE scheme can be upgraded to an RUF secure scheme. In a bit more detail, given any fh-IPFE FE, we construct the new scheme in the following way.

- $\mathsf{FE.Setup}$ also runs $\mathsf{Sig.KeyGen}(1^\lambda)$ and generates the signature secret key $\mathsf{sk}_{\mathsf{Sig}}$ and the verification public key $\mathsf{pk}_{\mathsf{Sig}}$. Let $\mathsf{sk}_{\mathsf{Sig}}$ be part of msk and $\mathsf{pk}_{\mathsf{Sig}}$ be part of pp.

- $\mathsf{FE.Enc}$ signs the encryption by $\mathsf{sk}_{\mathsf{Sig}}$.

- $\mathsf{FE.Dec}$ outputs the decryption if the verification succeeds. Otherwise, it outputs $\perp$.

If the adversary manages to find a $\tilde{\mathbf{z}}$ that is not equal to any output of $\mathcal{O}'_{\mathsf{Enc}}$ and $\mathsf{FE.Dec}$ on input $\tilde{\mathbf{z}}$ does not return $\perp$, the adversary is able to forge a valid signagure.

4.1 UF Security

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

Theorem 1. *Let $\mathsf{option} = \{\mathbf{c}_{\mathbf{x}}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Enroll}}\}$. For any distribution family $\mathbb{B}$, if FE is fh-IND secure and RUF secure for a $\gamma \geq \tau^2$, then Π is option-unforgeable.*

Proof. Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\mathsf{option}}$ game, consider the reduction adversary $\mathcal{R}$ in Algorithm 6 which plays the fh-IND game. $\mathcal{R}$ runs $\mathcal{A}$ and simulates $\mathcal{O}_{\mathsf{Enroll}}(\mathsf{esk}, \mathbf{x}')$ by $\mathcal{O}_{\mathsf{KeyGen}}(\mathbf{x}', \mathbf{x}')$ given in the fh-IND game. Note that since $\mathcal{R}$ never calls $\mathcal{O}_{\mathsf{Enc}}$, it is an admissible adversary.

If the challenge bit $b = 0$, then $\mathcal{R}$ perfectly simulates a $\mathsf{UF}_{\mathsf{option}}$ game for $\mathcal{A}$. Therefore, the probability that $\mathsf{Verify}(s) = 1$ in Line 7 is $\Pr[\mathsf{UF}_{\mathsf{option}}(\mathcal{A}) \rightarrow 1]$.

For the case when the challenge bit $b = 1$, consider an adversary $\mathcal{A}'$ in Algorithm 7 in the RUF game. $\mathcal{A}'$ runs Line 1 and 5 of $\mathcal{R}$ and simulates $\mathcal{O}_{\mathsf{Enroll}}(\mathsf{esk}, \mathbf{x}')$ by $\mathcal{O}'_{\mathsf{KeyGen}}(\mathbf{x}')$ given in the RUF game.

Algorithm 6 $\mathcal{R}^{\mathcal{O}_{\mathsf{KeyGen}},\mathcal{O}_{\mathsf{Enc}}}(\mathsf{pp})$

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

Algorithm 7 $\mathcal{A}'^{\mathcal{O}'_{\mathsf{KeyGen}},\mathcal{O}'_{\mathsf{Enc}}}(\mathsf{pp}, \mathbf{c})$

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

Now, if the challenge bit $b = 1$, then $\mathcal{R}$ perfectly simulates $\mathcal{A}'$ in the RUF game. The probability that $\mathsf{Verify}(s) = 1$, which is equivalent to $s \le \tau^2$, in Line 7 is $\Pr[\mathsf{RUF}_{\mathsf{FE}}^{\tau^2}(\mathcal{A}') \to 1]$

In conclusion, since $\gamma \ge \tau^2$,

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

Since both $\mathbf{Adv}_{\mathsf{FE},\mathcal{R}}^{\mathsf{fh\text{-}IND}} = \left|\Pr[\mathsf{fh\text{-}IND}(\mathcal{R}) \to 1] - \frac{1}{2}\right|$ and $\mathbf{Adv}_{\mathsf{FE},\mathcal{A}'}^{\mathsf{RUF},\gamma} = \Pr[\mathsf{RUF}_{\mathsf{FE}}^{\gamma}(\mathcal{A}') \to 1]$ are negligilbe,

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

$\square$

For option that includes $\mathcal{O}_{\mathsf{Probe}}$, 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 $\mathbf{y}$ such that $\mathbf{r}\mathbf{y}^T = d$.

Theorem 2. *Let option* $= \{\mathbf{c}_{\mathbf{x}}, \mathit{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathit{Probe}}\}$*. For any distribution family* $\mathbb{B}$*, if FE is fh-IND secure and RUF secure for a* $\gamma \ge \tau^2$*, then* Π *is option-unforgeable.*

Proof. Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\mathsf{option}}$ game, consider the reduction adversary $\mathcal{R}$ in Algorithm 8 which plays the fh-IND game. $\mathcal{R}$ runs $\mathcal{A}$ and simulates $\mathcal{O}_{\mathsf{Probe}}$ in the following way.

- $\mathcal{O}_{\mathsf{Probe}}(\mathsf{psk}, \mathbf{y}')$: It first computes $d \leftarrow \mathbf{x}\mathbf{y}'^T$ and finds a vector $\mathbf{y}''$ such that $\mathbf{r}\mathbf{y}''^T = d$. Next, it calls $\mathcal{O}_{\mathsf{Enc}}(\mathbf{y}', \mathbf{y}'')$, 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 $\mathcal{O}_{\mathsf{KeyGen}}$, and for any query $(\mathbf{y}', \mathbf{y}'')$ to $\mathcal{O}_{\mathsf{Enc}}$, it satisfies $\mathbf{x}\mathbf{y}'^T = \mathbf{r}\mathbf{y}''^T$. Hence, $\mathcal{R}$ is an admissible adversary.

Algorithm 8 $\mathcal{R}^{\mathcal{O}_{\mathsf{KeyGen}}, \mathcal{O}_{\mathsf{Enc}}}(\mathsf{pp})$

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

If the challenge bit $b = 0$, then $\mathcal{R}$ perfectly simulates a $\mathsf{UF}_{\mathsf{option}}$ game for $\mathcal{A}$. Therefore, the probability that $\mathsf{Verify}(s) = 1$ in Line 10 is $\Pr[\mathsf{UF}_{\mathsf{option}}(\mathcal{A}) \rightarrow 1]$.

For the case when the challenge bit $b = 1$, consider an adversary $\mathcal{A}'$ in Algorithm 9 in the RUF game. $\mathcal{A}'$ runs $\mathcal{A}$ and simulates $\mathcal{O}_{\mathsf{Probe}}$ in the following way.

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

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

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

Let $\mathcal{A}'$ play a tweaked $\mathsf{RUF}_{\mathsf{FE}}^{\tau^2}$ 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 $\mathcal{O}'_{\mathsf{Enc}}$ called by $\mathcal{O}_{\mathsf{Probe}}$ 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

Algorithm 9 $\mathcal{A}'^{\mathcal{O}'_{\mathsf{KeyGen}},\mathcal{O}'_{\mathsf{Enc}}}(\mathsf{pp}, \mathbf{c})$

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

when the challenge bit $b = 1$ and V, and the other one is a relation between $\mathcal{A}'$ playing a regular $\mathsf{RUF}_{\mathsf{FE}}^{\tau^2}$ game and the tweaked one.

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

$$\Pr[\mathsf{RUF}_{\mathsf{FE}}^{\tau^2}(\mathcal{A}') \to 1] = \Pr\left[V = 1 \mid \bigwedge_{i=1}^{k+2} \tilde{\mathbf{z}} \neq \mathbf{c}^{(i)}\right] \tag{2}$$

For Equation 1, consider that

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

For Equation 2, consider that

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

Note that each $\mathbf{c}^{(i)} = \mathsf{FE.Enc}(\mathsf{msk}, \mathsf{pp}, \mathbf{e}^{(i)})$ 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

$$\Pr[\mathsf{RUF}_{\mathsf{FE}}^{\tau^2}(\mathcal{A}') \to 1] \geq \Pr[V = 1] - \frac{k+2}{q^{k+2}-1}.$$

Combining both results from Equation 1 and 2, we derive

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

Finally, similar to the proof of Theorem 1, we derive

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

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

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

□

Unfortunately, for the instantiation in Section 2.3, we cannot achieve UF security when the adversary has psk, even if the false positive rate is negligible. The adversary can simply ask $\mathbf{c} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{0})$ and return $\mathbf{c}$. While in some fh-IPFE constructions [DDM15; Kim+16], FE.Enc disallows a zero input vector, the adversary can still ask $\mathbf{c} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{v})$, where $\mathbf{v} = (0, \cdots, 0, 1, 0)$ has only a single 1 in the $k+1$-th coefficient, and win the game with probability 1. 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 $\mathsf{FE.Enc}(\mathsf{msk}, \mathsf{pp}, \mathbf{v})$ for any vector $\mathbf{v}$. We state this result formally in the following theorem.

Theorem 3. *Let option include esk or psk. For any distribution family $\mathbb{B}$ and functional encryption FE, Π is not option-unforgeable.*

4.2 IND-MSV Security

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

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

$$\mathcal{D}_{\mathcal{B}}(t) = \left(\|\mathbf{b} - \mathbf{b}^{(1)}\|, \|\mathbf{b} - \mathbf{b}^{(2)}\|, \cdots, \|\mathbf{b} - \mathbf{b}^{(t)}\|\right)$$

where $\mathbf{b}, \mathbf{b}^{(1)} \cdots, \mathbf{b}^{(t)} \leftarrow_{\$} \mathcal{B}$.

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 when Π is instantiated by an fh-IPFE scheme as in Section 2.3, computational indistinguishability between $\mathcal{D}_{\mathcal{B}^{(0)}}(t)$ and $\mathcal{D}_{\mathcal{B}^{(1)}}(t)$ is a necessary condition to achieve IND-MSV security because

$$\left(\sqrt{\mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c_x}, \mathbf{c_y}^{(1)})}, \cdots, \sqrt{\mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c_x}, \mathbf{c_y}^{(t)})}\right) = \mathcal{D}_{\mathcal{B}^{(b)}}(t)$$

where b is the challenge bit.

Theorem 4. *For any distribution family $\mathbb{B}$ satisfying Assumption 1 and having a true positive rate $\mathsf{TP} > \frac{1}{\mathsf{poly}}$, if FE is fh-IND secure, then Π is IND-MSV secure.*

Proof. Given an adversary $\mathcal{A}$ in the $\mathsf{IND\text{-}MSV}$ game, consider the reduction adversary $\mathcal{R}$ in Algorithm 10 which plays the $\mathsf{fh\text{-}IND}$ game by running $\mathcal{A}$.

Algorithm 10 $\mathcal{R}^{\mathcal{O}_{\mathsf{KeyGen}}, \mathcal{O}_{\mathsf{Enc}}}(\mathsf{pp})$

```
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:  $\mathbf{x}^{(0)} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}^{(0)}}}()$
4:  $\mathbf{x}^{(1)} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}^{(1)}}}()$
5:  $\mathbf{c_x} \leftarrow \mathcal{O}_{\mathsf{KeyGen}}(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$
6:  for $i = 1$ to $t$ do
7:      $\mathbf{y}^{(0)} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}^{(0)}}}()$
8:      repeat
9:          $\mathbf{y}^{(1)} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}^{(1)}}}()$
10:     until $\mathbf{x}^{(0)}\mathbf{y}^{(0)^T} = \mathbf{x}^{(1)}\mathbf{y}^{(1)^T}$
11:     $\mathbf{c_y}^{(i)} \leftarrow \mathcal{O}_{\mathsf{Enc}}(\mathbf{y}^{(0)}, \mathbf{y}^{(1)})$
12: end for
13: $\tilde{b} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{B}^{(0)}}, \mathcal{O}_{\mathcal{B}^{(1)}}}(\mathsf{pp}, \mathbf{c_x}, \{\mathbf{c_y}^{(i)}\}_{i=1}^{t})$
14: return $\tilde{b}$
```

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

The probability that Line 10 is satisfied is

$$\begin{aligned}
\Pr[\mathcal{D}_{\mathcal{B}^{(0)}}(1) = \mathcal{D}_{\mathcal{B}^{(1)}}(1)] &\geq \sum_{i=0}^{\tau} \Pr[\mathcal{D}_{\mathcal{B}^{(0)}}(1) = i]^2 \qquad \text{(Assumption 1)} \\
&\geq \frac{1}{\tau+1} \cdot \left(\sum_{i=0}^{\tau} \Pr[\mathcal{D}_{\mathcal{B}^{(0)}} = i] \right)^2 \\
&= \frac{1}{\tau+1} \cdot \Pr[\mathbf{b}, \mathbf{b}' \leftarrow_{\$} \mathcal{B}^{(0)} : \|\mathbf{b} - \mathbf{b}'\| \leq \tau] \\
&= \frac{\mathsf{TP}(\mathcal{B}^{(0)})}{\tau+1} = \frac{\mathsf{TP}}{\tau+1} \qquad \text{(Assumption 1)}
\end{aligned}$$

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

$$1 - (1 - \frac{\mathsf{TP}}{\tau+1})^T \geq 1 - e^{-T \cdot \frac{\mathsf{TP}}{\tau+1}}$$

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

Now, we show that $\mathcal{R}$ perfectly simulate an $\mathsf{IND\text{-}MSV}$ game for $\mathcal{A}$. If the challenge bit b of the $\mathsf{fh\text{-}IND}$ 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 $\mathsf{IND\text{-}MSV}$ 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 $\mathbf{X}^{(0)}$ and $\mathbf{X}^{(1)}$ be the distribution of $\{\mathbf{x}^{(0)} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}^{(0)}}}\}$ and $\{\mathbf{x}^{(1)} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}^{(1)}}}\}$, respectively. Let $\{\mathbf{Y}_i^{(0)}\}_{i=1}^t$ and $\{\mathbf{Y}_i^{(1)}\}_{i=1}^t$ be t identical and independent distributions of $\{\mathbf{y}^{(0)} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}^{(0)}}}\}$ and $\{\mathbf{y}^{(1)} \leftarrow \mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}^{(1)}}}\}$, respectively. Let $\mathbf{Y}_i'$ be the distribution of $\mathbf{y}^{(1)}$ derived after the loop in Line 10 in the i-th iteration. For any $\{d_i\}_{i=1}^t, d_i > 0$,

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

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

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

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

In conclusion,

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

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

$\square$

5 Security Analysis: Relational Hash-based Instantiation

Let Π be an authentication scheme instantiated by a relational hash scheme RH as in Section 2.6. We discuss the UF and IND-MSV security of Π in this section. Note that in this instantiation, $\mathsf{esk}, \mathsf{psk}, \mathsf{csk}$ are all public hash keys pk of FE and assumed to be given to all adversaries.

Given a relational scheme RH for a relation $R \subseteq X \times Y$, we first define the unforgeability [MR14] of RH.

Definition 9 (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.

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

5.1 UF Security

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

Theorem 5. *Let $\mathsf{option} = \{\mathbf{c_x}, \mathsf{esk}, \mathsf{psk}, \mathsf{csk}\}$. If RH is unforgeable for the distribution*

$$\mathcal{X} = \{\mathcal{B} \leftarrow_{\$} \mathbb{B} : \mathbf{b} \leftarrow_{\$} \mathbb{B} \mid \mathbb{B}\}$$

, then Π is option-unforgeable.

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 in the instantiation of Section 2.6 is

$$\left\{\begin{array}{l} \mathcal{B} \leftarrow_\$ \mathbb{B} \\ \mathsf{pk} \leftarrow \mathsf{RH.KeyGen}(1^\lambda) : \mathbf{c_x} \leftarrow \mathsf{RH.Hash}_1(\mathsf{pk}, \mathbf{x}) \\ \mathbf{x} = \mathbf{b} \leftarrow_\$ \mathcal{B} \end{array}\right\}$$

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

□

Remark As we mentioned in Section 3.2, an adversary with psk can enjoy a winning rate of the false positive rate FP of $\mathbb{B}$. Theorem 5 thus implies that if FP is not negligible, there does not exist an RH that is unforgeable for the distribution $\{\mathcal{B} \leftarrow_\$ \mathbb{B} : \mathbf{b} \leftarrow_\$ \mathbb{B} \mid \mathbb{B}\}$.

Note that since $\mathsf{esk}, \mathsf{psk}$, and csk are all public in this instantiation, it is meaningless to discuss $\mathcal{O}_{\mathsf{Enroll}}, \mathcal{O}_{\mathsf{Probe}}$, or $\mathcal{O}_{\mathsf{log}}$. In addition, for option that includes $\mathcal{O}_{\mathcal{B}}$ or $\mathcal{O}'_{\mathsf{Probe}}$, as discussed in Section 3.2, we cannot achieve option-UF security since psk is public in this instantiation.

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

5.2 IND-MSV Security

Theorem 6. *For any distribution family $\mathbb{B}$ that $\mathsf{TP} - \mathsf{FP} > \frac{1}{\mathsf{poly}}$, and for any relational hash scheme RH, Π is not IND-MSV secure for any $t \geq 0$.*

Proof. Consider the adversary $\mathcal{A}$ in Algorithm 11. 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 $1 - \mathsf{FP}$. Now,

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

□

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

Theorem 7. *Given any distribution family $\mathbb{B}$ that $\mathsf{TP} - \mathsf{FP} > \frac{1}{\mathsf{poly}}$. If psk is public, Π is not IND-MSV secure for any $t \geq 0$. If esk is public, Π is not IND-MSV secure for any $t \geq 1$.*

Algorithm 11 $\mathcal{A}^{\mathcal{O}_{\mathcal{B}^{(0)}},\mathcal{O}_{\mathcal{B}^{(1)}}}(\mathsf{csk} = \mathsf{pk}, \mathbf{c_x}, \{\mathbf{c_y}^{(i)}\}_{i=1}^{t})$

```
1: $\mathbf{y}^{(0)} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}^{(0)}}}()$
2: $\mathbf{h_y}^{(0)} \leftarrow \mathsf{RH.Hash}_2(\mathsf{pk}, \mathbf{y}^{(0)})$
3: if $\mathsf{RH.Verify}(\mathsf{pk}, \mathbf{c_x}, \mathbf{h_y}^{(0)}) = 1$ then
4:     return 0
5: else
6:     return 1
7: end if
```

6 Rough Ideas

Define the $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O},\gamma}$ game in Algorithm 12 for a real number γ.

Algorithm 12 $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O},\gamma}(\mathcal{A})$

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

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

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

Definition 10 (RUF Security)**.** An fh-IPFE scheme FE is called $\mathcal{O}$-RUF secure for a real number γ if for any adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in the $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O},\gamma}$ game in Algorithm 5 is

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

Theorem 8 (Theorem 1)**.** *Let $\mathsf{option} = \{\mathbf{c_x}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Enroll}}\}$. For any distribution family $\mathbb{B}$, if FE is fh-IND secure and $\mathcal{O}'_{\mathsf{KeyGen}}$-RUF secure for a $\gamma \geq \tau^2$, then Π is option-unforgeable.*

Theorem 9 (Theorem 2)**.** *Let $\mathsf{option} = \{\mathbf{c_x}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Probe}}\}$. For any distribution family $\mathbb{B}$, if FE is fh-IND secure and $\mathcal{O}'_{\mathsf{Enc}}(\cdot)$-RUF secure for a $\gamma \geq \tau^2$, then Π is option-unforgeable.*

Assumption 2. Let $\mathbf{x} \in \mathbb{F}^k, \mathbf{c} \leftarrow \mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{x})$. Assume that $\mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c}, \mathbf{z})$ only returns when $\mathbf{z}$ corresponds to a *nonzero* vector $\mathbf{v} \in \mathbb{F}^k$. That is, assume that for any $\mathbf{z}$, there can only be two possibilities.

- There exists a vector $\mathbf{v} \in \mathbb{F}^k \backslash \{\mathbf{0}\}$ such that for any $\mathbf{x} \in \mathbb{F}^k, \mathbf{c} \leftarrow \mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{x})$, and $\mathbf{c}_\mathbf{v} \leftarrow \mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{v})$,
$$\mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c}, \mathbf{z}) = \mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c}, \mathbf{c}_\mathbf{v}).$$
- For any $\mathbf{x} \in \mathbb{F}^k$ and $\mathbf{c} \leftarrow \mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{x})$, $\mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c}, \mathbf{z}) \to \perp$.

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

Theorem 10. *Given Assumption 2. If FE is fh-IND secure, then FE is $\mathcal{O}'_{\mathsf{KeyGen}}$-RUF secure for any $\gamma \leq \|\mathbb{F}\|$.*

Proof. Given an adversary $\mathcal{A}$ in the $\mathsf{RUF}^{\mathcal{O}'_{\mathsf{KeyGen}},\gamma}$ game for any $\gamma < \|\mathbb{F}\|$. Let t be an integer, consider the reduction adversary $\mathcal{R}$. $\mathcal{R}$ simulates $\mathcal{O}'_{\mathsf{KeyGen}}(\mathbf{x}')$ by $\mathcal{O}_{\mathsf{KeyGen}}(\mathbf{x}', \mathbf{x}')$. If there exists an $s_i \neq \perp$ in Line 7, by Assumption 2, let $\tilde{\mathbf{z}}$ correspond to a vector $\tilde{\mathbf{v}}$.

Algorithm 13 $\mathcal{R}^{\mathcal{O}_{\mathsf{KeyGen}},\mathcal{O}_{\mathsf{Enc}}}(\mathsf{pp})$

```
1: $\mathbf{r}^{(0)}, \mathbf{r}^{(1)} \leftarrow_\$ \mathbb{F}^k$
2: $\mathbf{c} \leftarrow \mathcal{O}_{\mathsf{KeyGen}}(\mathbf{r}^{(0)}, \mathbf{r}^{(1)})$
3: $\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}'_{\mathsf{KeyGen}}}(\mathsf{pp}, \mathbf{c})$
4: for $i = 1$ to $t$ do
5:     $\mathbf{r}_i \leftarrow_\$ \mathbb{F}^k$
6:     $\mathbf{c}_i \leftarrow \mathcal{O}_{\mathsf{KeyGen}}(\mathbf{r}^{(0)}, \mathbf{r}_i)$
7:     $s_i \leftarrow \mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c}_i, \tilde{\mathbf{z}})$
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

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

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

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

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,

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

Take t be any integer larger than $\frac{\lambda}{1-\frac{\gamma}{\|\mathbb{F}\|}}$. Since $\mathbf{Adv}^{\mathsf{fh\text{-}IND}}_{\mathsf{FE},\mathcal{R}} = \left|\Pr[\mathsf{fh\text{-}IND}(\mathcal{R}) \to 1] - \frac{1}{2}\right|$ and $e^{-t\cdot(1-\frac{\gamma}{\|\mathbb{F}\|})} \leq e^{-\lambda}$ are negligible,

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

□

7 Agenda

1. If an fh-IPFE scheme FE is fh-IND secure, by our current definition, sampling a $\mathbf{c_v}$ that corresponds to $\mathsf{FE.Enc}(\mathsf{msk}, \mathsf{pp}, \mathbf{v})$ of a random $\mathbf{v}$ and $\mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c_x}, \mathbf{c_v})$ always returns is *impossible*. However, it is *possible* [DDM15; TAO16; Kim+16] if we allow decryption to return $\perp$ on most $\mathbf{x}, \mathbf{y} \in \mathbb{F}^k$.

2. I looked at some fh-IPFE constructions.

   (a) In [DDM15; TAO16; Kim+16]

   - The field size $|\mathbb{F}| = |\mathbb{Z}_q| = q$ is of exponential size of λ.
   - The decryption relies on finding $\langle \mathbf{x}, \mathbf{y} \rangle$ from $g^{\langle \mathbf{x}, \mathbf{y} \rangle}$ for a group generator g of order q. Discrete logarithm is hard. $\mathsf{FE.Dec}$ works only when $\langle \mathbf{x}, \mathbf{y} \rangle$ ranges in a pre-defined polynomial-size set.
   - [DDM15; TAO16] are fh-IND secure. [Kim+16] is fh-IND secure in the generic group model.
   - One can sample a random ciphertext $\mathbf{c_v}$, but then it is difficult to find any $\mathbf{c_x}$ such that $\mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c_x}, \mathbf{c_v}) \neq \perp$.
   - [DDM15; Kim+16] are $\mathcal{O}'_{\mathsf{KeyGen}}$-RUF secure by Theorem 10.

   (b) In [Lee+18; Che+21]:

   - The field size $|\mathbb{F}| = |\mathbb{Z}_q| = q$ is polynomial of λ
   - They prove security in a game that only allows the adversary to call $\mathcal{O}_{\mathsf{KeyGen}}$ once, and it must be before $\mathcal{O}_{\mathsf{Enc}}$.
   - They are not fh-IND secure.
   - They are also not RUF secure because one can sample random ciphertext $\mathbf{c_v}$.

3. In general, I think RUF security cannot be easliy proven without adding a signature.

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. URL: https://doi.org/10.1145/1030083.1030096.

[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.

[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.

[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. 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.