The Cryptographic Layer of Biometric Authentication

Keng-Yu Chen

LASEC

January 9th, 2025

Table of Contents

1. Introduction
2. Formalization
3. Security Models
4. Security Analysis
5. Conclusion

Table of Contents

1. Introduction
2. Formalization
3. Security Models
4. Security Analysis
5. Conclusion

Biometric Authentication

- An *error-tolerant* approach to user verification.
  - A server verifies identities by comparing the *similarity*, instead of equivalence.

Biometric Authentication

- An *error-tolerant* approach to user verification.
  - A server verifies identities by comparing the *similarity*, instead of equivalence.
- Unlike password, biometrics reveal personal information and cannot be changed.

Biometric Authentication

- An *error-tolerant* approach to user verification.
    - A server verifies identities by comparing the *similarity*, instead of equivalence.
- Unlike password, biometrics reveal personal information and cannot be changed.
- Possibly non-negligible false positive/negative rates.

Biometric Authentication

In this project, we formalize a biometric authenticaion scheme.

Biometric Authentication

In this project, we formalize a biometric authenticaion scheme.

- We formally define a biometric authenticaion scheme.

Biometric Authentication

In this project, we formalize a biometric authenticaion scheme.

- We formally define a biometric authenticaion scheme.
- We add a cryptographic layer on top of the authentication scheme to preserve privacy.

Biometric Authentication

In this project, we formalize a biometric authenticaion scheme.

- We formally define a biometric authenticaion scheme.
- We add a cryptographic layer on top of the authentication scheme to preserve privacy.
- We model two security notions of interest: *unforgeability* and *indistinguishability*

Biometric Authentication

In this project, we formalize a biometric authenticaion scheme.

- We formally define a biometric authenticaion scheme.
- We add a cryptographic layer on top of the authentication scheme to preserve privacy.
- We model two security notions of interest: *unforgeability* and *indistinguishability*
  - Unforgeability: An adversary cannot *impersonate* the user.

Biometric Authentication

In this project, we formalize a biometric authenticaion scheme.

- We formally define a biometric authenticaion scheme.
- We add a cryptographic layer on top of the authentication scheme to preserve privacy.
- We model two security notions of interest: *unforgeability* and *indistinguishability*
    - Unforgeability: An adversary cannot *impersonate* the user.
    - Indistinguishability: The server cannot *recognize* the user.

Biometric Authentication

In this project, we formalize a biometric authenticaion scheme.

- We formally define a biometric authenticaion scheme.
- We add a cryptographic layer on top of the authentication scheme to preserve privacy.
- We model two security notions of interest: *unforgeability* and *indistinguishability*
  - Unforgeability: An adversary cannot *impersonate* the user.
  - Indistinguishability: The server cannot *recognize* the user.
- We analyze two instantiations from previous works.

Biometric Authentication

In this project, we formalize a biometric authenticaion scheme.

- We formally define a biometric authenticaion scheme.
- We add a cryptographic layer on top of the authentication scheme to preserve privacy.
- We model two security notions of interest: *unforgeability* and *indistinguishability*
  - Unforgeability: An adversary cannot *impersonate* the user.
  - Indistinguishability: The server cannot *recognize* the user.
- We analyze two instantiations from previous works.
  - Function-hiding inner product functional encryption [EM23].

Biometric Authentication

In this project, we formalize a biometric authenticaion scheme.

- We formally define a biometric authenticaion scheme.
- We add a cryptographic layer on top of the authentication scheme to preserve privacy.
- We model two security notions of interest: *unforgeability* and *indistinguishability*
  - Unforgeability: An adversary cannot *impersonate* the user.
  - Indistinguishability: The server cannot *recognize* the user.
- We analyze two instantiations from previous works.
  - Function-hiding inner product functional encryption [EM23].
  - Relational hash [MR14].

Biometric Authentication

In this project, we formalize a biometric authenticaion scheme.

- We formally define a biometric authenticaion scheme.
- We add a cryptographic layer on top of the authentication scheme to preserve privacy.
- We model two security notions of interest: *unforgeability* and *indistinguishability*
  - Unforgeability: An adversary cannot *impersonate* the user.
  - Indistinguishability: The server cannot *recognize* the user.
- We analyze two instantiations from previous works.
  - Function-hiding inner product functional encryption [EM23].
  - Relational hash [MR14].

Notation

- λ: the security parameter.
- poly (negl) denotes a polynomial (negligible) function of λ.
- Sample a value r from a distribution $\mathcal{D}$ (uniformly from a set S) is $r \leftarrow_{\$} \mathcal{D}$ ($r \leftarrow_{\$} S$).

Table of Contents

1 Introduction

2 Formalization

3 Security Models

4 Security Analysis

5 Conclusion

Family of Biometric Distributions

Assume there is a family $\mathbb{B}$ of biometric distributions and the following algorithms:

Family of Biometric Distributions

Assume there is a family $\mathbb{B}$ of biometric distributions and the following algorithms:

- BioSamp(): Pick a random distribution $\mathcal{B} \in \mathbb{B}$. ($\mathcal{B} \leftarrow_\$ \mathbb{B}$)

Family of Biometric Distributions

Assume there is a family $\mathbb{B}$ of biometric distributions and the following algorithms:

- BioSamp(): Pick a random distribution $\mathcal{B} \in \mathbb{B}$. ($\mathcal{B} \leftarrow_{\$} \mathbb{B}$)
- BioDelete($\mathcal{B}$): Remove $\mathcal{B}$ from $\mathbb{B}$. ($\mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$).

Family of Biometric Distributions

Assume there is a family $\mathbb{B}$ of biometric distributions and the following algorithms:

- BioSamp(): Pick a random distribution $\mathcal{B} \in \mathbb{B}$. ($\mathcal{B} \leftarrow_{\$} \mathbb{B}$)
- BioDelete($\mathcal{B}$): Remove $\mathcal{B}$ from $\mathbb{B}$. ($\mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$).
- TempSamp($\mathcal{B}$): Sample a biometric template $\mathbf{b}$ from $\mathcal{B}$. ($\mathbf{b} \leftarrow_{\$} \mathcal{B}$)

Biometric Authentication

Biometric Authentication Scheme

A *biometric authentication shceme* Π associated with $\mathbb{B}$ consists of the following algorithms.

Biometric Authentication

Biometric Authentication Scheme

A *biometric authentication shceme* Π associated with $\mathbb{B}$ consists of the following algorithms.

- $\mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \to \mathbf{b}$: Given oracle $\mathcal{O}_{\mathcal{B}}$, output a biometric template $\mathbf{b}$ for enrollment.

- $\mathcal{O}_{\mathcal{B}}$: When queried, return a biometric template $\mathbf{b} \leftarrow_{\$} \mathcal{B}$.

Biometric Authentication

Biometric Authentication Scheme

A *biometric authentication shceme* Π associated with $\mathbb{B}$ consists of the following algorithms.

- $\mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \to \mathbf{b}$: Given oracle $\mathcal{O}_{\mathcal{B}}$, output a biometric template $\mathbf{b}$ for enrollment.
- $\mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}}}() \to \mathbf{b}'$: Given oracle $\mathcal{O}_{\mathcal{B}}$, output a biometric template $\mathbf{b}'$ for probe.

- $\mathcal{O}_{\mathcal{B}}$: When queried, return a biometric template $\mathbf{b} \leftarrow_{\$} \mathcal{B}$.

Biometric Authentication

Biometric Authentication Scheme

A *biometric authentication shceme* Π associated with $\mathbb{B}$ consists of the following algorithms.

- $\mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \to \mathbf{b}$: Given oracle $\mathcal{O}_{\mathcal{B}}$, output a biometric template $\mathbf{b}$ for enrollment.
- $\mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}}}() \to \mathbf{b}'$: Given oracle $\mathcal{O}_{\mathcal{B}}$, output a biometric template $\mathbf{b}'$ for probe.
- $\mathsf{BioCompare}(\mathbf{b}, \mathbf{b}') \to s$: Given two templates $\mathbf{b}$ and $\mathbf{b}'$, output a score s.

- $\mathcal{O}_{\mathcal{B}}$: When queried, return a biometric template $\mathbf{b} \leftarrow_{\$} \mathcal{B}$.

Biometric Authentication

Biometric Authentication Scheme

A *biometric authentication shceme* Π associated with $\mathbb{B}$ consists of the following algorithms.

- $\mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}() \to \mathbf{b}$: Given oracle $\mathcal{O}_{\mathcal{B}}$, output a biometric template $\mathbf{b}$ for enrollment.
- $\mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}}}() \to \mathbf{b}'$: Given oracle $\mathcal{O}_{\mathcal{B}}$, output a biometric template $\mathbf{b}'$ for probe.
- $\mathsf{BioCompare}(\mathbf{b}, \mathbf{b}') \to s$: Given two templates $\mathbf{b}$ and $\mathbf{b}'$, output a score s.
- $\mathsf{Verify}(s) \to r \in \{0, 1\}$: Determine whether this is a successful authentication ($r = 1$) or not ($r = 0$).

- $\mathcal{O}_{\mathcal{B}}$: When queried, return a biometric template $\mathbf{b} \leftarrow_{\$} \mathcal{B}$.

Cryptographic Layer

Cryptographic Layer

- $\mathsf{Setup}(1^\lambda) \to \mathsf{esk}, \mathsf{psk}, \mathsf{csk}$: Output the *enrollment secret key esk*, *probe secret key psk*, and *comparison secret key* csk.

Cryptographic Layer

- $\mathsf{Setup}(1^\lambda) \to \mathsf{esk}, \mathsf{psk}, \mathsf{csk}$: Output the *enrollment secret key esk*, *probe secret key psk*, and *comparison secret key* csk.
- $\mathsf{Enroll}(\mathsf{esk}, \mathbf{b}) \to \mathbf{c_x}$: On input a biometric template $\mathbf{b}$, it encodes it into a vector $\mathbf{x}$ and outputs the enrollment message $\mathbf{c_x}$.

Cryptographic Layer

- $\mathsf{Setup}(1^\lambda) \to \mathsf{esk}, \mathsf{psk}, \mathsf{csk}$: Output the *enrollment secret key esk*, *probe secret key psk*, and *comparison secret key* csk.
- $\mathsf{Enroll}(\mathsf{esk}, \mathbf{b}) \to \mathbf{c_x}$: On input a biometric template $\mathbf{b}$, it encodes it into a vector $\mathbf{x}$ and outputs the enrollment message $\mathbf{c_x}$.
- $\mathsf{Probe}(\mathsf{psk}, \mathbf{b}') \to \mathbf{c_y}$: On input a biometric template $\mathbf{b}'$, it encodes it into a vector $\mathbf{y}$ and outputs the probe message $\mathbf{c_y}$.

Cryptographic Layer

- $\mathsf{Setup}(1^\lambda) \to \mathsf{esk}, \mathsf{psk}, \mathsf{csk}$: Output the *enrollment secret key esk*, *probe secret key psk*, and *comparison secret key* csk.
- $\mathsf{Enroll}(\mathsf{esk}, \mathbf{b}) \to \mathbf{c_x}$: On input a biometric template $\mathbf{b}$, it encodes it into a vector $\mathbf{x}$ and outputs the enrollment message $\mathbf{c_x}$.
- $\mathsf{Probe}(\mathsf{psk}, \mathbf{b}') \to \mathbf{c_y}$: On input a biometric template $\mathbf{b}'$, it encodes it into a vector $\mathbf{y}$ and outputs the probe message $\mathbf{c_y}$.
- $\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_y}) \to s$: It compares the enrollment message $\mathbf{c_x}$ and probe message $\mathbf{c_y}$ and outputs a score s.

Cryptographic Layer

Correctness

For any $\mathcal{B}, \mathcal{B}' \in \mathbb{B}$,

Cryptographic Layer

Correctness

For any $\mathcal{B}, \mathcal{B}' \in \mathbb{B}$, let $\mathbf{b} \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$, $\mathbf{b}' \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}'}}()$,

Cryptographic Layer

Correctness

For any $\mathcal{B}, \mathcal{B}' \in \mathbb{B}$, let $\mathbf{b} \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$, $\mathbf{b}' \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}'}}()$, $\mathsf{esk}, \mathsf{psk}, \mathsf{csk} \leftarrow \mathsf{Setup}(1^{\lambda})$,

Cryptographic Layer

Correctness

For any $\mathcal{B}, \mathcal{B}' \in \mathbb{B}$, let $\mathbf{b} \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$, $\mathbf{b}' \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}'}}()$, $\mathsf{esk}, \mathsf{psk}, \mathsf{csk} \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{b})$,

Cryptographic Layer

Correctness

For any $\mathcal{B}, \mathcal{B}' \in \mathbb{B}$, let $\mathbf{b} \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$, $\mathbf{b}' \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}'}}()$, $\mathsf{esk}, \mathsf{psk}, \mathsf{csk} \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{b})$, $\mathbf{c_y} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{b}')$.

Cryptographic Layer

Correctness

For any $\mathcal{B}, \mathcal{B}' \in \mathbb{B}$, let $\mathbf{b} \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$, $\mathbf{b}' \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}'}}()$, $\mathsf{esk}, \mathsf{psk}, \mathsf{csk} \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{b})$, $\mathbf{c_y} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{b}')$. Then

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

Usage Example: Enrollment

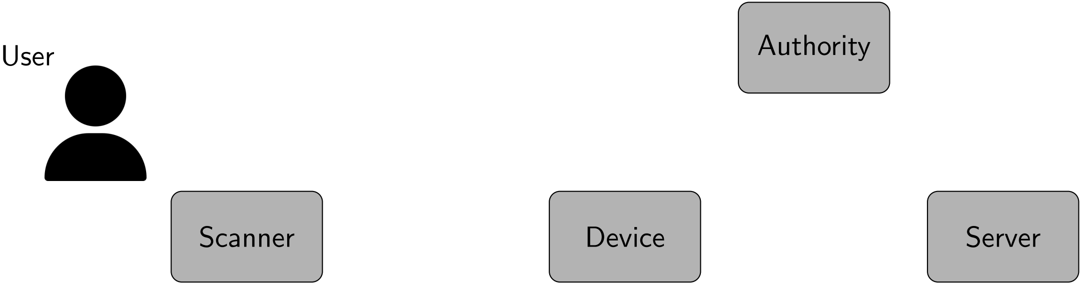

Figure: Usage Example: Enrollment

Usage Example: Enrollment

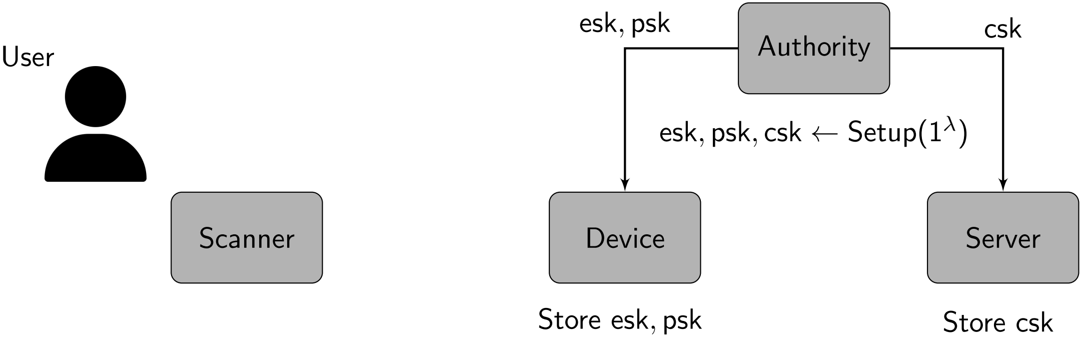

Figure: Usage Example: Enrollment

Usage Example: Enrollment

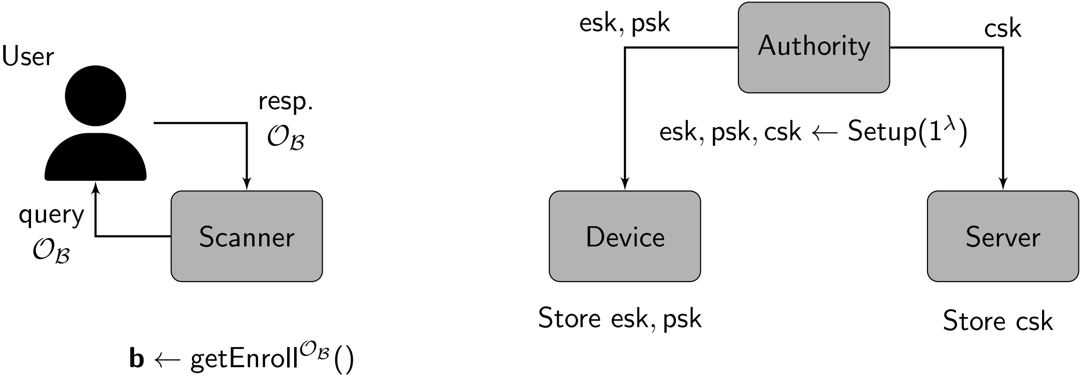

Figure: Usage Example: Enrollment

Usage Example: Enrollment

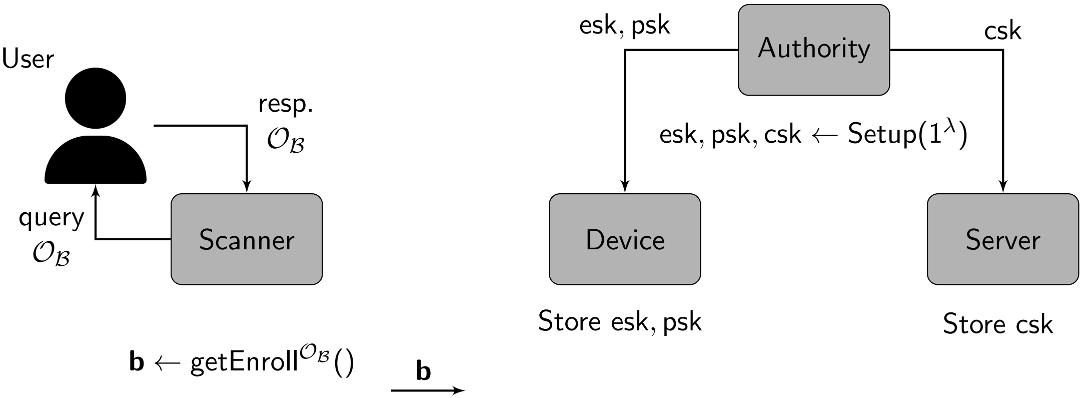

Figure: Usage Example: Enrollment

Usage Example: Enrollment

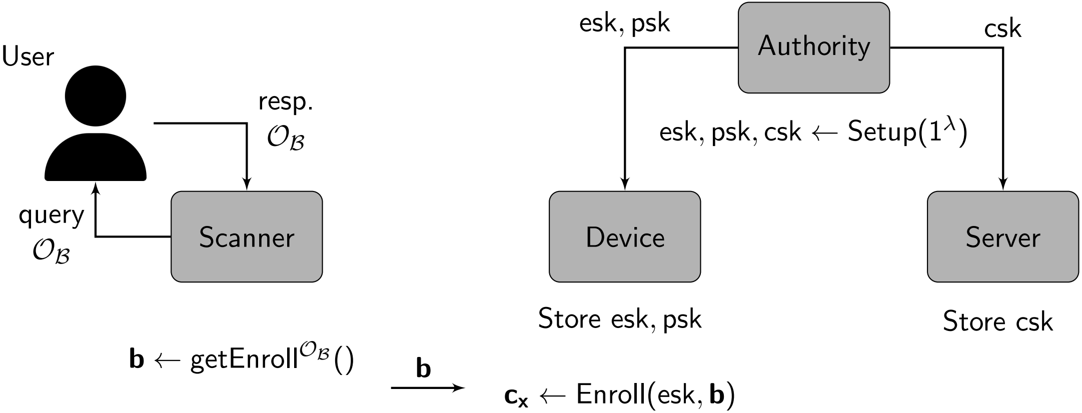

Figure: Usage Example: Enrollment

Usage Example: Enrollment

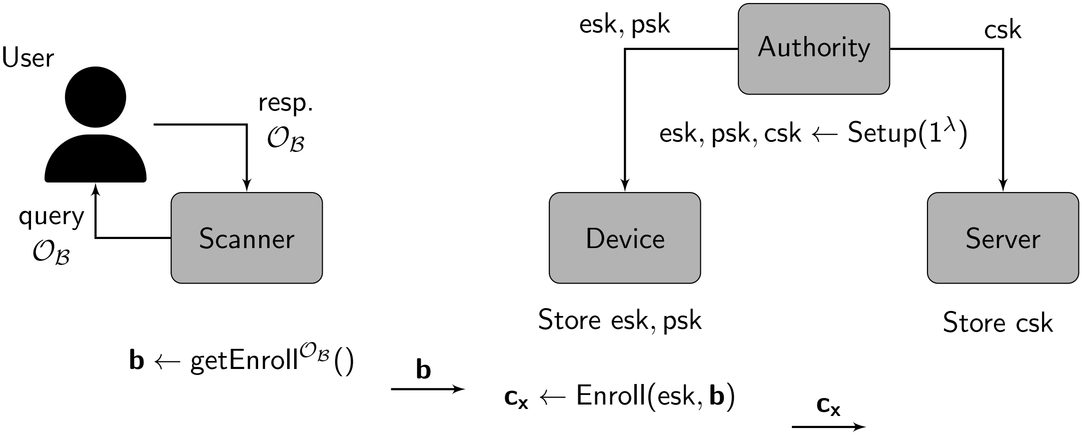

Figure: Usage Example: Enrollment

Usage Example: Enrollment

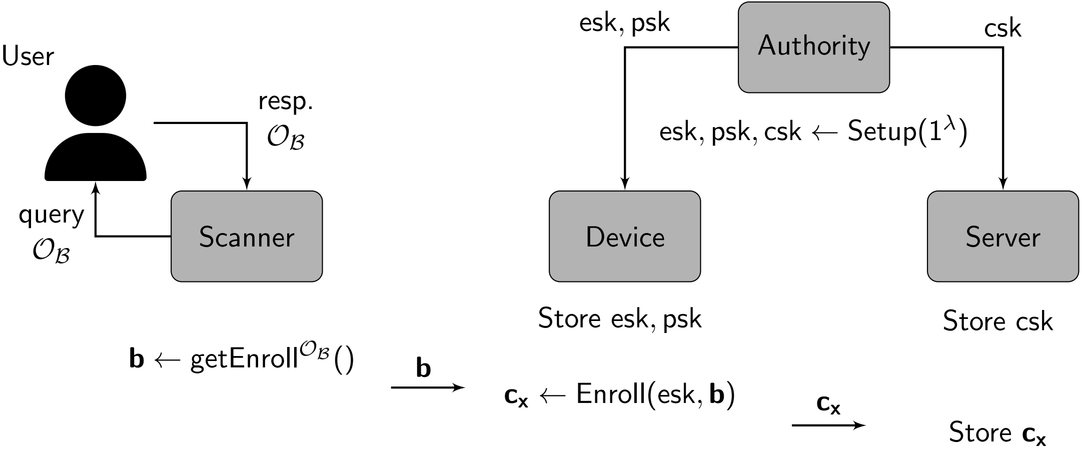

Figure: Usage Example: Enrollment

Usage Example: Authentication

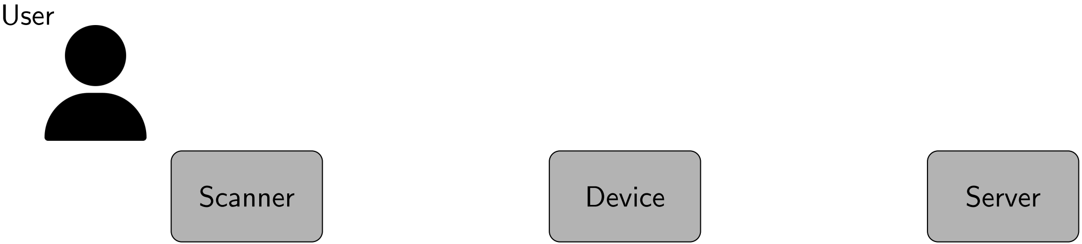

Figure: Usage Example: Authentication

Usage Example: Authentication

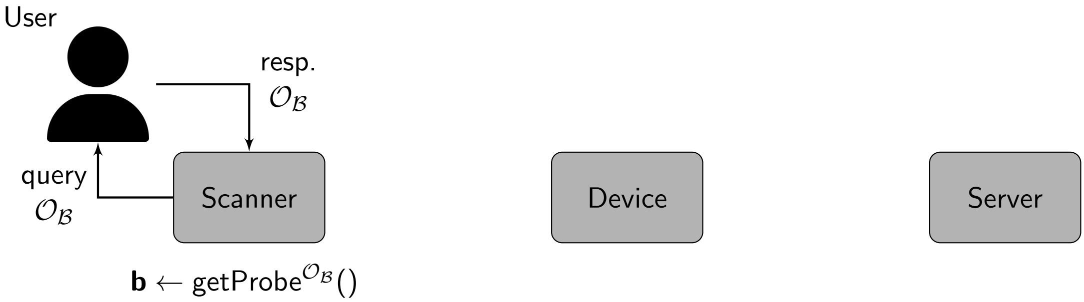

Figure: Usage Example: Authentication

Usage Example: Authentication

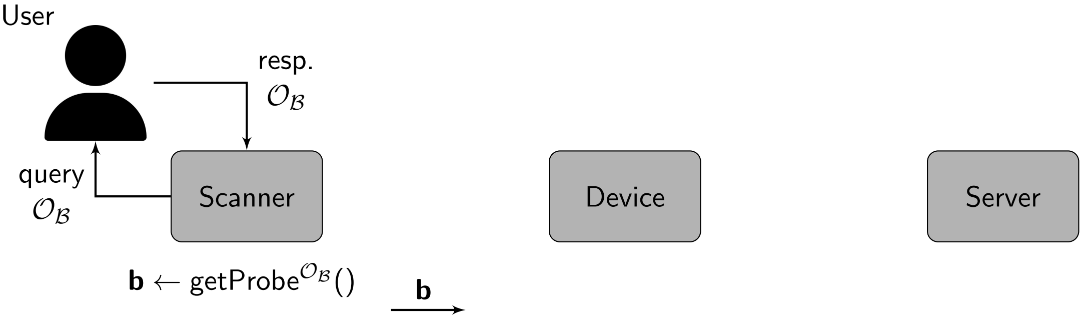

Figure: Usage Example: Authentication

Usage Example: Authentication

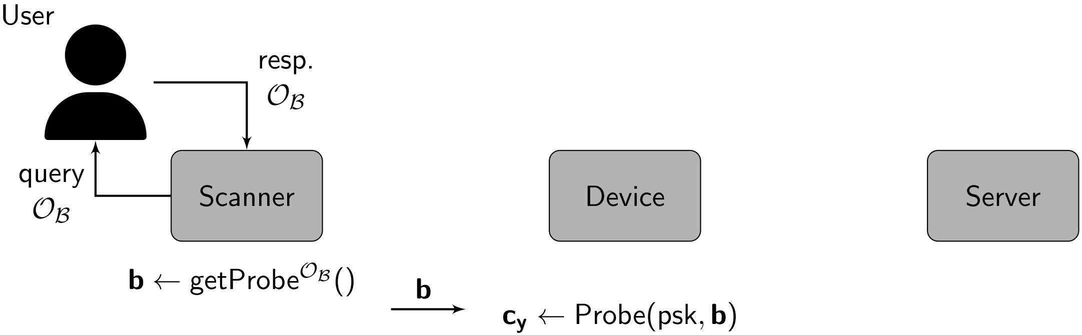

Figure: Usage Example: Authentication

Usage Example: Authentication

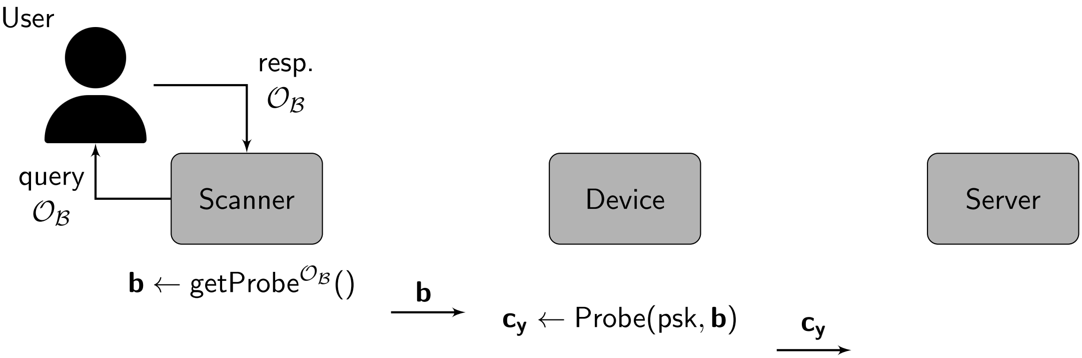

Figure: Usage Example: Authentication

Usage Example: Authentication

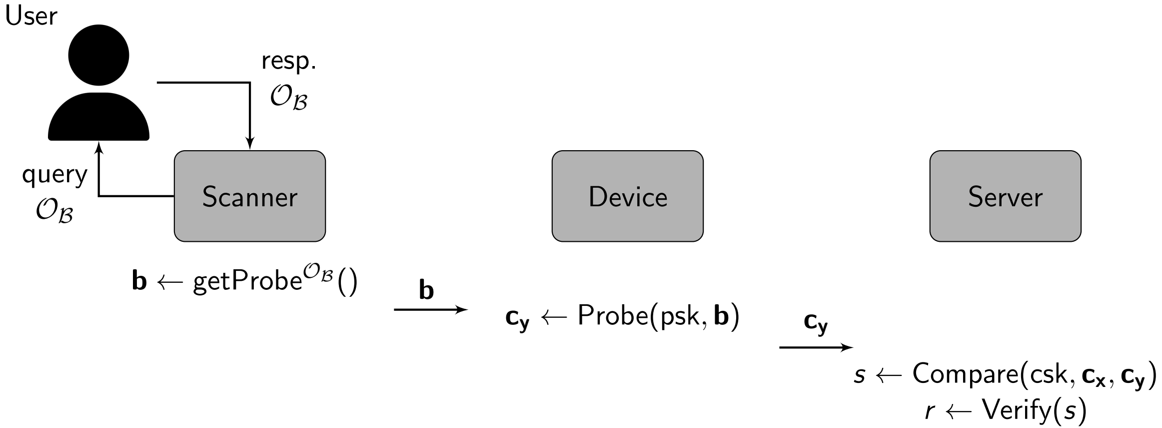

Figure: Usage Example: Authentication

Table of Contents

1. Introduction
2. Formalization
3. Security Models
   - Unforgeability
   - Indistinguishability
4. Security Analysis
5. Conclusion

Table of Contents

1 Introduction

2 Formalization

3 Security Models
- Unforgeability
- Indistinguishability

4 Security Analysis

5 Conclusion

Unforgeability

Let Π be a biometric authentication scheme. For an adversary $\mathcal{A}$, define $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}(\mathcal{A})$.

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

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: $\mathbf{b} \leftarrow \mathsf{getEnroll}^{\mathcal{O}_\mathcal{B}}()$
4: $\mathbf{c_x} \leftarrow \mathsf{Enroll}(\mathsf{esk}, \mathbf{b})$
5: $\tilde{\mathbf{z}} \leftarrow \mathcal{A}(\mathsf{option})$
6: $s \leftarrow \mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \tilde{\mathbf{z}})$
7: **return** $\mathsf{Verify}(s)$

Π is called *option-unforgeable* (option-UF) if for any PPT adversary $\mathcal{A}$,

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

Possible Choice of option

- Enrollment message: $\mathbf{c_x}$.

Possible Choice of option

- Enrollment message: $\mathbf{c_x}$.
- Keys: $\mathsf{esk}, \mathsf{psk}, \mathsf{csk}$.

Possible Choice of option

- Enrollment message: $\mathbf{c_x}$.
- Keys: esk, psk, csk.
- Oracle $\mathcal{O}_{\mathcal{B}}$.

Possible Choice of option

- Enrollment message: $\mathbf{c}_{\mathbf{x}}$.
- Keys: $\mathsf{esk}, \mathsf{psk}, \mathsf{csk}$.
- Oracle $\mathcal{O}_{\mathcal{B}}$.
- Oracle $\mathcal{O}_{\mathsf{Enroll}}(\mathsf{esk}, \cdot)$: On input $\mathbf{b}'$, output $\mathsf{Enroll}(\mathsf{esk}, \mathbf{b}')$.

Possible Choice of option

- Enrollment message: $\mathbf{c_x}$.
- Keys: $\mathsf{esk}, \mathsf{psk}, \mathsf{csk}$.
- Oracle $\mathcal{O}_{\mathcal{B}}$.
- Oracle $\mathcal{O}_{\mathsf{Enroll}}(\mathsf{esk}, \cdot)$: On input $\mathbf{b}'$, output $\mathsf{Enroll}(\mathsf{esk}, \mathbf{b}')$.
- Oracle $\mathcal{O}_{\mathsf{Probe}}(\mathsf{psk}, \cdot)$: On input $\mathbf{b}'$, output $\mathsf{Probe}(\mathsf{psk}, \mathbf{b}')$.

Possible Choice of option

- Enrollment message: $\mathbf{c_x}$.
- Keys: $\mathsf{esk}, \mathsf{psk}, \mathsf{csk}$.
- Oracle $\mathcal{O}_{\mathcal{B}}$.
- Oracle $\mathcal{O}_{\mathsf{Enroll}}(\mathsf{esk}, \cdot)$: On input $\mathbf{b}'$, output $\mathsf{Enroll}(\mathsf{esk}, \mathbf{b}')$.
- Oracle $\mathcal{O}_{\mathsf{Probe}}(\mathsf{psk}, \cdot)$: On input $\mathbf{b}'$, output $\mathsf{Probe}(\mathsf{psk}, \mathbf{b}')$.

Note that some combinations of option induce a winning probability *true positive rate* or *false positive rate*.

True/False Positive Rates

True/False Positive Rates

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

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

Possible Choice of option

- psk and $\mathcal{O}_{\mathcal{B}}$ are not given at the same time.

Possible Choice of option

- psk and $\mathcal{O}_{\mathcal{B}}$ are not given at the same time.
  - $\mathcal{A}_1$ has a winning probability TP.

$\mathcal{A}_1^{\mathcal{O}_{\mathcal{B}}}(\mathsf{psk})$

1: $\mathbf{b} \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$
2: $\mathbf{c_y} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{b})$
3: **return** $\mathbf{c_y}$

Possible Choice of option

- psk and $\mathcal{O}_{\mathcal{B}}$ are not given at the same time.
  - $\mathcal{A}_1$ has a winning probability TP.
- psk is given only when FP is negligible.

$\mathcal{A}_1^{\mathcal{O}_{\mathcal{B}}}(\mathsf{psk})$

1: $\mathbf{b} \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$
2: $\mathbf{c_y} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{b})$
3: **return** $\mathbf{c_y}$

Possible Choice of option

- psk and $\mathcal{O}_{\mathcal{B}}$ are not given at the same time.
  - $\mathcal{A}_1$ has a winning probability TP.
- psk is given only when FP is negligible.
  - $\mathcal{A}_2$ has a winning probability FP.

$\mathcal{A}_1^{\mathcal{O}_{\mathcal{B}}}(\mathsf{psk})$

1: $\mathbf{b} \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$
2: $\mathbf{c_y} \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{b})$
3: **return** $\mathbf{c_y}$

$\mathcal{A}_2(\mathsf{psk})$

1: $\mathcal{B}' \leftarrow_{\$} \mathbb{B}$.
2: $\mathbf{b}' \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}'}}()$
3: $\mathbf{c_y}' \leftarrow \mathsf{Probe}(\mathsf{psk}, \mathbf{b}')$
4: **return** $\mathbf{c_y}'$

Possible Choice of option

- psk and $\mathcal{O}_{\mathcal{B}}$ are not given at the same time.
  - $\mathcal{A}_1$ has a winning probability TP.
- psk is given only when FP is negligible.
  - $\mathcal{A}_2$ has a winning probability FP.
- When $\mathcal{O}_{\text{Probe}}$ is given, we forbid the adversary to return an answer of $\mathcal{O}_{\text{Probe}}$.

$\mathcal{A}_1^{\mathcal{O}_{\mathcal{B}}}(\text{psk})$

1: $\mathbf{b} \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$
2: $\mathbf{c_y} \leftarrow \text{Probe}(\text{psk}, \mathbf{b})$
3: **return** $\mathbf{c_y}$

$\mathcal{A}_2(\text{psk})$

1: $\mathcal{B}' \leftarrow_{\$} \mathbb{B}$.
2: $\mathbf{b}' \leftarrow \text{getProbe}^{\mathcal{O}_{\mathcal{B}'}}()$
3: $\mathbf{c_y}' \leftarrow \text{Probe}(\text{psk}, \mathbf{b}')$
4: **return** $\mathbf{c_y}'$

Table of Contents

1. Introduction
2. Formalization
3. Security Models
   - Unforgeability
   - Indistinguishability
4. Security Analysis
5. Conclusion

Indistinguishability

Let t be an integer. For an adversary $\mathcal{A}$, define $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$.

$\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$

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

Indistinguishability

- In $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$, we model the server's knowledge about the user.

Indistinguishability

- In $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$, we model the server's knowledge about the user.
- Π is called *indistinguishable* (IND) if for any PPT adversary $\mathcal{A}$,

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

Table of Contents

1 Introduction

2 Formalization

3 Security Models

4 Security Analysis
- Instantiation using fh-IPFE
- Security of fh-IPFE
- Security of Instantiation using fh-IPFE

5 Conclusion

Security Analysis

In this section, we will discuss

- Function-hiding inner product functional encryption (fh-IPFE) [Kim+16].
- An instantiation Π using an fh-IPFE [EM23].
- UF and IND security of Π.

In our project, we also discuss an instantiation using *relational hash*.

Table of Contents

1 Introduction

2 Formalization

3 Security Models

4 Security Analysis
- Instantiation using fh-IPFE
- Security of fh-IPFE
- Security of Instantiation using fh-IPFE

5 Conclusion

fh-IPFE

Function-Hiding Inner Product Functional Encryption (adapted from [Kim+16])

A *function-hiding inner product functional encryption* (fh-IPFE) scheme FE for a field $\mathbb{F}$ is composed of PPT algorithms:

fh-IPFE

Function-Hiding Inner Product Functional Encryption (adapted from [Kim+16])

A *function-hiding inner product functional encryption* (fh-IPFE) scheme FE for a field $\mathbb{F}$ is composed of PPT algorithms:

- $\mathsf{FE.Setup}(1^\lambda)$: Output the public parameter pp and the master secret key msk.

fh-IPFE

Function-Hiding Inner Product Functional Encryption (adapted from [Kim+16])

A *function-hiding inner product functional encryption* (fh-IPFE) scheme FE for a field $\mathbb{F}$ is composed of PPT algorithms:

- $\mathsf{FE.Setup}(1^\lambda)$: Output the public parameter pp and the master secret key msk.
- $\mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{x})$: On input a vector $\mathbf{x} \in \mathbb{F}^k$, output the decryption key $f_{\mathbf{x}}$.

fh-IPFE

Function-Hiding Inner Product Functional Encryption (adapted from [Kim+16])

A *function-hiding inner product functional encryption* (fh-IPFE) scheme FE for a field $\mathbb{F}$ is composed of PPT algorithms:

- $\mathsf{FE.Setup}(1^\lambda)$: Output the public parameter pp and the master secret key msk.
- $\mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{x})$: On input a vector $\mathbf{x} \in \mathbb{F}^k$, output the decryption key $f_{\mathbf{x}}$.
- $\mathsf{FE.Enc}(\mathsf{msk}, \mathsf{pp}, \mathbf{y})$: On input a vector $\mathbf{y} \in \mathbb{F}^k$, output the ciphertext $\mathbf{c}_{\mathbf{y}}$.

fh-IPFE

Function-Hiding Inner Product Functional Encryption (adapted from [Kim+16])

A *function-hiding inner product functional encryption* (fh-IPFE) scheme FE for a field $\mathbb{F}$ is composed of PPT algorithms:

- $\mathsf{FE.Setup}(1^\lambda)$: Output the public parameter pp and the master secret key msk.
- $\mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{x})$: On input a vector $\mathbf{x} \in \mathbb{F}^k$, output the decryption key $f_\mathbf{x}$.
- $\mathsf{FE.Enc}(\mathsf{msk}, \mathsf{pp}, \mathbf{y})$: On input a vector $\mathbf{y} \in \mathbb{F}^k$, output the ciphertext $\mathbf{c_y}$.
- $\mathsf{FE.Dec}(\mathsf{pp}, f_\mathbf{x}, \mathbf{c_y})$: Output a value $z \in \mathbb{F}$ or an error symbol $\perp$.

fh-IPFE

Correctness

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 fh-IPFE [EM23]

- $\text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$, $\text{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$: Output vectors in $\{0, 1, \cdots, m\}^k$ for all $\mathcal{B} \in \mathbb{B}$.

Instantiation using fh-IPFE [EM23]

- $\text{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$, $\text{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$: Output vectors in $\{0, 1, \cdots, m\}^k$ for all $\mathcal{B} \in \mathbb{B}$.
- $\text{BioCompare}(\mathbf{b}, \mathbf{b}') \to \|\mathbf{b} - \mathbf{b}'\|^2$.

Instantiation using fh-IPFE [EM23]

- $\mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$, $\mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$: Output vectors in $\{0, 1, \cdots, m\}^k$ for all $\mathcal{B} \in \mathbb{B}$.
- $\mathsf{BioCompare}(\mathbf{b}, \mathbf{b}') \to \|\mathbf{b} - \mathbf{b}'\|^2$.
- For a pre-defined real number $\tau \geq 0$,

$$\mathsf{Verify}(s) \to \begin{cases} 1 & \text{if } \sqrt{s} \leq \tau \\ 0 & \text{if } \sqrt{s} > \tau \end{cases}.$$

Instantiation using fh-IPFE [EM23]

- $\mathsf{getEnroll}^{\mathcal{O}_\mathcal{B}}()$, $\mathsf{getProbe}^{\mathcal{O}_\mathcal{B}}()$: Output vectors in $\{0, 1, \cdots, m\}^k$ for all $\mathcal{B} \in \mathbb{B}$.
- $\mathsf{BioCompare}(\mathbf{b}, \mathbf{b}') \to \|\mathbf{b} - \mathbf{b}'\|^2$.
- For a pre-defined real number $\tau \geq 0$,

$$\mathsf{Verify}(s) \to \begin{cases} 1 & \text{if } \sqrt{s} \leq \tau \\ 0 & \text{if } \sqrt{s} > \tau \end{cases}.$$

- An fh-IPFE FE associated with a field $\mathbb{F} = \mathbb{Z}_q$.
  - q is a prime number larger than the maximum possible Euclidean distance $m^2 \cdot k$.

Instantiation using fh-IPFE [EM23]

- $\mathsf{Setup}(1^\lambda)$: Run $\mathsf{FE.Setup}(1^\lambda) \to \mathsf{msk}, \mathsf{pp}$ and output
  - $\mathsf{esk} \leftarrow (\mathsf{msk}, \mathsf{pp})$
  - $\mathsf{psk} \leftarrow (\mathsf{msk}, \mathsf{pp})$
  - $\mathsf{csk} \leftarrow \mathsf{pp}$

Instantiation using fh-IPFE [EM23]

- $\mathsf{Setup}(1^\lambda)$: Run $\mathsf{FE.Setup}(1^\lambda) \to \mathsf{msk}, \mathsf{pp}$ and output
  - $\mathsf{esk} \leftarrow (\mathsf{msk}, \mathsf{pp})$
  - $\mathsf{psk} \leftarrow (\mathsf{msk}, \mathsf{pp})$
  - $\mathsf{csk} \leftarrow \mathsf{pp}$
- $\mathsf{Enroll}(\mathsf{esk}, \mathbf{b})$: On input a template vector $\mathbf{b} = (b_1, b_2, \cdots, b_k)$,
  - Encode $\mathbf{b}$ as $\mathbf{x} = (x_1, x_2, \cdots, x_{k+2}) = (b_1, b_2, \cdots, b_k, 1, \|\mathbf{b}\|^2)$.
  - Run and output $\mathbf{c}_{\mathbf{x}} \leftarrow \mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{x})$.

Instantiation using fh-IPFE [EM23]

- $\mathsf{Setup}(1^\lambda)$: Run $\mathsf{FE.Setup}(1^\lambda) \to \mathsf{msk}, \mathsf{pp}$ and output
  - $\mathsf{esk} \leftarrow (\mathsf{msk}, \mathsf{pp})$
  - $\mathsf{psk} \leftarrow (\mathsf{msk}, \mathsf{pp})$
  - $\mathsf{csk} \leftarrow \mathsf{pp}$
- $\mathsf{Enroll}(\mathsf{esk}, \mathbf{b})$: On input a template vector $\mathbf{b} = (b_1, b_2, \cdots, b_k)$,
  - Encode $\mathbf{b}$ as $\mathbf{x} = (x_1, x_2, \cdots, x_{k+2}) = (b_1, b_2, \cdots, b_k, 1, \|\mathbf{b}\|^2)$.
  - Run and output $\mathbf{c_x} \leftarrow \mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{x})$.
- $\mathsf{Probe}(\mathsf{psk}, \mathbf{b}')$: On input a template vector $\mathbf{b}' = (b'_1, b'_2, \cdots, b'_k)$,
  - Encode $\mathbf{b}'$ as $\mathbf{y} = (y_1, y_2, \cdots, y_{k+2}) = (-2b'_1, -2b'_2, \cdots, -2b'_k, \|\mathbf{b}'\|^2, 1)$.
  - Run and output $\mathbf{c_y} \leftarrow \mathsf{FE.Enc}(\mathsf{msk}, \mathsf{pp}, \mathbf{y})$.

Instantiation using fh-IPFE [EM23]

- Setup(1^λ): Run FE.Setup(1^λ) $\to$ msk, pp and output
  - esk $\leftarrow$ (msk, pp)
  - psk $\leftarrow$ (msk, pp)
  - csk $\leftarrow$ pp
- Enroll(esk, $\mathbf{b}$): On input a template vector $\mathbf{b} = (b_1, b_2, \cdots, b_k)$,
  - Encode $\mathbf{b}$ as $\mathbf{x} = (x_1, x_2, \cdots, x_{k+2}) = (b_1, b_2, \cdots, b_k, 1, \|\mathbf{b}\|^2)$.
  - Run and output $\mathbf{c_x} \leftarrow$ FE.KeyGen(msk, pp, $\mathbf{x}$).
- Probe(psk, $\mathbf{b}'$): On input a template vector $\mathbf{b}' = (b'_1, b'_2, \cdots, b'_k)$,
  - Encode $\mathbf{b}'$ as $\mathbf{y} = (y_1, y_2, \cdots, y_{k+2}) = (-2b'_1, -2b'_2, \cdots, -2b'_k, \|\mathbf{b}'\|^2, 1)$.
  - Run and output $\mathbf{c_y} \leftarrow$ FE.Enc(msk, pp, $\mathbf{y}$).
- Compare(csk, $\mathbf{c_x}$, $\mathbf{c_y}$): Run and output $s \leftarrow$ FE.Dec(pp, $\mathbf{c_x}$, $\mathbf{c_y}$).

Instantiation using fh-IPFE [EM23]

By the correctness of FE,

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

which is equal to $\mathrm{BioCompare}(\mathbf{b}, \mathbf{b}')$.

Instantiation using fh-IPFE [EM23]

By the correctness of FE,

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

which is equal to $\mathrm{BioCompare}(\mathbf{b}, \mathbf{b}')$.
Note that in this instantiation,

Instantiation using fh-IPFE [EM23]

By the correctness of FE,

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

which is equal to $\mathrm{BioCompare}(\mathbf{b}, \mathbf{b}')$.
Note that in this instantiation,

- $\mathrm{esk} = \mathrm{psk}$.

Instantiation using fh-IPFE [EM23]

By the correctness of FE,

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

which is equal to BioCompare$(\mathbf{b}, \mathbf{b}')$.
Note that in this instantiation,

- esk $=$ psk.
- csk $=$ pp, the public parameter of FE. Therefore, we offer csk for all adversaries.

Table of Contents

1. Introduction
2. Formalization
3. Security Models
4. Security Analysis
   - Instantiation using fh-IPFE
   - Security of fh-IPFE
   - Security of Instantiation using fh-IPFE
5. Conclusion

fh-IND of fh-IPFE

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

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

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

fh-IND of fh-IPFE

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

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

A trivial adversary can ask $\mathcal{O}_{\mathsf{KeyGen}}(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$ and $\mathcal{O}_{\mathsf{Enc}}(\mathbf{y}^{(0)}, \mathbf{y}^{(1)})$ such that

$$\mathbf{x}^{(0)}{\mathbf{y}^{(0)}}^T \neq \mathbf{x}^{(1)}{\mathbf{y}^{(1)}}^T.$$

fh-IND of fh-IPFE

Admissible Adversary

Let $\mathcal{A}$ be an adversary in an fh-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 $\mathcal{O}_{\mathsf{Enc}}$.

fh-IND of fh-IPFE

Admissible Adversary

Let $\mathcal{A}$ be an adversary in an fh-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 $\forall i \in [Q_K], \forall j \in [Q_E]$,

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

fh-IND of fh-IPFE

Admissible Adversary

Let $\mathcal{A}$ be an adversary in an fh-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 $\forall i \in [Q_K], \forall j \in [Q_E]$,

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

fh-IND Security

FE is called *fh-IND* secure if for any admissible adversary $\mathcal{A}$,

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

fh-IND Security

- fh-IND security is the standard notion of an fh-IPFE scheme.
    - Constructions in [DDM15; TAO16; Kim+16] are proven fh-IND.

fh-IND Security

- fh-IND security is the standard notion of an fh-IPFE scheme.
  - Constructions in [DDM15; TAO16; Kim+16] are proven fh-IND.
- However, fh-IND security is not sufficient for the UF security of Π.
  - We found that instantiation Π using [Kim+16] is not option-unforgeable for any option.

fh-IND Security

- fh-IND security is the standard notion of an fh-IPFE scheme.
  - Constructions in [DDM15; TAO16; Kim+16] are proven fh-IND.
- However, fh-IND security is not sufficient for the UF security of Π.
  - We found that instantiation Π using [Kim+16] is not option-unforgeable for any option.
- For this, we define another extra security notion of FE: *RUF Security*.

RUF of fh-IPFE

Let $\gamma \geq 0$ be a real number and $\mathbb{F} = \mathbb{Z}_q$ for a prime number q. Define the RUF game.

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

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: $s \leftarrow \mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c}, \tilde{\mathbf{z}})$
6: **return** $1_{s \leq \gamma}$

Oracle $\mathcal{O}$ can be nothing or include

- $\mathcal{O}'_{\mathsf{KeyGen}}(\cdot)$: On input $\mathbf{x}'$, output $\mathsf{FE.KeyGen}(\mathsf{msk}, \mathsf{pp}, \mathbf{x}')$.
- $\mathcal{O}'_{\mathsf{Enc}}(\cdot)$: On input $\mathbf{y}'$, output $\mathsf{FE.Enc}(\mathsf{msk}, \mathsf{pp}, \mathbf{y}')$.

RUF of fh-IPFE

- We forbid the adversary to return $\tilde{\mathbf{z}}$ that is an answer of $\mathcal{O}'_{\mathsf{Enc}}(\cdot)$.
  - Otherwise, returning $\mathcal{O}'_{\mathsf{Enc}}(\mathbf{0})$ wins with probability 1.

RUF Security

FE is called $\mathcal{O}$-RUF secure for a real number γ if for any adversary $\mathcal{A}$,

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

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

RUF of fh-IPFE

- RUF security is a new notion of an fh-IPFE scheme.

RUF of fh-IPFE

- RUF security is a new notion of an fh-IPFE scheme.
- In our project, we provide two theorems to obtain RUF security.

RUF of fh-IPFE

- RUF security is a new notion of an fh-IPFE scheme.
- In our project, we provide two theorems to obtain RUF security.

Theorem

If FE is fh-IND, and if the RUF adversary can only return $\tilde{\mathbf{z}}$ that is an encryption of a nonzero vector, then FE is $\mathcal{O}'_{KeyGen}$-RUF for any $\gamma \leq \|\mathbb{F}\|$.

RUF of fh-IPFE

- RUF security is a new notion of an fh-IPFE scheme.
- In our project, we provide two theorems to obtain RUF security.

Theorem

If FE is fh-IND, and if the RUF adversary can only return $\tilde{\mathbf{z}}$ *that is an encryption of a nonzero vector, then FE is* $\mathcal{O}'_{KeyGen}$*-RUF for any* $\gamma \leq \|\mathbb{F}\|$.

Theorem

Given an sEUF-CMA digital signature scheme Sig and any fh-IPFE FE, we can obtain an fh-IPFE FE′ that is RUF for any γ.

RUF of fh-IPFE

- RUF security is a new notion of an fh-IPFE scheme.
- In our project, we provide two theorems to obtain RUF security.

Theorem

If FE is fh-IND, and if the RUF adversary can only return $\tilde{\mathbf{z}}$ that is an encryption of a nonzero vector, then FE is $\mathcal{O}'_{KeyGen}$-RUF for any $\gamma \leq \|\mathbb{F}\|$.

Theorem

Given an sEUF-CMA digital signature scheme Sig and any fh-IPFE FE, we can obtain an fh-IPFE FE′ that is RUF for any γ.

We provide details in Appendix - Achievability of RUF Security.

Table of Contents

1. Introduction
2. Formalization
3. Security Models
4. Security Analysis
   - Instantiation using fh-IPFE
   - Security of fh-IPFE
   - Security of Instantiation using fh-IPFE
5. Conclusion

Security of Instantiation using fh-IPFE

For the rest of this section, let Π be a biomtric authentication scheme instantiated by an fh-IPFE FE.

Security of Instantiation using fh-IPFE

For the rest of this section, let Π be a biomtric authentication scheme instantiated by an fh-IPFE FE. In our project, we show

Security of Instantiation using fh-IPFE

For the rest of this section, let Π be a biomtric authentication scheme instantiated by an fh-IPFE FE. In our project, we show

Theorem

For any distribution family $\mathbb{B}$, if FE is fh-IND and $\mathcal{O}'_{KeyGen}$-RUF for a $\gamma \geq \tau^2$, then Π is $\{\mathbf{c_x}, csk, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{Enroll}\}$-UF.

Security of Instantiation using fh-IPFE

For the rest of this section, let Π be a biomtric authentication scheme instantiated by an fh-IPFE FE. In our project, we show

Theorem

For any distribution family $\mathbb{B}$, if FE is fh-IND and $\mathcal{O}'_{KeyGen}$-RUF for a $\gamma \geq \tau^2$, then Π is $\{\mathbf{c_x}, csk, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{Enroll}\}$-UF.

Theorem

For any distribution family $\mathbb{B}$, if FE is fh-IND and $\mathcal{O}'_{Enc}$-RUF for a $\gamma \geq \tau^2$, then Π is $\{\mathbf{c_x}, csk, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{Probe}\}$-UF.

Security of Instantiation using fh-IPFE

For the rest of this section, let Π be a biomtric authentication scheme instantiated by an fh-IPFE FE. In our project, we show

Theorem

For any distribution family $\mathbb{B}$, if FE is fh-IND and $\mathcal{O}'_{KeyGen}$-RUF for a $\gamma \geq \tau^2$, then Π is $\{\mathbf{c_x}, csk, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{Enroll}\}$-UF.

Theorem

For any distribution family $\mathbb{B}$, if FE is fh-IND and $\mathcal{O}'_{Enc}$-RUF for a $\gamma \geq \tau^2$, then Π is $\{\mathbf{c_x}, csk, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{Probe}\}$-UF.

Theorem

For any distribution family $\mathbb{B}$ satisfying some "reasonable conditions", if FE is fh-IND, then Π is IND.

Security of Instantiation using fh-IPFE

For the rest of this section, let Π be a biomtric authentication scheme instantiated by an fh-IPFE FE. In our project, we show

Theorem

For any distribution family $\mathbb{B}$, if FE is fh-IND and $\mathcal{O}'_{KeyGen}$-RUF for a $\gamma \geq \tau^2$, then Π is $\{\mathbf{c_x}, csk, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{Enroll}\}$-UF.

Theorem

For any distribution family $\mathbb{B}$, if FE is fh-IND and $\mathcal{O}'_{Enc}$-RUF for a $\gamma \geq \tau^2$, then Π is $\{\mathbf{c_x}, csk, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{Probe}\}$-UF.

Theorem

For any distribution family $\mathbb{B}$ satisfying some "reasonable conditions", if FE is fh-IND, then Π is IND.

$\{\mathbf{c_x}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Enroll}}\}$-UF

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

$\{\mathbf{c_x}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Enroll}}\}$-UF

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- If the challenge bit $b = 0$, $\mathcal{R}$ simulates a $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}(\mathcal{A})$ game.

$\{\mathbf{c_x}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Enroll}}\}$-UF

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- If the challenge bit $b = 0$, $\mathcal{R}$ simulates a $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}(\mathcal{A})$ game.
- If the challenge bit $b = 1$, $\mathcal{R}$ simulates a $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O}'_{\mathsf{KeyGen}},\gamma}(\mathcal{A}')$ game, for some $\mathcal{A}'$.

$\{\mathbf{c_x}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Enroll}}\}$-UF

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- If the challenge bit $b = 0$, $\mathcal{R}$ simulates a $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}(\mathcal{A})$ game.
- If the challenge bit $b = 1$, $\mathcal{R}$ simulates a $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O}'_{\mathsf{KeyGen}},\gamma}(\mathcal{A}')$ game, for some $\mathcal{A}'$.
- Advantage of $\mathcal{R}$ is bounded by the difference between advantages of $\mathcal{A}$ and $\mathcal{A}'$, and

$$\mathsf{Adv}^{\mathsf{UF}}_{\Pi,\mathbb{B},\mathcal{A},\mathsf{option}} \leq 4 \cdot \mathsf{Adv}^{\mathsf{fh\text{-}IND}}_{\mathsf{FE},\mathcal{R}} + \mathsf{Adv}^{\mathsf{RUF},\mathcal{O}'_{\mathsf{KeyGen}},\gamma}_{\mathsf{FE},\mathcal{A}'} = \mathsf{negl}.$$

$\{\mathbf{c_x}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Enroll}}\}$-UF

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- If the challenge bit $b = 0$, $\mathcal{R}$ simulates a $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}(\mathcal{A})$ game.
- If the challenge bit $b = 1$, $\mathcal{R}$ simulates a $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O}'_{\mathsf{KeyGen}},\gamma}(\mathcal{A}')$ game, for some $\mathcal{A}'$.
- Advantage of $\mathcal{R}$ is bounded by the difference between advantages of $\mathcal{A}$ and $\mathcal{A}'$, and

$$\mathsf{Adv}^{\mathsf{UF}}_{\Pi,\mathbb{B},\mathcal{A},\mathsf{option}} \leq 4 \cdot \mathsf{Adv}^{\mathsf{fh\text{-}IND}}_{\mathsf{FE},\mathcal{R}} + \mathsf{Adv}^{\mathsf{RUF},\mathcal{O}'_{\mathsf{KeyGen}},\gamma}_{\mathsf{FE},\mathcal{A}'} = \mathsf{negl}.$$

- $\mathcal{O}_{\mathsf{Enroll}}$ is "encoding + FE.KeyGen". We can simulate $\mathcal{O}_{\mathsf{Enroll}}$ by $\mathcal{O}_{\mathsf{KeyGen}}$ in fh-IND game and $\mathcal{O}'_{\mathsf{KeyGen}}$ in RUF game.

□

$\{\mathbf{c}_{\mathbf{x}}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Enroll}}\}$-UF

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- If the challenge bit $b = 0$, $\mathcal{R}$ simulates a $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}(\mathcal{A})$ game.
- If the challenge bit $b = 1$, $\mathcal{R}$ simulates a $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O}'_{\mathsf{KeyGen}},\gamma}(\mathcal{A}')$ game, for some $\mathcal{A}'$.
- Advantage of $\mathcal{R}$ is bounded by the difference between advantages of $\mathcal{A}$ and $\mathcal{A}'$, and

$$\mathsf{Adv}^{\mathsf{UF}}_{\Pi,\mathbb{B},\mathcal{A},\mathsf{option}} \leq 4 \cdot \mathsf{Adv}^{\mathsf{fh\text{-}IND}}_{\mathsf{FE},\mathcal{R}} + \mathsf{Adv}^{\mathsf{RUF},\mathcal{O}'_{\mathsf{KeyGen}},\gamma}_{\mathsf{FE},\mathcal{A}'} = \mathsf{negl}.$$

- $\mathcal{O}_{\mathsf{Enroll}}$ is "encoding + FE.KeyGen". We can simulate $\mathcal{O}_{\mathsf{Enroll}}$ by $\mathcal{O}_{\mathsf{KeyGen}}$ in fh-IND game and $\mathcal{O}'_{\mathsf{KeyGen}}$ in RUF game.
- $\mathcal{R}$ never calls $\mathcal{O}_{\mathsf{Enc}}$, so it is an admissible adversary.

□

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

$\mathcal{B} \leftarrow_{\$} \mathbb{B}, \quad \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$
$\mathbf{b} = (b_1, \cdots, b_k) \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$
$\mathbf{x} \leftarrow (b_1, \cdots, b_k, 1, \|\mathbf{b}\|^2)$
$\mathbf{r} \leftarrow_{\$} \mathbb{F}^{k+2}$
$\mathbf{c} \leftarrow \mathcal{O}_{\mathsf{KeyGen}}(\mathbf{x}, \mathbf{r})$
$\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{B}},\mathcal{O}_{\mathsf{Enroll}}}(\mathbf{c}, \mathsf{pp})$
$s \leftarrow \mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c}, \tilde{\mathbf{z}})$
if $\mathsf{Verify}(s) = 1$ **then**
    return $\tilde{b} = 0$
else
    return $\tilde{b} \leftarrow_{\$} \{0, 1\}$

Security of Instantiation using fh-IPFE

For the rest of this section, let Π be a biomtric authentication scheme instantiated by an fh-IPFE FE. In our project, we show

Theorem

For any distribution family $\mathbb{B}$, if FE is fh-IND and $\mathcal{O}'_{KeyGen}$-RUF for a $\gamma \geq \tau^2$, then Π is $\{\mathbf{c_x}, csk, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{Enroll}\}$-UF.

Theorem

For any distribution family $\mathbb{B}$, if FE is fh-IND and $\mathcal{O}'_{Enc}$-RUF for a $\gamma \geq \tau^2$, then Π is $\{\mathbf{c_x}, csk, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{Probe}\}$-UF.

Theorem

For any distribution family $\mathbb{B}$ satisfying some "reasonable conditions", if FE is fh-IND, then Π is IND.

$\{\mathbf{c_x}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Probe}}\}$-UF

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

$\{\mathbf{c_x}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Probe}}\}$-UF

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- If the challenge bit $b = 0$, $\mathcal{R}$ simulates a $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}(\mathcal{A})$ game.
- If the challenge bit $b = 1$, $\mathcal{R}$ simulates a $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O}'_{\mathsf{Enc}},\gamma}(\mathcal{A}')$ game, for some $\mathcal{A}'$.

$\{\mathbf{c}_{\mathbf{x}}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Probe}}\}$-UF

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- If the challenge bit $b = 0$, $\mathcal{R}$ simulates a $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}(\mathcal{A})$ game.
- If the challenge bit $b = 1$, $\mathcal{R}$ simulates a $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O}'_{\mathsf{Enc}},\gamma}(\mathcal{A}')$ game, for some $\mathcal{A}'$.
- To let $\mathcal{R}$ be admissible, we cannot directly simulate $\mathcal{O}_{\mathsf{Probe}}$ by $\mathcal{O}_{\mathsf{Enc}}$.
  - $\mathcal{R}$ uses $\mathcal{O}_{\mathsf{KeyGen}}(\mathbf{x}, \mathbf{r})$ to prepare $\mathbf{c}$ for $\mathcal{A}$ and $\mathcal{A}'$.

$\{\mathbf{c_x}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Probe}}\}$-UF

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- If the challenge bit $b = 0$, $\mathcal{R}$ simulates a $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}(\mathcal{A})$ game.
- If the challenge bit $b = 1$, $\mathcal{R}$ simulates a $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O}'_{\mathsf{Enc}},\gamma}(\mathcal{A}')$ game, for some $\mathcal{A}'$.
- To let $\mathcal{R}$ be admissible, we cannot directly simulate $\mathcal{O}_{\mathsf{Probe}}$ by $\mathcal{O}_{\mathsf{Enc}}$.
  - $\mathcal{R}$ uses $\mathcal{O}_{\mathsf{KeyGen}}(\mathbf{x}, \mathbf{r})$ to prepare $\mathbf{c}$ for $\mathcal{A}$ and $\mathcal{A}'$.
- We simulate $\mathcal{O}_{\mathsf{Probe}}(\mathsf{psk}, \mathbf{b}')$ by
  - Encode $\mathbf{b}'$ to $\mathbf{y}' = (-2b'_1, \cdots, -2b'_k, \|\mathbf{b}'\|^2, 1)$
  - Compute $d \leftarrow \mathbf{x}\mathbf{y}'^T$ and find a vector $\mathbf{y}''$ such that $\mathbf{r}\mathbf{y}''^T = d$
  - $\mathcal{O}_{\mathsf{Enc}}(\mathbf{y}', \mathbf{y}'')$.

- $\mathcal{R}$ is now admissible, but then we have to simulate the tweaked $\mathcal{O}_{\mathsf{Probe}}$ in $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O}'_{\mathsf{Enc}},\gamma}(\mathcal{A}')$ game.

$\square$

- $\mathcal{R}$ is now admissible, but then we have to simulate the tweaked $\mathcal{O}_{\mathsf{Probe}}$ in $\mathsf{RUF}_{\mathsf{FE}}^{\mathcal{O}'_{\mathsf{Enc}},\gamma}(\mathcal{A}')$ game.
- Advantage of $\mathcal{R}$ is bounded by the difference between advantages of $\mathcal{A}$ and $\mathcal{A}'$, and

$$\mathsf{Adv}^{\mathsf{UF}}_{\Pi,\mathbb{B},\mathcal{A},\mathsf{option}} \leq 4 \cdot \mathsf{Adv}^{\mathsf{fh\text{-}IND}}_{\mathsf{FE},\mathcal{R}} + \mathsf{Adv}^{\mathsf{RUF},\mathcal{O}'_{\mathsf{Enc}},\gamma}_{\mathsf{FE},\mathcal{A}'} + \frac{k+2}{q^{k+2}-1} + \frac{1}{q^{k+2}} = \mathsf{negl}.$$

$\square$

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

$\mathcal{B} \leftarrow_{\$} \mathbb{B}, \quad \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}$
$\mathbf{b} = (b_1, \cdots, b_k) \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$
$\mathbf{x} \leftarrow (b_1, \cdots, b_k, 1, \|\mathbf{b}\|^2)$
$\mathbf{r} \leftarrow_{\$} \mathbb{F}^{k+2}$
$\mathbf{c} \leftarrow \mathcal{O}_{\mathsf{KeyGen}}(\mathbf{x}, \mathbf{r})$
$\tilde{\mathbf{z}} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{B}},\mathcal{O}_{\mathsf{Probe}}}(\mathbf{c}, \mathsf{pp})$
if $\tilde{\mathbf{z}}$ is equal to any output of $\mathcal{O}_{\mathsf{Probe}}$ **then**
    return $\perp$
$s \leftarrow \mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c}, \tilde{\mathbf{z}})$
if $\mathsf{Verify}(s) = 1$ **then**
    return $\tilde{b} = 0$
else
    return $\tilde{b} \leftarrow_{\$} \{0, 1\}$

Security of Instantiation using fh-IPFE

For the rest of this section, let Π be a biomtric authentication scheme instantiated by an fh-IPFE FE. In our project, we show

Theorem

For any distribution family $\mathbb{B}$, if FE is fh-IND and $\mathcal{O}'_{KeyGen}$-RUF for a $\gamma \geq \tau^2$, then Π is $\{\mathbf{c_x}, csk, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{Enroll}\}$-UF.

Theorem

For any distribution family $\mathbb{B}$, if FE is fh-IND and $\mathcal{O}'_{Enc}$-RUF for a $\gamma \geq \tau^2$, then Π is $\{\mathbf{c_x}, csk, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{Probe}\}$-UF.

Theorem

For any distribution family $\mathbb{B}$ satisfying some "reasonable conditions", if FE is fh-IND, then Π is IND.

IND

Assumption 1

Let t be an integer. Assume that for any distribution $\mathcal{B} \in \mathbb{B}$, the following distribution is identical.

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

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

IND

Assumption 1

Let t be an integer. Assume that for any distribution $\mathcal{B} \in \mathbb{B}$, the following distribution is identical.

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

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

Note that indistinguishability of $\mathcal{D}_{\mathcal{B}}(t)$ for $\mathcal{B} \in \mathbb{B}$ is a necessary condition to achieve IND security because

$$\left(\mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_y}^{(1)}), \cdots, \mathsf{Compare}(\mathsf{csk}, \mathbf{c_x}, \mathbf{c_y}^{(t)})\right) = \mathcal{D}_{\mathcal{B}}(t)$$

where b is the challenge bit.

IND

Theorem

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

IND

Theorem

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

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\text{IND}_{\Pi,\mathbb{B}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

IND

Theorem

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

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\text{IND}_{\Pi,\mathbb{B}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- $\mathcal{R}$ first samples $\mathcal{B}^{(0)}$ and $\mathcal{B}^{(1)}$.

IND

Theorem

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

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\mathsf{IND}_{\Pi,\mathbb{B}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- $\mathcal{R}$ first samples $\mathcal{B}^{(0)}$ and $\mathcal{B}^{(1)}$.
- $\mathcal{R}$ then calls $\mathbf{c_x} \leftarrow \mathcal{O}_{\mathsf{KeyGen}}(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$, where $\mathbf{x}^{(0)}$ and $\mathbf{x}^{(1)}$ are created from $\mathcal{B}^{(0)}$ and $\mathcal{B}^{(1)}$.

IND

Theorem

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

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\text{IND}_{\Pi,\mathbb{B}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- $\mathcal{R}$ first samples $\mathcal{B}^{(0)}$ and $\mathcal{B}^{(1)}$.
- $\mathcal{R}$ then calls $\mathbf{c_x} \leftarrow \mathcal{O}_{\text{KeyGen}}(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$, where $\mathbf{x}^{(0)}$ and $\mathbf{x}^{(1)}$ are created from $\mathcal{B}^{(0)}$ and $\mathcal{B}^{(1)}$.
- $\mathcal{R}$ prepares $\mathbf{y}^{(0)}$ and $\mathbf{y}^{(1)}$ from $\mathcal{B}^{(0)}$ and $\mathcal{B}^{(1)}$ in a way that, $\mathbf{x}^{(0)}\mathbf{y}^{(0)\,T} = \mathbf{x}^{(1)}\mathbf{y}^{(1)\,T}$, and calls $\mathbf{c_y} \leftarrow \mathcal{O}_{\text{Enc}}(\mathbf{y}^{(0)}, \mathbf{y}^{(1)})$.

IND

Theorem

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

Proof Sketch.

Given an adversary $\mathcal{A}$ in the $\text{IND}_{\Pi,\mathbb{B}}$ game, we build a reduction adversary $\mathcal{R}$ in the fh-IND game such that:

- $\mathcal{R}$ first samples $\mathcal{B}^{(0)}$ and $\mathcal{B}^{(1)}$.
- $\mathcal{R}$ then calls $\mathbf{c_x} \leftarrow \mathcal{O}_{\text{KeyGen}}(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$, where $\mathbf{x}^{(0)}$ and $\mathbf{x}^{(1)}$ are created from $\mathcal{B}^{(0)}$ and $\mathcal{B}^{(1)}$.
- $\mathcal{R}$ prepares $\mathbf{y}^{(0)}$ and $\mathbf{y}^{(1)}$ from $\mathcal{B}^{(0)}$ and $\mathcal{B}^{(1)}$ in a way that, $\mathbf{x}^{(0)}{\mathbf{y}^{(0)}}^T = \mathbf{x}^{(1)}{\mathbf{y}^{(1)}}^T$, and calls $\mathbf{c_y} \leftarrow \mathcal{O}_{\text{Enc}}(\mathbf{y}^{(0)}, \mathbf{y}^{(1)})$.
- By Assumption 1, $\mathbf{y}^{(0)}$ and $\mathbf{y}^{(1)}$ still follow the correct distribution.

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

$\mathcal{B}^{(0)} \leftarrow_{\$} \mathbb{B}, \quad \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}^{(0)}$
$\mathcal{B}^{(1)} \leftarrow_{\$} \mathbb{B}, \quad \mathbb{B} \leftarrow \mathbb{B} \setminus \mathcal{B}^{(1)}$
$\mathbf{b}^{(0)} \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}^{(0)}}}(), \mathbf{x}^{(0)} \leftarrow (b_1^{(0)}, \cdots, b_k^{(0)}, 1, \|\mathbf{b}^{(0)}\|^2)$
$\mathbf{b}^{(1)} \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}^{(1)}}}(), \mathbf{x}^{(1)} \leftarrow (b_1^{(1)}, \cdots, b_k^{(1)}, 1, \|\mathbf{b}^{(1)}\|^2)$
$\mathbf{c}_{\mathbf{x}} \leftarrow \mathcal{O}_{\mathsf{KeyGen}}(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$
for $i = 1$ to t **do**
    $\mathbf{b}'^{(0)} \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}^{(0)}}}()$
    $\mathbf{y}^{(0)} \leftarrow (-2{b'_1}^{(0)}, \cdots, -2{b'_k}^{(0)}, \|\mathbf{b}'^{(0)}\|^2, 1)$
    repeat
        $\mathbf{b}'^{(1)} \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}^{(1)}}}()$
        $\mathbf{y}^{(1)} \leftarrow (-2{b'_1}^{(1)}, \cdots, -2{b'_k}^{(1)}, \|\mathbf{b}'^{(1)}\|^2, 1)$
    until $\mathbf{x}^{(0)}{\mathbf{y}^{(0)}}^T = \mathbf{x}^{(1)}{\mathbf{y}^{(1)}}^T$
    ${\mathbf{c}_{\mathbf{y}}}^{(i)} \leftarrow \mathcal{O}_{\mathsf{Enc}}(\mathbf{y}^{(0)}, \mathbf{y}^{(1)})$
$\tilde{b} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{B}^{(0)}},\mathcal{O}_{\mathcal{B}^{(1)}}}(\mathsf{pp}, \mathbf{c}_{\mathbf{x}}, \{{\mathbf{c}_{\mathbf{y}}}^{(i)}\}_{i=1}^t)$
return $\tilde{b}$

Table of Contents

1. Introduction
2. Formalization
3. Security Models
4. Security Analysis
5. Conclusion

Conclusion

In this project,

Conclusion

In this project,

- We provide a framework of a biometric authentication scheme with a cryptographic layer to preserve privacy.

Conclusion

In this project,

- We provide a framework of a biometric authentication scheme with a cryptographic layer to preserve privacy.
- We define $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game and $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game.

Conclusion

In this project,

- We provide a framework of a biometric authentication scheme with a cryptographic layer to preserve privacy.
- We define $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game and $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game.
  - $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game models the ability of an adversary to impersonate the user.

Conclusion

In this project,

- We provide a framework of a biometric authentication scheme with a cryptographic layer to preserve privacy.
- We define $\mathsf{UF}_{\Pi,\mathbb{B},\text{option}}$ game and $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game.
  - $\mathsf{UF}_{\Pi,\mathbb{B},\text{option}}$ game models the ability of an adversary to impersonate the user.
  - $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game models the ability of the server to recognize the user.

Conclusion

In this project,

- We provide a framework of a biometric authentication scheme with a cryptographic layer to preserve privacy.
- We define $\mathsf{UF}_{\Pi,\mathbb{B},\text{option}}$ game and $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game.
  - $\mathsf{UF}_{\Pi,\mathbb{B},\text{option}}$ game models the ability of an adversary to impersonate the user.
  - $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game models the ability of the server to recognize the user.
- We introduce two instantiations using fh-IPFE and relational hash.

Conclusion

In this project,

- We provide a framework of a biometric authentication scheme with a cryptographic layer to preserve privacy.
- We define $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game and $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game.
  - $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game models the ability of an adversary to impersonate the user.
  - $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game models the ability of the server to recognize the user.
- We introduce two instantiations using fh-IPFE and relational hash.
  - In the presentation, we focus on fh-IPFE-based instantiation.

Conclusion

In this project,

- We provide a framework of a biometric authentication scheme with a cryptographic layer to preserve privacy.
- We define $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game and $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game.
  - $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game models the ability of an adversary to impersonate the user.
  - $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game models the ability of the server to recognize the user.
- We introduce two instantiations using fh-IPFE and relational hash.
  - In the presentation, we focus on fh-IPFE-based instantiation.
  - We introduce fh-IND and RUF security of fh-IPFE.

Conclusion

In this project,

- We provide a framework of a biometric authentication scheme with a cryptographic layer to preserve privacy.
- We define $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game and $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game.
  - $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game models the ability of an adversary to impersonate the user.
  - $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game models the ability of the server to recognize the user.
- We introduce two instantiations using fh-IPFE and relational hash.
  - In the presentation, we focus on fh-IPFE-based instantiation.
  - We introduce fh-IND and RUF security of fh-IPFE.
  - We provide two theorems to achieve RUF security.

Conclusion

In this project,

- We provide a framework of a biometric authentication scheme with a cryptographic layer to preserve privacy.
- We define $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game and $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game.
  - $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game models the ability of an adversary to impersonate the user.
  - $\mathsf{IND}_{\Pi,\mathbb{B}}(\mathcal{A})$ game models the ability of the server to recognize the user.
- We introduce two instantiations using fh-IPFE and relational hash.
  - In the presentation, we focus on fh-IPFE-based instantiation.
  - We introduce fh-IND and RUF security of fh-IPFE.
  - We provide two theorems to achieve RUF security.
- We show that how an fh-IPFE-based instantiation can be $\{\mathbf{c}_{\mathbf{x}}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Enroll}}\}$-UF, $\{\mathbf{c}_{\mathbf{x}}, \mathsf{csk}, \mathcal{O}_{\mathcal{B}}, \mathcal{O}_{\mathsf{Probe}}\}$-UF, and IND.

Discussion and Future Work

- We can consider other two oracles in option in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game:

Discussion and Future Work

- We can consider other two oracles in option in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game:
  - $\mathcal{O}'_{\mathsf{Enroll}}(\cdot)$: On input esk', first sample $\mathbf{b}' \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ and output $\mathsf{Enroll}(\mathsf{esk}', \mathbf{b}')$.
  - $\mathcal{O}'_{\mathsf{Probe}}(\cdot)$: On input psk', first sample $\mathbf{b}' \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$ and output $\mathsf{Probe}(\mathsf{psk}', \mathbf{b}')$.

Discussion and Future Work

- We can consider other two oracles in option in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game:
  - $\mathcal{O}'_{\mathsf{Enroll}}(\cdot)$: On input esk', first sample $\mathbf{b}' \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ and output $\mathsf{Enroll}(\mathsf{esk}', \mathbf{b}')$.
  - $\mathcal{O}'_{\mathsf{Probe}}(\cdot)$: On input psk', first sample $\mathbf{b}' \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$ and output $\mathsf{Probe}(\mathsf{psk}', \mathbf{b}')$.

  This models a scenario when an adversary can set illegal keys to do bad things.

Discussion and Future Work

- We can consider other two oracles in option in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game:
  - $\mathcal{O}'_{\mathsf{Enroll}}(\cdot)$: On input esk', first sample $\mathbf{b}' \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ and output $\mathsf{Enroll}(\mathsf{esk}', \mathbf{b}')$.
  - $\mathcal{O}'_{\mathsf{Probe}}(\cdot)$: On input psk', first sample $\mathbf{b}' \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$ and output $\mathsf{Probe}(\mathsf{psk}', \mathbf{b}')$.

  This models a scenario when an adversary can set illegal keys to do bad things.
- Analyses of other instantiations of a biometric authentication scheme.

Discussion and Future Work

- We can consider other two oracles in option in the $\mathsf{UF}_{\Pi,\mathbb{B},\mathsf{option}}$ game:
  - $\mathcal{O}'_{\mathsf{Enroll}}(\cdot)$: On input esk', first sample $\mathbf{b}' \leftarrow \mathsf{getEnroll}^{\mathcal{O}_{\mathcal{B}}}()$ and output $\mathsf{Enroll}(\mathsf{esk}', \mathbf{b}')$.
  - $\mathcal{O}'_{\mathsf{Probe}}(\cdot)$: On input psk', first sample $\mathbf{b}' \leftarrow \mathsf{getProbe}^{\mathcal{O}_{\mathcal{B}}}()$ and output $\mathsf{Probe}(\mathsf{psk}', \mathbf{b}')$.

  This models a scenario when an adversary can set illegal keys to do bad things.
- Analyses of other instantiations of a biometric authentication scheme.
  - Two-input Inner Product Functional Encryption.
  - Fuzzy Extractor.
  - Homomorphic Encryption.

Discussion and Future Work

- We can consider other two oracles in option in the $\mathsf{UF}_{\Pi,\mathbb{B},\text{option}}$ game:
  - $\mathcal{O}'_{\text{Enroll}}(\cdot)$: On input esk', first sample $\mathbf{b}' \leftarrow \mathsf{getEnroll}^{\mathcal{O}_\mathcal{B}}()$ and output $\mathsf{Enroll}(\mathsf{esk}', \mathbf{b}')$.
  - $\mathcal{O}'_{\text{Probe}}(\cdot)$: On input psk', first sample $\mathbf{b}' \leftarrow \mathsf{getProbe}^{\mathcal{O}_\mathcal{B}}()$ and output $\mathsf{Probe}(\mathsf{psk}', \mathbf{b}')$.

  This models a scenario when an adversary can set illegal keys to do bad things.
- Analyses of other instantiations of a biometric authentication scheme.
  - Two-input Inner Product Functional Encryption.
  - Fuzzy Extractor.
  - Homomorphic Encryption.

  Some of them have different structures from our framework, such as a challenge-based protocol.

Reference I

[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.

[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.

Table of Contents

6 Appendix - Achievability of RUF Security

fh-IND almost implies $\mathcal{O}'_{\mathsf{KeyGen}}$-RUF

Theorem

If FE is fh-IND, and if the RUF adversary can only return $\tilde{\mathbf{z}}$ that is an encryption of a nonzero vector, then FE is $\mathcal{O}'_{KeyGen}$-RUF for any $\gamma \leq \|\mathbb{F}\|$.

Given a $\mathsf{RUF}^{\mathcal{O}'_{\mathsf{KeyGen}},\gamma}$ adversary $\mathcal{A}$, consider the following fh-IND adversary:

1. Run $\mathbf{c} \leftarrow \mathcal{O}_{\mathsf{KeyGen}}(\mathbf{r}^{(0)}, \mathbf{r}^{(1)})$, where $\mathbf{r}^{(0)}, \mathbf{r}^{(1)} \leftarrow_{\$} \mathbb{F}^k$.
2. Run $\tilde{\mathbf{z}} \leftarrow \mathbf{A}^{\mathcal{O}'_{\mathsf{KeyGen}}}(\mathsf{pp}, \mathbf{c})$.
3. Let $\tilde{\mathbf{z}}$ be encryption of $\mathbf{v} \neq \mathbf{0}$.
4. Run $\mathbf{c}_i \leftarrow \mathcal{O}_{\mathsf{KeyGen}}(\mathbf{r}^{(0)}, \mathbf{r}_i)$, where $\mathbf{r}_i \leftarrow_{\$} \mathbb{F}^k$.
5. If $\mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c}_i, \tilde{\mathbf{z}}) \leq \gamma$ for all i, return $\tilde{b} = 0$. Otherwise, return $\tilde{b} \leftarrow_{\$} \{0, 1\}$.
6. If $b = 0$ and $\mathcal{A}$ wins, $\mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c}_i, \tilde{\mathbf{z}}) \leq \gamma$ for all i. Otherwise, $\mathsf{FE.Dec}(\mathsf{pp}, \mathbf{c}_i, \tilde{\mathbf{z}}) \leq \gamma$ is a random number in $\{0, 1, \cdots, q-1\}$.

Upgrading FE to RUF FE′

Theorem

Given an sEUF-CMA digital signature scheme Sig and any fh-IPFE FE, we can obtain an fh-IPFE FE′ that is RUF for any γ.

- $\mathsf{FE}'.\mathsf{Setup}(1^\lambda)$: Run $\mathsf{FE}.\mathsf{Setup}(1^\lambda) \to (\mathsf{msk}, \mathsf{pp})$ and $\mathsf{Sig}.\mathsf{KeyGen}(1^\lambda) \to (\mathsf{sk}_{\mathsf{Sig}}, \mathsf{pk}_{\mathsf{Sig}})$. Output $\mathsf{msk}' = (\mathsf{msk}, \mathsf{sk}_{\mathsf{Sig}})$ and $\mathsf{pp}' = (\mathsf{pp}, \mathsf{pk}_{\mathsf{Sig}})$.

Upgrading FE to RUF FE$'$

Theorem

Given an sEUF-CMA digital signature scheme Sig and any fh-IPFE FE, we can obtain an fh-IPFE FE$'$ that is RUF for any γ.

- $\mathsf{FE}'.\mathsf{Setup}(1^\lambda)$: Run $\mathsf{FE}.\mathsf{Setup}(1^\lambda) \to (\mathsf{msk}, \mathsf{pp})$ and $\mathsf{Sig}.\mathsf{KeyGen}(1^\lambda) \to (\mathsf{sk}_{\mathsf{Sig}}, \mathsf{pk}_{\mathsf{Sig}})$. Output $\mathsf{msk}' = (\mathsf{msk}, \mathsf{sk}_{\mathsf{Sig}})$ and $\mathsf{pp}' = (\mathsf{pp}, \mathsf{pk}_{\mathsf{Sig}})$.
- $\mathsf{FE}'.\mathsf{KeyGen}(\mathsf{msk}', \mathbf{x})$: Run and output $f_{\mathbf{x}} \leftarrow \mathsf{FE}.\mathsf{Enc}(\mathsf{msk}, \mathbf{x})$.

Upgrading FE to RUF FE$'$

Theorem

Given an sEUF-CMA digital signature scheme Sig and any fh-IPFE FE, we can obtain an fh-IPFE FE$'$ that is RUF for any γ.

- $\mathsf{FE}'.\mathsf{Setup}(1^\lambda)$: Run $\mathsf{FE}.\mathsf{Setup}(1^\lambda) \to (\mathsf{msk}, \mathsf{pp})$ and $\mathsf{Sig}.\mathsf{KeyGen}(1^\lambda) \to (\mathsf{sk}_{\mathsf{Sig}}, \mathsf{pk}_{\mathsf{Sig}})$. Output $\mathsf{msk}' = (\mathsf{msk}, \mathsf{sk}_{\mathsf{Sig}})$ and $\mathsf{pp}' = (\mathsf{pp}, \mathsf{pk}_{\mathsf{Sig}})$.
- $\mathsf{FE}'.\mathsf{KeyGen}(\mathsf{msk}', \mathbf{x})$: Run and output $f_{\mathbf{x}} \leftarrow \mathsf{FE}.\mathsf{Enc}(\mathsf{msk}, \mathbf{x})$.
- $\mathsf{FE}'.\mathsf{Enc}(\mathsf{msk}', \mathbf{y})$: Run $\mathsf{FE}.\mathsf{Enc}(\mathsf{msk}, \mathbf{y}) \to \mathbf{c_y}$ and sign $\mathbf{c_y}$ by $\mathsf{Sig}.\mathsf{Sign}(\mathsf{sk}_{\mathsf{Sig}}, \mathbf{c_y}) \to \sigma$. Output $\mathbf{c_y}' = (\mathbf{c_y}, \sigma)$.

Upgrading FE to RUF FE′

Theorem

Given an sEUF-CMA digital signature scheme Sig and any fh-IPFE FE, we can obtain an fh-IPFE FE′ that is RUF for any γ.

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

Upgrading FE to RUF FE′

Theorem

Given an sEUF-CMA digital signature scheme Sig and any fh-IPFE FE, we can obtain an fh-IPFE FE′ that is RUF for any γ.

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

If an adverary can find $\tilde{\mathbf{z}}$ such that $\mathsf{FE}'.\mathsf{Dec}(\mathsf{pp}', \mathbf{c}, \tilde{\mathbf{z}}) \neq \perp$, it can forge a signature.