Biometrics Authentication: Formalization and Instantiation

Keng-Yu Chen

October 7, 2024

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

1 Preliminaries

In this report, 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.

Definition 1 (Functional Hiding Inner Product Functional Encryption)**.** A *functional 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 $\mathbf{c}_{\mathbf{y}}$.

- $\mathsf{FE.Dec}(\mathsf{pp}, f_{\mathbf{x}}, \mathbf{c}_{\mathbf{y}}) \to z \in \mathbb{F}$: It outputs a value z.

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

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

2 Formalization

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

- $\mathsf{USetup}(1^\lambda) \to \mathsf{esk}, \mathsf{psk}, \mathsf{csk}$: 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.

- $\mathsf{Enroll}(\mathsf{esk}, \mathbf{x}) \to \mathbf{c}_{\mathbf{x}}$: It outputs the enrollment message $\mathbf{c}_{\mathbf{x}}$ 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.

- $\mathsf{Probe}(\mathsf{psk}, \mathbf{y}) \to \mathbf{c}_{\mathbf{y}}$: It outputs the probe message $\mathbf{c}_{\mathbf{y}}$ from $\mathbf{y}$.

- $\mathsf{Compare}(\mathsf{csk}, \mathbf{c}_{\mathbf{x}}, \mathbf{c}_{\mathbf{y}}) \to s$: It compares the enrollment message $\mathbf{c}_{\mathbf{x}}$ and probe message $\mathbf{c}_{\mathbf{y}}$ 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 ($r = 0$).

The usage model we consider is described in Figure 1 (on enrollment) and Figure 2 (on authentication).

- 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 is responsible for most computation and storing the enrollment secret key esk and probe secret key psk. It queries $\mathcal{O}_{\mathcal{B}}$ for biometric data through the Scanner.

- Server: The server responsible for authenticating the user. It stores the comparison secret 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.

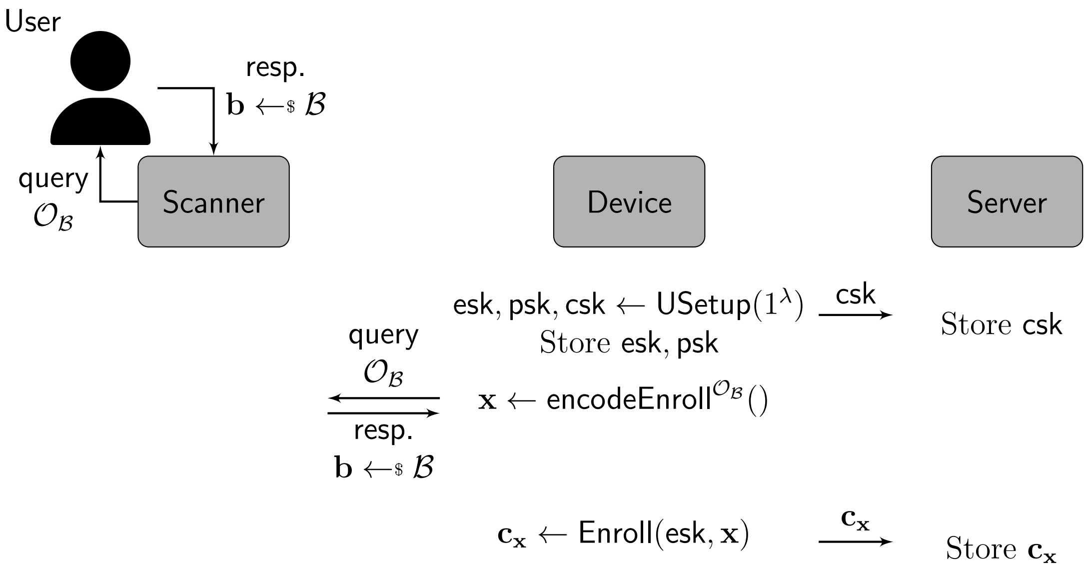

Figure 1: Usage Model on Enrollment

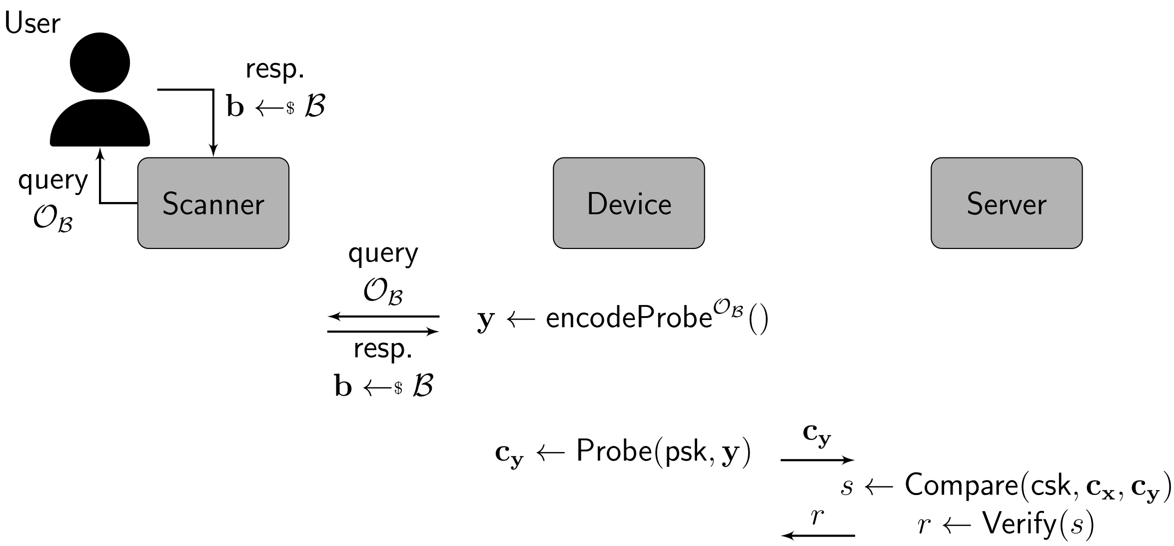

Figure 2: Usage Model on Authentication

For example, 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 templates $\mathbf{b}$ and $\mathbf{b}'$ be sampled from some 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.

- $\mathsf{USetup}(1^\lambda)$: It calls $\mathsf{FE.Setup}(1^\lambda) \to \mathsf{msk}, \mathsf{pp}$ and outputs $(\mathsf{esk}, \mathsf{psk}) \leftarrow ((\mathsf{msk}, \mathsf{pp}), (\mathsf{msk}, \mathsf{pp}))$ 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 $\sqrt{s} < \tau$, 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

$$s = \mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c_x}, \mathbf{c_y}) = \langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{k} -2b_i b'_i + \|\mathbf{b}\|^2 + \|\mathbf{b}'\|^2 = \|\mathbf{b} - \mathbf{b}'\|^2.$$

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 $\|\mathbf{b} - \mathbf{b}'\| < \tau$, the scheme results in $r = 1$, a successful authentication.

3 Security Games

From now on, we consider a family of biometric distributions $\mathbb{B}$. Removing a person $\mathcal{B}$ from $\mathbb{B}$ is written as $\mathbb{B} \setminus \mathcal{B}$.

3.1 Forgery Game

In the forgery game, we model the ability of a malicious $\mathsf{Scanner}$ who has access to the server's database of registered enrollments and tries to forge the user. The adversary $\mathcal{A}$ is given the enrollment message $\mathbf{c_x}$ and oracle $\mathcal{O}$ and tries to find a valid probe message $\tilde{\mathbf{z}}$. The whole game is defined in Figure 3.

The given oracle $\mathcal{O}$ consists of the following oracles:

- $\mathcal{O}_\mathcal{B}$: It outputs a biometric sample $\mathbf{b} \leftarrow_\$ \mathcal{B}$.
- $\mathcal{O}_{\mathsf{samp}}(\cdot)$: On input an index i,

$\underline{\mathsf{Forg}_{\Pi,\mathbb{B}}(\mathcal{A}^{\mathcal{O}})}$
$\mathcal{B} \leftarrow_\$ \mathbb{B}, \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$
$\mathsf{esk}, \mathsf{psk}, \mathsf{csk} \leftarrow \mathsf{USetup}(1^\lambda)$
$\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}()$
$\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$
$\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}}(\mathbf{c_x})$
$s \leftarrow \mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \tilde{\mathbf{z}})$
return $\mathsf{Verify}(s)$

Figure 3: The Forgery Game

$\underline{\mathsf{Forg'}_{\Pi,\mathbb{B}}(\mathcal{A}'^{\mathcal{O}'})}$
$\mathcal{B} \leftarrow_\$ \mathbb{B}, \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$
$\mathsf{esk}, \mathsf{psk}, \mathsf{csk} \leftarrow \mathsf{USetup}(1^\lambda)$
$\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_\mathcal{B}}()$
$\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$
$\tilde{\mathbf{z}} \leftarrow \mathcal{A}'^{\mathcal{O}'}()$
$s \leftarrow \mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \tilde{\mathbf{z}})$
return $\mathsf{Verify}(s)$

Figure 4: The Plain Forgery Game

  - If i was not queried before, it first samples a biometric distribution $\mathcal{B}_i \in \mathbb{B}$ and then outputs a biometric sample $\mathbf{b} \leftarrow_\$ \mathcal{B}_i$.
  - If i has been queried before, it outputs a biometric sample $\mathbf{b} \leftarrow_\$ \mathcal{B}_i$.

- $\mathcal{O}^q_{\mathsf{auth}}(\mathsf{csk}, \mathbf{c_x}, \cdot)$: If this oracle has been queried over q times, it aborts. Otherwise, on input $\mathbf{z}$, it outputs $\mathsf{Verify}(\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{z}))$.

Depending on the threat model, we can also consider the following oracles:

- $\mathcal{O}_{\mathsf{Enroll}}(\mathsf{esk}, \cdot)$: 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 $\mathsf{Probe}(\mathsf{psk}, \mathbf{y}')$.
- $\mathcal{O}^q_{\mathsf{Compare}}(\mathsf{csk}, \cdot, \cdot)$: If this oracle has been queried over q times, it aborts. Otherwise, on input $\mathbf{c}'_\mathbf{x}$ and $\mathbf{c}'_\mathbf{y}$, it outputs the comparison result $\mathsf{Compare}(\mathsf{csk}, \mathbf{c}'_\mathbf{x}, \mathbf{c}'_\mathbf{y})$.

Note that if the enrollment secret key esk, probe secret key psk, or the comparison secret key csk is an empty or public string in the scheme, then the corresponding oracles are naturally and implicitly given since the adversary can compute them itself.

To consider potential false positives of biometrics match, we consider the plain forgery game in Figure 4, in which the adversary does not have any knowledge about the template. The given oracle $\mathcal{O}'$ consists of:

- $\mathcal{O}_{\mathsf{samp}}(\cdot)$: On input an index i,
  - If i was not queried before, it first samples a biometric distribution $\mathcal{B}_i \in \mathbb{B}$ and then outputs a biometric sample $\mathbf{b} \leftarrow_\$ \mathcal{B}_i$.
  - If i has been queried before, it outputs a biometric sample $\mathbf{b} \leftarrow_\$ \mathcal{B}_i$.
- $\mathcal{O}^q_{\mathsf{auth}}(\mathsf{csk}, \mathbf{c_x}, \cdot)$: If this oracle has been queried over q times, it aborts. Otherwise, on input $\mathbf{z}$, it outputs $\mathsf{Verify}(\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{z}))$.

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

$$\mathbf{Adv}^{\mathsf{Forg}}_{\Pi,\mathbb{B},\mathcal{A}^{\mathcal{O}}} := \Pr[\mathsf{Forg}_{\Pi,\mathbb{B}}(\mathcal{A}^{\mathcal{O}}) \to 1] - \max_{\text{PPT } \mathcal{A}'} \Pr[\mathsf{Forg'}_{\Pi,\mathbb{B}}(\mathcal{A}'^{\mathcal{O}'}) \to 1].$$

An authentication scheme Π associated with a family of distributions $\mathbb{B}$ is called *forgery secure* if for any PPT adversary $\mathcal{A}$,

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

3.2 Simulation Game

In the simulation game, we model the ability of the Server who tries to learn something more than the comparison result of the enrollment and probe messages. The adversary $\mathcal{A}$ is given an enrollment message and a list of t probe messages, and it needs to guess whether these are real messages or simulation results of a simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_{\mathsf{Enroll}}, \mathcal{S}_{\mathsf{Probe}})$ based on the $\mathsf{Compare}$ function. Intuitively, the simulator receives a list of $\mathsf{Compare}$ results of real enrollment and probe messages, and it returns some manual enrollment or probe messages that look similar to real ones. The whole game is defined in Figure 5.

$\mathsf{SIM} - \mathsf{Real}_{\Pi,\mathbb{B}}(\mathcal{A}^{\mathcal{O}})$

$\mathcal{B} \leftarrow_{\$} \mathbb{B}, \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$
$\mathsf{esk}, \mathsf{psk}, \mathsf{csk} \leftarrow \mathsf{USetup}(1^\lambda)$
$\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}}}()$
$\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$
$\{\mathbf{y}^{(i)}\}_{i=1}^t \leftarrow \{\mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}}}()\}_{i=1}^t$
$\{\mathbf{c_y}^{(i)}\}_{i=1}^t \leftarrow \left\{\mathsf{Probe}(\mathsf{psk}, \mathbf{y}^{(i)})\right\}_{i=1}^t$
$b \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{csk}, \mathbf{c_x}, \{\mathbf{c_y}^{(i)}\}_{i=1}^t)$
return b

$\mathsf{SIM} - \mathsf{Ideal}_{\Pi,\mathbb{B}}(\mathcal{A}^{\tilde{\mathcal{O}}}, \mathcal{S})$

$\mathcal{B} \leftarrow_{\$} \mathbb{B}, \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$
$\mathsf{esk}, \mathsf{psk}, \mathsf{csk}, \leftarrow \mathsf{USetup}(1^\lambda)$
$\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}}}()$
$\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$
$\{\mathbf{y}^{(i)}\}_{i=1}^t \leftarrow \{\mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}}}()\}_{i=1}^t$
$\{\mathbf{c_y}^{(i)}\}_{i=1}^t \leftarrow \left\{\mathsf{Probe}(\mathsf{psk}, \mathbf{y}^{(i)})\right\}_{i=1}^t$
$\mathsf{C} \leftarrow \{\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_y}^{(i)})\}_{i=1}^t$
$\tilde{\mathbf{c}}_{\mathbf{x}}, \{\tilde{\mathbf{c}}_{\mathbf{y}}^{(i)}\}_{i=1}^t \leftarrow \mathcal{S}_0(\mathsf{csk}, \mathsf{C})$
$b \leftarrow \mathcal{A}^{\tilde{\mathcal{O}}}(\mathsf{csk}, \tilde{\mathbf{c}}_{\mathbf{x}}, \{\tilde{\mathbf{c}}_{\mathbf{y}}^{(i)}\}_{i=1}^t)$
return b

Figure 5: The Simulation Game

The given oracle $\mathcal{O}$ and $\tilde{\mathcal{O}}$ consist of the oracle $\mathcal{O}_{\mathsf{samp}}$:

- $\mathcal{O}_{\mathsf{samp}}(\cdot)$: On input an index i,
  - If i was not queried before, it first samples a biometric distribution $\mathcal{B}_i \in \mathbb{B}$ and then outputs a biometric sample $\mathbf{b} \leftarrow_{\$} \mathcal{B}_i$.
  - If i has been queried before, it outputs a biometric sample $\mathbf{b} \leftarrow_{\$} \mathcal{B}_i$.

Depending on the threat model, we can also consider the following oracles for $\mathcal{O}$:

- $\mathcal{O}_{\mathsf{Enroll}}(\mathsf{esk}, \cdot)$: 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 $\mathsf{Probe}(\mathsf{psk}, \mathbf{y}')$.

The corresponding oracles of $\mathcal{O}_{\mathsf{Enroll}}$ and $\mathcal{O}_{\mathsf{Probe}}$ for $\tilde{\mathcal{O}}$ are $\tilde{\mathcal{O}}_{\mathsf{Enroll}}$ and $\tilde{\mathcal{O}}_{\mathsf{Probe}}$. They are stateful and memorize all the previous queries, and they include two simulators: $\mathcal{S}_{\mathsf{Enroll}}$ and $\mathcal{S}_{\mathsf{Probe}}$.

- $\tilde{\mathcal{O}}_{\mathsf{Enroll}}(\mathsf{csk}, \mathsf{esk}, \cdot)$: On input $\mathbf{x}^{(j)}$, it updates the collection of comparison results C by adding the results with all previous probe messages $\mathbf{c_y}^{(i)}$. Then it calls the simulator $\mathcal{S}_{\mathsf{Enroll}}(\mathsf{csk}, \mathsf{C})$ and returns whatever the simulator returns.

- $\tilde{\mathcal{O}}_{\mathsf{Probe}}(\mathsf{csk}, \mathsf{psk}, \cdot)$: On input $\mathbf{y}^{(j)}$, it updates the collection of comparison results C by adding the results with all previous enrollment messages $\mathbf{c_x}^{(i)}$. Then it calls the simulator $\mathcal{S}_{\mathsf{Probe}}(\mathsf{csk}, \mathsf{C})$ and returns whatever the simulator returns.

The details of oracles $\tilde{\mathcal{O}}_{\mathsf{Enroll}}$ and $\tilde{\mathcal{O}}_{\mathsf{Probe}}$ are given in Figure 6.

$\tilde{\mathcal{O}}_{\mathsf{Enroll}}(\mathsf{csk}, \mathsf{esk}, \mathbf{x}^{(j)})$

$\mathbf{c_x}^{(j)} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{x}^{(j)})$
$\mathsf{C} \leftarrow \mathsf{C} \cup \{\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}^{(j)}, \mathbf{c_y}^{(i)})\}_i$
$\mathbf{c_x}' \leftarrow \mathcal{S}_{\mathsf{Enroll}}(\mathsf{csk}, \mathsf{C})$
return $\mathbf{c_x}'$

$\tilde{\mathcal{O}}_{\mathsf{Probe}}(\mathsf{csk}, \mathsf{psk}, \mathbf{y}^{(j)})$

$\mathbf{c_y}^{(j)} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{y}^{(j)})$
$\mathsf{C} \leftarrow \mathsf{C} \cup \{\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}^{(i)}, \mathbf{c_y}^{(j)})\}_i$
$\mathbf{c_y}' \leftarrow \mathcal{S}_{\mathsf{Probe}}(\mathsf{csk}, \mathsf{C})$
return $\mathbf{c_y}'$

Figure 6: Choice of the Oracle $\tilde{\mathcal{O}}$

We define the advantage of an adversary $\mathcal{A}$ and a simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_{\mathsf{Enroll}}, \mathcal{S}_{\mathsf{Probe}})$ in the simulation game of a scheme Π associated with a family of distributions $\mathbb{B}$ as

$$\mathbf{Adv}^{\mathsf{SIM}}_{\Pi,\mathbb{B},\mathcal{A}^{\mathcal{O},\tilde{\mathcal{O}}},\mathcal{S}} := |\Pr[\mathsf{SIM}-\mathsf{Real}_{\Pi,\mathbb{B}}(\mathcal{A}^{\mathcal{O}}) \to 1] - \Pr[\mathsf{SIM}-\mathsf{Ideal}_{\Pi,\mathbb{B}}(\mathcal{A}^{\tilde{\mathcal{O}}}, \mathcal{S}) \to 1]|.$$

An authentication scheme Π associated with a family of distributions $\mathbb{B}$ is called *simulation secure* if for any PPT adversary $\mathcal{A}$, there exists a PPT simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_{\mathsf{Enroll}}, \mathcal{S}_{\mathsf{Probe}})$ such that

$$\mathbf{Adv}^{\mathsf{SIM}}_{\Pi,\mathbb{B},\mathcal{A}^{\mathcal{O},\tilde{\mathcal{O}}},\mathcal{S}} = \mathsf{negl}.$$

3.3 Identification Game

In the identification game, we model the ability of the Server who tries to identify the user. The adversary $\mathcal{A}$ is given an enrollment message $\mathbf{c_x}$, a list of t probe messages $\{\mathbf{c_y}^{(i)}\}_{i=1}^t$, and oracles to two distinct distributions $\mathcal{B}^{(0)}, \mathcal{B}^{(1)}$. It tries to guess from which these messages are generated. The whole game is defined in Figure 7.

$\mathsf{Id}_{\Pi,\mathbb{B}}(\mathcal{A}^{\mathcal{O}})$

$b \leftarrow_\$ \{0,1\}$
$\mathcal{B}^{(0)}, \mathcal{B}^{(1)} \leftarrow_\$ \mathbb{B}, \mathbb{B} \leftarrow \mathbb{B} \setminus \{\mathcal{B}^{(0)}, \mathcal{B}^{(1)}\}$
$\mathsf{esk}, \mathsf{psk}, \mathsf{csk}, \leftarrow \mathsf{USetup}(1^\lambda)$
$\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}^{(b)}}}()$
$\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$
$\{\mathbf{y}^{(i)}\}_{i=1}^t \leftarrow \{\mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}^{(b)}}}()\}_{i=1}^t$
$\{\mathbf{c_y}^{(i)}\}_{i=1}^t \leftarrow \left\{\mathsf{Probe}(\mathsf{psk}, \mathbf{y}^{(i)})\right\}_{i=1}^t$
$\tilde{b} \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{csk}, \mathbf{c_x}, \{\mathbf{c_y}^{(i)}\}_{i=1}^t)$
return $1_{\tilde{b}=b}$

Figure 7: The Identification Game

The given oracle $\mathcal{O}$ consists of the following oracles:

- $\mathcal{O}_{\mathcal{B}^{(0)}}$: It outputs a biometric sample $\mathbf{b} \leftarrow_\$ \mathcal{B}^{(0)}$.
- $\mathcal{O}_{\mathcal{B}^{(1)}}$: It outputs a biometric sample $\mathbf{b} \leftarrow_\$ \mathcal{B}^{(1)}$.
- $\mathcal{O}_{\mathsf{samp}}(\cdot)$: On input an index i,
  - If i was not queried before, it first samples a biometric distribution $\mathcal{B}_i \in \mathbb{B}$ and then outputs a biometric sample $\mathbf{b} \leftarrow_\$ \mathcal{B}_i$.
  - If i has been queried before, it outputs a biometric sample $\mathbf{b} \leftarrow_\$ \mathcal{B}_i$.

Depending on the threat model, we can also consider the oracle:

- $\mathcal{O}_{\mathsf{Enroll}}(\mathsf{esk}, \cdot)$: On input $\mathbf{x}'$, it outputs the enrollment message $\mathsf{Enroll}(\mathsf{esk}, \mathbf{x}')$.

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

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

An authentication scheme Π associated with a family of distributions $\mathbb{B}$ is called *identification secure* if for any PPT adversary $\mathcal{A}$,

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

3.4 Reusability Game

In the reusability game, we model the ability of a malicious application who tries to forge the user in another application. The adversay $\mathcal{A}$ is given the enrollment message $\mathbf{c}_{\mathbf{x}}$ and oracle $\mathcal{O}$ and tries to find a valid probe message $\tilde{\mathbf{z}}$. The whole game is defined in Figure 8.

$\underline{\mathsf{Reuse}_{\Pi,\mathbb{B}}(\mathcal{A}^{\mathcal{O}})}$
$\mathcal{B} \leftarrow_\$ \mathbb{B}, \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$
$\mathsf{esk}, \mathsf{psk}, \mathsf{csk} \leftarrow \mathsf{USetup}(1^\lambda)$
$\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}}}()$
$\mathbf{c}_{\mathbf{x}} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$
$\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}}(\mathbf{c}_{\mathbf{x}})$
$s \leftarrow \mathsf{Compare}(\mathsf{csk}, \mathbf{c}_{\mathbf{x}}, \tilde{\mathbf{z}})$
return $\mathsf{Verify}(s)$

Figure 8: The Reusability Game

$\underline{\mathsf{Reuse}'_{\Pi,\mathbb{B}}(\mathcal{A}'^{\mathcal{O}'})}$
$\mathcal{B} \leftarrow_\$ \mathbb{B}, \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$
$\mathsf{esk}, \mathsf{psk}, \mathsf{csk} \leftarrow \mathsf{USetup}(1^\lambda)$
$\mathbf{x} \leftarrow \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}}}()$
$\mathbf{c}_{\mathbf{x}} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{x})$
$\tilde{\mathbf{z}} \leftarrow \mathcal{A}'^{\mathcal{O}'}()$
$s \leftarrow \mathsf{Compare}(\mathsf{csk}, \mathbf{c}_{\mathbf{x}}, \tilde{\mathbf{z}})$
return $\mathsf{Verify}(s)$

Figure 9: The Plain Reusability Game

The given oracle $\mathcal{O}$ consist of the following oracles:

- $\mathcal{O}_{\mathsf{samp}}(\cdot)$: On input an index i,
  - If i was not queried before, it first samples a biometric distribution $\mathcal{B}_i \in \mathbb{B}$ and then outputs a biometric sample $\mathbf{b} \leftarrow_\$ \mathcal{B}_i$.

  - If i has been queried before, it outputs a biometric sample $\mathbf{b} \leftarrow_{\$} \mathcal{B}_i$.

- $\mathcal{O}^{\mathsf{Re}}_{\mathsf{Enroll}}(\cdot)$: On input esk', it first samples $\mathbf{x}' \leftarrow_{\$} \mathsf{encodeEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ and outputs $\mathsf{Enroll}(\mathsf{esk}', \mathbf{x}')$.

- $\mathcal{O}^{\mathsf{Re}}_{\mathsf{Probe}}(\cdot)$: On input psk', it first samples $\mathbf{y}' \leftarrow_{\$} \mathsf{encodeProbe}^{\mathcal{O}_{\mathcal{B}}}()$ and outputs $\mathsf{Probe}(\mathsf{psk}', \mathbf{y}')$.

- $\mathcal{O}^{q}_{\mathsf{auth}}(\mathsf{csk}, \mathbf{c_x}, \cdot)$: If this oracle has been queried over q times, it aborts. Otherwise, on input $\mathbf{z}$, it outputs $\mathsf{Verify}(\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{z}))$.

Depending on the threat model, we can also consider the following oracles:

- $\mathcal{O}_{\mathsf{Enroll}}(\mathsf{esk}, \cdot)$: 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 $\mathsf{Probe}(\mathsf{psk}, \mathbf{y}')$.

- $\mathcal{O}^{q}_{\mathsf{Compare}}(\mathsf{csk}, \cdot, \cdot)$: If this oracle has been queried over q times, it aborts. Otherwise, on input $\mathbf{c}'_{\mathbf{x}}$ and $\mathbf{c}'_{\mathbf{y}}$, it outputs the comparison result $\mathsf{Compare}(\mathsf{csk}, \mathbf{c}'_{\mathbf{x}}, \mathbf{c}'_{\mathbf{y}})$.

Note that the resuability game is defined in a similar way as the forgery game in Section 3.1. The difference is their use cases and oracles. In the forgery game, the targeted adversary is an eavesdropper who has access to the server's database. If it can also access the user's device, it may be able to execute $\mathcal{O}_{\mathsf{Enroll}}$ and $\mathcal{O}_{\mathsf{Probe}}$. In contrast, the reusability game targets a malicious application, who can ask the user to enroll and probe with crafted secret keys. We assume the malicious application cannot access other functions in other applications.

To consider potential false positives of biometrics match, we consider the plain reusability game in Figure 9, in which the adversary does not have any knowledge about the template. The given oracle $\mathcal{O}'$ consists of:

- $\mathcal{O}_{\mathsf{samp}}(\cdot)$: On input an index i,
  - If i was not queried before, it first samples a biometric distribution $\mathcal{B}_i \in \mathbb{B}$ and then outputs a biometric sample $\mathbf{b} \leftarrow_{\$} \mathcal{B}_i$.
  - If i has been queried before, it outputs a biometric sample $\mathbf{b} \leftarrow_{\$} \mathcal{B}_i$.

- $\mathcal{O}^{q}_{\mathsf{auth}}(\mathsf{csk}, \mathbf{c_x}, \cdot)$: If this oracle has been queried over q times, it aborts. Otherwise, on input $\mathbf{z}$, it outputs $\mathsf{Verify}(\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{z}))$.

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

$$\mathbf{Adv}^{\mathsf{Reuse}}_{\Pi,\mathbb{B},\mathcal{A}^{\mathcal{O}}} := \Pr[\mathsf{Reuse}_{\Pi,\mathbb{B}}(\mathcal{A}^{\mathcal{O}}) \to 1] - \max_{\text{PPT } \mathcal{A}'} \Pr[\mathsf{Reuse}'_{\Pi,\mathbb{B}}(\mathcal{A}'^{\mathcal{O}'}) \to 1].$$

An authentication scheme Π associated with a family of distributions $\mathbb{B}$ is called *reusability secure* if for any PPT adversary $\mathcal{A}$,

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

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.

[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.

[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.