Witness Encryptionによる適応的安全なMatchmaking Encryptionの構成

松下 大晟[1,a)]　知久 奏斗[1,2,3,b)]　原 啓祐[3,2,c)]　四方 順司[4,2,d)]

概要：Matchmaking Encryption(ME) とは，送受信者間で相手のポリシーを指定できる暗号化方式である．ME では，送信者が受信者の満たすべき属性を指定でき，また受信者も送信者がもつべき属性を指定でき，これらが一致した場合のみ受信者は暗号文を復号可能となる．ME はこれまで，Predicarte Encryption と Lockable Obfuscation を用いた選択的安全な構成が知られていたが，適応的安全な構成はまだ Indistinguishable Obfuscation(IO) によるものしか知られていない．そこで，本研究では (IO より弱い仮定であると信じられている)Witness Encryption を使うことで，適応的安全な ME を構成する．

Adaptive Matchmaking Encryption from Witness Encryption

Taisei Matsushita[1,a)]　Sohto Chiku[1,2,3,b)]　Keisuke Hara[3,2,c)]　Junji Shikata[4,2,d)]

Abstract: Matchmaking encryption (ME) is an advanced cryptographic primitive that allows senders and receivers to specify each other's policies. In ME, the sender can specify the attributes that the receiver must satisfy, and the receiver can specify the attributes that the sender must have. Only when these attributes match can the receiver decrypt the ciphertext. ME has previously been known to have a selectively secure construction using predicate encryption and lockable obfuscation, but an adaptively secure construction is only known to exist using indistinguishable obfuscation (IO). In this work, we propose an adaptively secure ME scheme using witness encryption (which is believed to be weaker than IO).

1. はじめに

Matchmaking Encryption(ME) とは，Ateniese ら [1] によって提案された，送受信者間で相手のポリシーを指定できる暗号化方式である．ME では，送信者が受信者の満たすべき属性を指定できると同時に，受信者も送信者がもつべき属性を指定でき，これらが一致した場合のみ受信者は暗号文を復号できる．分かりやすさのために，属性 σ，ポリシー $\mathbb{R}$ をもつ送信者と，属性 ρ，ポリシー $\mathbb{S}$ をもつ受信者が居た場合を考える．このとき ME では，受信者属性が送信者のポリシーを満たす（$\mathbb{R}(\rho) = 1$）のと同時に，送信者属性が受信者のポリシーを満たす（$\mathbb{S}(\sigma) = 1$）場合のみ，受信者は暗号文を復号可能となる．ME に求める安全性には，Privacy と Authenticity の２要件が存在する．まず Privacy とは，送信した暗号文から送信者に関する情報が漏れないことに関する安全性であり，一方 Authenticity とは，暗号文の偽造が行えないことに関する安全性である．特に Privacy については，攻撃者がクエリできる鍵の種類に応じて Mismatch Condition と Match Condition の２種類が存在する．ME は比較的新しい暗号技術であるが，いくつかの先行研究が存在する．まず Ateniese ら [1]

1 横浜国立大学 大学院環境情報学府
Graduate School of Environment and Information Sciences, Yokohama National University
2 横浜国立大学 先端科学高等研究院
Institute of Advanced Science, Yokohama National University
3 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology (AIST)
4 横浜国立大学 大学院環境情報研究院
Faculty of Environment and Information Sciences, Yokohama National University
a) matsushita-taisei-bk@ynu.jp
b) chiku-sohto-tw@ynu.jp
c) hara-keisuke@aist.go.jp
d) shikata-junji-rb@ynu.ac.jp

は ME の概念を導入し，ME の安全性の定義，確率的関数に対する関数型暗号を用いた ME の構成，2 入力関数型暗号を用いた ME の構成を示した．これらの構成は，Match Condition と Mismatch Condition の両方において適応的 Privacy を満たすが，識別不能難読化 (iO) を仮定する．また，効率的に実装可能な ME の特殊ケースとして，Bilinear Diffie-Helman 仮定に基づく ID ベース ME の構成の提案と実装評価 [2] が行われた．次に，Francati ら [4] は 2 鍵 Predicate Encryption (PE) からの ME の構成を示した．特に，PE の CPA-1-sided 安全性と CPA-2-sided 安全性をそれぞれ，Mismatch Condition と Match Condition に反映している．また，Mismatch Condition と Match Condition の両方において Privacy を満たすために必要な CPA-2-sided 安全性を満たす PE については iO を包含することが知られており，Mismatch Condition のみの適応的 Privacy を満たす構成は Complexity Leveraging が必要となる．以上のように，適応的安全性を満たす方式は iO や Complexity Leveraging といった強力な仮定に依存しており，制約が大きい．

1.1 貢献

本稿では，iO や Complexity Leveraging に依存することなく，Mismatch Case において適応的 Privacy を満たす ME を構成する．具体的には，WE [5]，コミットメント，NIZK，関数タグシステム [9]，Lockable Obfuscation(LO) [6, 10] を用いて ME を構成する．構成の主なアイデアは，Waters と Wichs の WE から適応的安全な属性ベース暗号を構成した手法 [9] を応用している．次に構成した ME について，Mismatch Condition の適応的 Privacy を証明した．

2. 準備

ここでは，本稿において用いる記法と暗号プリミティブを導入する．

表記． 任意の文字列 $x \in \{0,1\}^*$ に対し，$|x|$ をその長さとする．$\mathcal{X}$ を集合とする．x が $\mathcal{X}$ から一様ランダムに選ばれることを，$x \leftarrow\!\!\$\ \mathcal{X}$ と書く．x を入力として A を動作し y を出力することを，$y \leftarrow \mathsf{A}(x)$ と書く．もし A が確率的アルゴリズムであれば，y はランダムな値となり，$\mathsf{A}(x;\ r)$ は入力 x と乱数 r に対して A を動作することを表す．また，A が確率的アルゴリズムでかつ全ての入力 $x,\ r \in \{0,1\}^*$ に対し $\mathsf{A}(x;r)$ の計算が（入力サイズに対し）多項式ステップで完了することを，A が確率的多項式時間 (PPT) アルゴリズムであるという．本論文を通して，我々はセキュリティパラメータを $\lambda \in \mathbb{N}$ で表し，全てのアルゴリズムが λ を暗黙的に入力とすると仮定する．関数 $v:\ \mathbb{N} \to [0,1]$ がセキュリティパラメータ λ に対し negligible であるとは，関数 $v(\lambda)$ が λ の任意多項式の逆数よりも早く動作しない，すなわち全ての正の多項式 $p(\lambda)$ に対し $v(\lambda) \in O(\frac{1}{p(\lambda)})$ であることを言う．また，セキュリティパラメータに関し特定されていない negligible 関数を $\mathsf{negl}(\lambda)$ と表記する．

2.1 マッチメイキング暗号

本章では，マッチメイキング暗号 (ME) [1, 2] を導入する．本稿では，Authentication を考えないため，[1] で導入されている送信者の属性に対する鍵を考えないことから，送信者の属性に関する鍵生成 SKGen は省略するが，NIZK を用いることによって Authenticity を達成することは容易である．

定義 1 (マッチメイキング暗号)．*ME* 方式は以下の PPT アルゴリズムの組 $\mathsf{ME} = (\mathsf{Setup}, \mathsf{RKGen}, \mathsf{PolGen}, \mathsf{Enc}, \mathsf{Dec})$ からなる*:*

- $\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$*:* セットアップアルゴリズムはセキュリティパラメータ 1^λ を入力として，マスター公開鍵，マスター秘密鍵の組 $(\mathsf{mpk}, \mathsf{msk})$ を出力する．
- $\mathsf{RKGen}(\mathsf{msk}, \rho) \to \mathsf{dk}_\rho$*:* 受信者鍵生成アルゴリズムはマスター秘密鍵 msk と属性 ρ を入力として，属性 ρ に関する復号鍵 dk_ρ を出力する．
- $\mathsf{PolGen}(\mathsf{msk}, \mathbb{S}) \to \mathsf{dk}_\mathbb{S}$*:* ポリシー生成アルゴリズムはマスター秘密鍵 msk とポリシー $\mathbb{S}$ を入力として，ポリシー $\mathbb{S}$ に関する復号鍵 $\mathsf{dk}_\mathbb{S}$ を出力する．
- $\mathsf{Enc}(\mathsf{mpk}, \sigma, \mathbb{R}, \mathsf{m}) \to \mathsf{ct}$*:* 暗号化アルゴリズムはマスター公開鍵 mpk と属性 σ，ポリシー $\mathbb{R}$，平文 m を入力として，暗号文 ct を出力する．
- $\mathsf{Dec}(\mathsf{mpk}, \mathsf{dk}_\rho, \mathsf{dk}_\mathbb{S}, \mathsf{ct}) \to \mathsf{m}/\perp$*:* 復号アルゴリズムはマスター公開鍵 mpk と属性 ρ に関する復号鍵 dk_ρ，ポリシー $\mathbb{S}$ に関する復号鍵 $\mathsf{dk}_\mathbb{S}$，暗号文 ct を入力として，平文 m もしくは復号失敗を意味する $\perp$ を出力する．

マッチメイキング暗号に対して正当性と安全性要件を定義する．

正当性: マッチメイキング暗号方式 ME が正当性を満たすとは，任意の $\lambda \in \mathbb{N}$，$\mathbb{R}(\rho) = 1$ であるような $\mathbb{R}$ と ρ，$\mathbb{S}(\sigma) = 1$ であるような $\mathbb{S}$ と σ，任意の m に対して以下が成り立つことである*:*

$$\Pr[\mathsf{Dec}(\mathsf{mpk}, \mathsf{dk}_\rho, \mathsf{dk}_\mathbb{S}, \mathsf{ct}) = \mathsf{m}] = 1,$$

ここで，$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$，$\mathsf{dk}_\rho \leftarrow \mathsf{RKGen}(\mathsf{msk}, \rho)$，$\mathsf{dk}_\mathbb{S} \leftarrow \mathsf{PolGen}(\mathsf{msk}, \mathbb{S})$，$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, \sigma, \mathbb{R}, \mathsf{m})$ である．

Privacy: マッチメイキング暗号に対して以下のような攻撃者 $\mathcal{A}$ と挑戦者間の安全性ゲーム $\mathsf{Expt}^{\mathsf{privacy}}_{\mathcal{A},\mathsf{ME}}(\lambda)$ を考える*:*

セットアップフェーズ: 挑戦者はチャレンジコイン $b \leftarrow\!\!\$\ \{0,1\}$ を選び，$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$ を実行し，mpk

を攻撃者 $\mathcal{A}$ に渡す．

クエリフェーズ: $\mathcal{A}$ が各オラクルクエリを行うと挑戦者は以下のように返答する:

- 復号鍵生成オラクル: $\mathcal{A}$ は多項式回の復号鍵生成クエリを行う．$\mathcal{A}$ は ρ を選択する．挑戦者は $\mathsf{dk}_\rho \leftarrow \mathsf{RKGen}(\mathsf{msk}, \rho)$ を実行し dk_ρ を $\mathcal{A}$ に返す．
- ポリシー生成オラクル: $\mathcal{A}$ は多項式回のポリシー生成クエリを行う．$\mathcal{A}$ は $\mathbb{S}$ を選択する．挑戦者は $\mathsf{dk}_\mathbb{S} \leftarrow \mathsf{PolGen}(\mathsf{msk}, \mathbb{S})$ を実行し $\mathsf{dk}_\mathbb{S}$ を $\mathcal{A}$ に返す．
- 暗号化オラクル: $\mathcal{A}$ は *1* 回のみ暗号化クエリを行う．$\mathcal{A}$ は $(\sigma_0, \sigma_1, \mathbb{R}_0, \mathbb{R}_1, \mathsf{m}_0, \mathsf{m}_1)$ を選択する．挑戦者は $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, \sigma_b, \mathbb{R}_b, \mathsf{m}_b)$ を実行し ct を $\mathcal{A}$ に返す．

推測フェーズ: 最後に $\mathcal{A}$ が推測したコイン $b' \in \{0,1\}$ を出力する．$b = b'$ ならば，挑戦者は 1 を出力する．そうでなければ，挑戦者は 0 を出力する．

いま，攻撃者 $\mathcal{A}$ が有効な攻撃者であるとは，すべてのクエリを行った $\rho, \mathbb{S}$ に対して以下が成り立つときである:

(1) Mismatch Condition: Either

$$(\mathbb{R}_0(\rho) = \mathbb{R}_1(\rho) = 0) \vee (\mathbb{S}(\sigma_0) = \mathbb{S}(\sigma_1) = 0) \vee$$
$$(\mathbb{R}_0(\rho) = \mathbb{S}(\sigma_1) = 0) \vee (\mathbb{S}(\sigma_0) = \mathbb{R}_1(\rho) = 0)$$

(2) Match Condition: Or

$$(\mathsf{m}_0 = \mathsf{m}_1) \wedge (\mathbb{R}_0(\rho) = \mathbb{R}_1(\rho)) \wedge (\mathbb{S}(\sigma_0) = \mathbb{S}(\sigma_1))$$

攻撃者の優位性 $\mathsf{Adv}^{\mathsf{privacy}}_{\mathcal{A},\mathsf{ME}}(\lambda)$ を

$$\mathsf{Adv}^{\mathsf{privacy}}_{\mathcal{A},\mathsf{ME}}(\lambda) := \left| \Pr\left[\mathsf{Expt}^{\mathsf{privacy}}_{\mathcal{A},\mathsf{ME}}(\lambda) \Rightarrow 1\right] - \frac{1}{2} \right|$$

と定義する．マッチメイキング暗号方式 ME が *privacy* を満たすとは，すべての有効な PPT 攻撃者 $\mathcal{A}$ に対して，$\mathsf{Adv}^{\mathsf{privacy}}_{\mathcal{A},\mathsf{ME}}(\lambda) = \mathsf{negl}(\lambda)$ となる無視できる関数 $\mathsf{negl}(\cdot)$ が存在することである．

2.2 コミットメント

本章では，statistically binding なコミットメントを，共通参照文字列（CRS）モデルで定義する [8]．

定義 2 (コミットメント)．コミットメントスキームは以下の PPT アルゴリズムの組 $\mathsf{COM} = (\mathsf{Setup}, \mathsf{Commit})$ からなる：

- $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$：共通参照文字列 crs を生成する．
- $\mathsf{com} := \mathsf{Commit}_{\mathsf{crs}}(x; r)$：乱数 $r \in \{0,1\}^\lambda$ を使い，文字列 $x \in \{0,1\}^n$ に対するコミットメント com を生成する．

コミットメントに対し，以下の 2 性質を要求する：

Hiding: $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda), \mathsf{com}_b \leftarrow \mathsf{Commit}_{\mathsf{crs}}(b)$ に対して $(\mathsf{crs}, \mathsf{com}_0) \stackrel{c}{\approx} (\mathsf{crs}, \mathsf{com}_1)$ が成立する．

Statistical Binding: crs が binding であるとは，$\mathsf{Commit}_{\mathsf{crs}}(x_0; r_0) = \mathsf{Commit}_{\mathsf{crs}}(x_1; r_1)$ となるような乱数 r_0, r_1 が存在しないことである．つまり，以下が成立する：

$$\Pr[\mathsf{crs}\,\text{is binding} : \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)] = 1 - \mathsf{negl}(\lambda).$$

一方向性関数を仮定すれば，statistically binding なコミットメントが存在する [8]．

2.3 非対話ゼロ知識証明（NIZK）

本章では statistically sound なゼロ知識証明（NIZK）を，witness indistinguishability をもつ CRS モデルにより定義する [3, 7]．

定義 3 (非対話ゼロ知識証明（NIZK）)．NP 言語 $L_\lambda = \{x : \exists w \ \ (x, w) \in R_\lambda\}$ における NP 関係 $R_\lambda \subseteq \{0,1\}^{n(\lambda)} \times \{0,1\}^{m(\lambda)}$ に対する NIZK 証明は，以下の PPT アルゴリズムの組 $\mathsf{NIZK} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ で構成される．

- $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$：共通参照文字列 crs を生成する．
- $\pi \leftarrow \mathsf{Prove}_{\mathsf{crs}}(x, w)$：witness w を使い，ステートメント x に対する証明 π を生成する．
- $b \leftarrow \mathsf{Verify}_{\mathsf{crs}}(x, \pi)$：与えられたステートメント x に対する証明 π を検証し，決定ビット $b \in \{0,1\}$ を出力する．

NIZK 証明に対し，以下の 3 性質を要求する：

Completeness: 全ての $\lambda \in \mathbb{N}$, $(x, w) \in R_\lambda$ に対し以下を満たす $\mathsf{negl}(\cdot)$ が存在する：

$$\Pr\left[\mathsf{Verify}_{\mathsf{crs}}(x, w) = 1 \;\middle|\; \begin{array}{l} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda) \\ \pi \leftarrow \mathsf{Prove}_{\mathsf{crs}}(x, w) \end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

Statistical Soundness: 全ての $x \notin L_\lambda, \pi$ に対し $\mathsf{Verify}_{\mathsf{crs}}(x, \pi) = 0$ であるとき，crs は健全であり，以下が成立しなければならない：

$$\Pr\left[\mathsf{crs}\ \textit{is sound} : \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)\right] = 1 - \mathsf{negl}(\lambda).$$

Witness Indistinguishability: $b \in \{0,1\}$ において $(x_\lambda, w^b_\lambda) \in R_\lambda$ であるような任意の $x_\lambda.w^0_\lambda.w^1_\lambda$ に対し，$\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$, $\pi_b \leftarrow \mathsf{Prove}_{\mathsf{crs}}(x_\lambda.w^b_\lambda)$ *for* $b \in \{0,1\}$ の下で $(\mathsf{crs}, \pi_0) \approx_c (\mathsf{crs}, \pi_1)$ が成立する．

2.4 関数タグシステム

本章では，関数タグシステムを導入する [9]．

定義 4 (関数タグシステム)．関数タグシステムは以下の PPT アルゴリズムの組 $\mathsf{FTS} = (\mathsf{DInputTag}, \mathsf{DFunctionTag}, \mathsf{SGen}, \mathsf{SInputTag}, \mathsf{SFunctionTag},$

Trigger) からなる:

- $\mathsf{DInputTag}(x) \to \mathsf{tag}_x$: ダミー入力タグ生成アルゴリズムは入力 x を入力として，入力 x に関するタグ tag_x を出力する．
- $\mathsf{DFunctionTag}(f) \to \mathsf{tag}_f$: ダミー関数タグ生成アルゴリズムは関数 f を入力として，関数 f に関するタグ tag_f を出力する．
- $\mathsf{SGen}(1^\lambda) \to \mathsf{tsk}$: 鍵生成アルゴリズムはセキュリティパラメータ 1^λ を入力として，タグ生成鍵 tsk を出力する．
- $\mathsf{SInputTag}(\mathsf{tsk}, x) \to \mathsf{tag}_x$: スマート入力タグ生成アルゴリズムはタグ生成鍵 tsk と入力 x を入力として，入力 x に関するタグ tag_x を出力する．
- $\mathsf{SFunctionTag}(\mathsf{tsk}, f) \to \mathsf{tag}_f$: スマート関数タグ生成アルゴリズムはタグ生成鍵 tsk と関数 f を入力として，関数 f に関するタグ tag_f を出力する．
- $\mathsf{Trigger}(\mathsf{tag}_x, \mathsf{tag}_f) \to 1/0$: トリガーアルゴリズムは入力 x に関するタグ tag_x と関数 f に関するタグ tag_f を入力として，1 または 0 を出力する．

関数タグシステムに対して，正当性と安全性要件を定義する．

ダミー正当性: 関数タグシステム FTS がダミー正当性を満たすとは，任意の $\lambda \in \mathbb{N}$，f，x に対して，

$$\Pr[\mathsf{Trriger}(\mathsf{DInputTag}(x), \mathsf{DFunctionTag}(f)) = 1] = \mathsf{negl}(\lambda)$$

となる無視できる関数 $\mathsf{negl}(\cdot)$ が存在することである．

スマート正当性: 関数タグシステム FTS がスマート正当性を満たすとは，任意の $\lambda \in \mathbb{N}$, $f(x) = 1$ を満たすような f, x，任意の $\mathsf{tsk} \leftarrow \mathsf{SGen}(1^\lambda)$ に対して，以下が成り立つことである:

$$\Pr[\mathsf{Trriger}(\mathsf{SInputTag}(\mathsf{tsk}, x), \mathsf{SFunctionTag}(\mathsf{tsk}, f)) = 1] = 1.$$

安全性: 関数タグシステムに対して以下のような攻撃者 $\mathcal{A}$ と挑戦者間の安全性ゲーム $\mathsf{Expt}^{\mathsf{IND}}_{\mathcal{A},\mathsf{FTS}}(\lambda)$ を考える:

セットアップフェーズ: 挑戦者はチャレンジコイン $b \leftarrow\!\!\$ \{0,1\}$ を選び，$\mathsf{tsk} \leftarrow \mathsf{SGen}(1^\lambda)$ を実行する．

クエリフェーズ: $\mathcal{A}$ が各オラクルクエリを行うと挑戦者は以下のように返答する:

- 関数タグ生成オラクル: $\mathcal{A}$ は多項式回の関数タグ生成クエリを行う．$\mathcal{A}$ は f を選択する．挑戦者はコインに応じて，

$$\begin{cases} \mathsf{tag}_f \leftarrow \mathsf{DFunctionTag}(f) & \textbf{if } b = 0 \\ \mathsf{tag}_f \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f) & \textbf{if } b = 1 \end{cases}$$

を実行し，tag_f を $\mathcal{A}$ に返す．

- 入力タグ生成オラクル: $\mathcal{A}$ は 1 回のみ入力タグ生成クエリを行う．$\mathcal{A}$ は x を選択する．挑戦者はコインに応じて，

$$\begin{cases} \mathsf{tag}_x \leftarrow \mathsf{DFunctionTag}(x) & \textbf{if } b = 0 \\ \mathsf{tag}_x \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, x) & \textbf{if } b = 1 \end{cases}$$

を実行し，tag_x を $\mathcal{A}$ に返す．

推測フェーズ: 最後に $\mathcal{A}$ が推測したコイン $b' \in \{0,1\}$ を出力する．$b = b'$ ならば，挑戦者は 1 を出力する．そうでなければ，挑戦者は 0 を出力する．

いま，攻撃者 $\mathcal{A}$ が有効な攻撃者であるとは，すべてのクエリを行った f に対して $f(x) = 0$ が成り立つことである．攻撃者の優位性

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{A},\mathsf{FTS}}(\lambda) := \Big| \Pr\Big[\mathsf{Expt}^{\mathsf{IND}}_{\mathcal{A},\mathsf{FTS}}(\lambda) \Rightarrow 1 \;\Big|\; b = 0\Big] - \Pr\Big[\mathsf{Expt}^{\mathsf{IND}}_{\mathcal{A},\mathsf{FTS}}(\lambda) \Rightarrow 1 \;\Big|\; b = 1\Big] \Big|$$

と定義する．関数タグシステムが安全であるとは，すべての有効な PPT 攻撃者 $\mathcal{A}$ に対して $\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{A},\mathsf{FTS}}(\lambda) = \mathsf{negl}(\lambda)$ となる無視できる関数 $\mathsf{negl}(\cdot)$ が存在することである．

2.5 Witness Encryption

本章では，Witness Encryption を定義する [5].

定義 5 (Witness Encryption)**.** 対応する証言関係 R を有する NP 言語 L に対する *Witness Encryption* は以下の PPT アルゴリズムの組 $\mathsf{WE} = (\mathsf{Enc}, \mathsf{Dec})$ からなる:

- $\mathsf{Enc}(1^\lambda, x, \mu) \to \mathsf{ct}$: 暗号化アルゴリズムはセキュリティパラメータ 1^λ とインスタンス x，平文 μ を入力として，暗号文 ct を出力する．
- $\mathsf{Dec}(w, \mathsf{ct}) \to \mu$: 復号アルゴリズムは *witness* w と暗号文 ct を入力として，平文 μ を出力する．

Witness Encryption に対して，正当性と安全性要件を定義する．

正当性: *Witness Encryption* 方式 WE が正当性を満たすとは任意の $\lambda \in \mathbb{N}$，$R(x, w) = 1$ を満たすような w, x と任意の $\mu \in \{0,1\}$ に対して以下が成り立つことである:

$$\Pr\big[\mathsf{Dec}(w, \mathsf{Enc}(1^\lambda, x, \mu)) = \mu\big] = 1.$$

Soundness: *Witness Encryption* 方式 WE が *Soundness* を満たすとは，任意の $\lambda \in \mathbb{N}$，x，w，$\mu_0, \mu_1 \in \{0,1\}$ に対して以下が成り立つことである:

$$\mathsf{Enc}(1^\lambda, x, \mu_0) \overset{c}{\approx} \mathsf{Enc}(1^\lambda, x, \mu_1).$$

2.6 Lockable Obfuscation

本章では Lockable Obfuscation（LO）を定義する [6].

定義 6 (Lockable Obfuscation). 深さ $d(\lambda)$，$n(\lambda)$ ビット入力，$m(\lambda)$ ビット出力をもつ回路のクラス $\mathcal{C}_{n,m,d}$，平文空間 $\mathcal{M}$ に対する LO は，以下の PPT アルゴリズムの組 $\mathsf{LO} = (\mathsf{Obf}, \mathsf{Eval})$ からなる：

- $\mathsf{Obf}(1^\lambda, P, \mathrm{msg}, \alpha) \to \tilde{P}$：難読化アルゴリズムは，セキュリティパラメータ λ，プログラム $P \in \mathcal{C}_{n,m,d}$，メッセージ $\mathrm{msg} \in \mathcal{M}$，ロック文字列 $\alpha \in \{0,1\}^{m(\lambda)}$ を入力とし，難読化されたプログラム $\tilde{P}$ を出力する．

- $\mathsf{Eval}(\tilde{P}, x) \to y \in \mathcal{M} \cup \{\perp\}$：評価アルゴリズムは，難読化されたプログラム $\tilde{P}$，文字列 $x \in \{0,1\}^{n(\lambda)}$ を入力とし，$y \in \mathcal{M} \cup \{\perp\}$ を出力する．

LO に対し，正当性と安全性要件を定義する．

正当性: LO が正当性を満たすとは，任意の $\lambda \in \mathbb{N}$，入力 $x \in \{0,1\}^{n(\lambda)}$，プログラム $P \in \mathcal{C}_{n,m,d}$，メッセージ $\mathrm{msg} \in \mathcal{M}$，$\alpha \in \{0,1\}^{m(\lambda)}$ に対し，以下が成立することである：

- If $P(x) = \alpha$:

$$\Pr\left[\mathsf{Eval}(\mathsf{Obf}(1^\lambda, P, \mathrm{msg}, \alpha), x) = \mathrm{msg}\right] = 1,$$

- If $P(x) \neq \alpha$:

$$\Pr\left[\mathsf{Eval}(\mathsf{Obf}(1^\lambda, P, \mathrm{msg}, \alpha), x) = \mathrm{msg}\right] \leq \mathrm{negl}(\lambda).$$

安全性: LO が安全性を満たすとは，全ての PPT 攻撃者 $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ に対し，ある PPT シミュレータ Sim が存在して，以下が成り立つことである：

$$\left| \Pr\left[\mathcal{A}_1(\tilde{P}_b, \mathrm{st}) = b \;\middle|\; \begin{array}{c} (P, \mathrm{msg}, \mathrm{st}) \leftarrow \mathcal{A}_0(1^\lambda) \\ b \leftarrow\$ \{0,1\},\ \alpha \leftarrow\$ \{0,1\}^{m(\lambda)} \\ \tilde{P}_0 \leftarrow \mathsf{Obf}(1^\lambda, P, \mathrm{msg}, \alpha) \\ \tilde{P}_1 \leftarrow \mathsf{Sim}(1^\lambda, 1^{|P|}, 1^{|\mathrm{msg}|}) \end{array} \right] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda).$$

3. マッチメイキング暗号の一般的構成法

本章では，WE に基づく適応的安全な ME の一般的構成法を示し，安全性証明を行う．主なアイデアは，Waters と Wichs が提案した関数タグシステムから適応的安全な ABE を構成する技術 [9] に基づく．また，暗号文中の属性を隠すために，Goyal ら [6] の LO を使用する手法も構成に取り入れた．

3.1 提案構成法

本章では，WE，コミットメント，NIZK，関数タグシステム，LO を用いて ME を構成する．関数クラス $\mathcal{F}_\lambda$ を $\mathcal{F}_\lambda \subseteq \{f : \{0,1\}^{n(\lambda)} \to \{0,1\}\}$ となる関数集合とし，関数 $f_\rho : \mathcal{F}_\lambda \to \{0,1\}$ を，ポリシー $\mathbb{R} \in \mathcal{F}_\lambda$ を入力とし，$\mathbb{R}(\rho) \in \{0,1\}$ を出力するような関数として定義する．また，ME の構成に以下の技術を用いる：

- Witness Encryption 方式 $\mathsf{WE} = (\mathsf{WE.Enc}, \mathsf{WE.Dec})$
- コミットメント方式 $\mathsf{COM} = (\mathsf{Setup}, \mathsf{Commit})$
- NIZK 証明システム $\mathsf{NIZK} = (\mathsf{Setup}, \mathsf{Prove}.\mathsf{Verify})$
- 関数タグシステム $\mathsf{FTS} = (\mathsf{DInputTag}, \mathsf{DFunctionTag}, \mathsf{SGen}, \mathsf{SInputTag}, \mathsf{SFunctionTag}, \mathsf{Trriger})$
- Lockable Obfuscation $\mathsf{LO} = (\mathsf{Obf}, \mathsf{Eval})$

我々の提案する ME 方式 ME は，以下のように構成される [*1]:

- $\mathsf{Setup}(1^\lambda)$: まず $\mathsf{NIZK.crs}_0 \leftarrow \mathsf{NIZK.Setup}(1^\lambda), \mathsf{NIZK.crs}_1, \leftarrow \mathsf{NIZK.Setup}(1^\lambda), \mathsf{Com.crs} \leftarrow \mathsf{Com.Setup}(1^\lambda)$ を実行する．また，$r_0, r_1 \leftarrow\$ \{0,1\}^{2\lambda}$ を選ぶ．次に，$\mathsf{com}_0 \leftarrow \mathsf{Commit}_{\mathsf{Com.crs}}(0; r_0), \mathsf{com}_1 \leftarrow \mathsf{Commit}_{\mathsf{Com.crs}}(0^{l(\lambda)}; r_1)$ を計算する．最後に，$\mathsf{mpk} := (\mathsf{Com.crs}, \mathsf{NIZK.crs}_0, \mathsf{NIZK.crs}_1, \mathsf{com}_0, \mathsf{com}_1), \mathsf{msk} := r_0$ を出力する．

- $\mathsf{RKGen}(\mathsf{msk}, \rho)$: $\rho \in \{0,1\}^{n(\lambda)}$ を受信者の属性とし，上記で定義した f_ρ に対し，関数タグを $\mathsf{tag}_{f_\rho} \leftarrow \mathsf{DFunctionTag}(1^\lambda, f_\rho)$ で生成する．次に，以下の NP 関係 $\mathrm{NIZK}.R^0$ を考える:

$$\mathrm{NIZK}.R^0 = \left\{ (x_0, w_0) : \begin{array}{l} x_0 = (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, \rho, \mathsf{tag}_{f_\rho}) \\ \text{either } w_0 = r_0 \\ \quad : \mathsf{com}_0 = \mathsf{Commit}_{\mathsf{Com.crs}}(0; r_0) \\ \text{or } w_0 = (\mathsf{tsk}, r_1, r_2) \\ \quad : \mathsf{com}_1 = \mathsf{Commit}_{\mathsf{Com.crs}}(\mathsf{tsk}; r_1) \\ \quad \wedge\ \mathsf{tag}_{f_\rho} = \mathsf{SFunctionTag}(\mathsf{tsk}, f_\rho; r_2) \end{array} \right\}.$$

ここで $x_0 := (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, \rho, \mathsf{tag}_{f_\rho}), w_0 := r_0$ をセットし，$\mathrm{NIZK}.R^0$ に対し $\pi_0 \leftarrow \mathsf{Prove}_{\mathsf{NIZK.crs}_0}(x_0, w_0)$ を計算する．最後に，$\mathsf{dk}_\rho := \left(\rho, \mathsf{tag}_{f_\rho}, \pi_0\right)$ を出力する．

- $\mathsf{PolGen}(\mathsf{msk}, \mathbb{S})$: $\mathbb{S} \in \mathcal{F}_\lambda$ を受信者のポリシーとし，これに対し関数タグを $\mathsf{tag}_\mathbb{S} \leftarrow \mathsf{DFunctionTag}(1^\lambda, \mathbb{S})$ で生成する．次に，以下の NP 関係 $\mathrm{NIZK}.R^1$ を考える：

[*1] 前述のように本来 ME の安全性として Authenticity を考える必要があるが，これについては NIZK とデジタル署名を用いて [1] の方法で自明に証明可能なので省略する．

$\mathrm{NIZK}.R^1 =$

$$\left\{ (x_1, w_1) : \begin{array}{l} x_1 = (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, \mathbb{S}, \mathsf{tag}_{\mathbb{S}}) \\ \text{either } w_1 = r_0 \\ \quad : \mathsf{com}_0 = \mathsf{Commit}_{\mathsf{Com.crs}}(0; r_0) \\ \text{or } w_1 = (\mathsf{tsk}, r_1, r_2) \\ \quad : \mathsf{com}_1 = \mathsf{Commit}_{\mathsf{Com.crs}}(\mathsf{tsk}; r_1) \\ \quad \wedge\, \mathsf{tag}_{\mathbb{S}} = \mathsf{SFunctionTag}(\mathsf{tsk}, \mathbb{S}; r_2) \end{array} \right\}.$$

ここで $x_1 := (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, \mathbb{S}, \mathsf{tag}_{\mathbb{S}}), w_1 := r_0$ をセットし，$\mathrm{NIZK}.R^1$ に対して $\pi_1 \leftarrow \mathsf{Prove}_{\mathsf{NIZK.crs}_1}(x_1, w_1)$ を計算する．最後に，$\mathsf{dk}_{\mathbb{S}} := (\mathbb{S}, \mathsf{tag}_{\mathbb{S}}, \pi_1)$ を出力する．

– $\mathsf{Enc}(\mathsf{mpk}, \sigma, \mathbb{R}, \mathsf{m})$：$\mathsf{tag}_\sigma \leftarrow \mathsf{DInputTag}(1^\lambda, \sigma), \mathsf{tag}_{\mathbb{R}} \leftarrow \mathsf{DInputTag}(1^\lambda, \mathbb{R})$ を実行する．次に，以下の NP 関係 $\mathrm{WE}.R$ を考える：

$\mathrm{WE}.R =$

$$\left\{ \begin{array}{l} (x_{WE}, w_{WE}) : \\ x_{WE} = (\mathsf{Com.crs}, \mathsf{NIZK.crs}_0, \mathsf{NIZK.crs}_1, \\ \qquad\qquad \mathsf{com}_0, \mathsf{com}_1, \sigma, \mathsf{tag}_\sigma, \mathbb{R}, \mathsf{tag}_{\mathbb{R}}) \\ w_{WE} = (\rho, \mathsf{tag}_{f_\rho}, \mathbb{S}, \mathsf{tag}_{\mathbb{S}}, \pi_0, \pi_1) : \\ \mathbb{R}(\rho) = 1 \wedge \mathsf{Verify}_{\mathsf{NIZK.crs}_0}(x_0, \pi_0) = 1 \\ \wedge \mathbb{S}(\sigma) = 1 \wedge \mathsf{Verify}_{\mathsf{NIZK.crs}_1}(x_1, \pi_1) = 1 \\ \wedge \mathsf{Trriger}(\mathsf{tag}_{\mathbb{S}}, \mathsf{tag}_\sigma) = 0 \wedge \mathsf{Trriger}(\mathsf{tag}_{\mathbb{R}}, \mathsf{tag}_{f_\rho}) = 0 \end{array} \right\}.$$

また，LO のロック文字列 α を $\alpha \leftarrow\!\!\$\ \{0,1\}^{d(\lambda)}$ でサンプルする．ここで $x_{WE} := (\mathsf{Com.crs}, \mathsf{NIZK.crs}_0, \mathsf{NIZK.crs}_1, \mathsf{com}_0, \mathsf{com}_1, \sigma, \mathsf{tag}_\sigma, \mathbb{R}, \mathsf{tag}_{\mathbb{R}})$，$w_{WE} := (\rho, \mathsf{tag}_{f_\rho}, \mathbb{S}, \mathsf{tag}_{\mathbb{S}}, \pi_0, \pi_1)$ をセットし，$\mathrm{WE}.R$ および α に対し Witness Encryption の暗号文 $\mathsf{WE.ct}_\alpha \leftarrow \mathsf{WE.Enc}(1^\lambda, x_{WE}, \alpha)$ を計算する．さらに，WE の復号アルゴリズム $\mathsf{WE.Dec}(\mathsf{WE.ct}_\alpha, \cdot), \mathsf{m}, \alpha$ に対し，LO により難読化プログラムを $\tilde{P} \leftarrow \mathsf{Obf}(1^\lambda, \mathsf{WE.Dec}(\cdot, \mathsf{WE.ct}_\alpha), \mathsf{m}, \alpha)$ で出力する．最後に暗号文 $\mathsf{ct} := \tilde{P}$ を出力する．

– $\mathsf{Dec}(\mathsf{dk}_\rho, \mathsf{dk}_{\mathbb{S}}, \mathsf{ct})$：$\mathsf{m} \leftarrow \mathsf{Eval}(\tilde{P}, (\mathsf{dk}_\rho, \mathsf{dk}_{\mathbb{S}}))$ を計算し，m を出力する．

3.2 正当性

全てのセキュリティパラメータ $\lambda \in \mathbb{N}$，$\mathbb{S}(\sigma) = 1 \wedge \mathbb{R}(\rho) = 1$ を満たす属性 σ，ρ，ポリシー $\mathbb{S}$，$\mathbb{R}$，メッセージ m に対し，まず $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$，$\mathsf{sk}_\rho \leftarrow \mathsf{RKGen}(\mathsf{msk}, \rho)$，$\mathsf{dk}_{\mathbb{S}} \leftarrow \mathsf{PolGen}(\mathsf{msk}, \mathbb{S})$，$\mathsf{ct} \leftarrow \mathsf{Enc}(\sigma, \mathbb{R}, \mathsf{m})$ を正しく計算する．このとき，生成された復号鍵，ポリシー鍵に含まれる NIZK 証明，関数タグシステムについて，$\mathsf{Verify}_{\mathsf{NIZK.crs}_0}(x_0, \pi_0) = 1 \wedge \mathsf{Verify}_{\mathsf{NIZK.crs}_1}(x_1, \pi_1) = 1 \wedge \mathsf{Trriger}(\mathsf{tag}_{\mathbb{S}}, \mathsf{tag}_\sigma) = 0 \wedge \mathsf{Trriger}(\mathsf{tag}_{\mathbb{R}}, \mathsf{tag}_{f_\rho}) = 0$ が成立するため $\mathrm{WE}.R(x_{WE}, w_{WE}) = 1$ である．よって WE の正当性から，$\mathsf{WE.Dec}((\mathsf{dk}_\rho, \mathsf{dk}_{\mathbb{S}}), \mathsf{WE.ct}_\alpha) = \alpha$ が成立する．したがって $\mathsf{Dec}(\mathsf{dk}_\rho, \mathsf{dk}_{\mathbb{S}}, \mathsf{ct})$ について，LO の正当性から，$\mathsf{Eval}(\tilde{P}, (\mathsf{dk}_\rho, \mathsf{dk}_{\mathbb{S}})) = \mathsf{Eval}(\mathsf{Obf}(1^\lambda, \mathsf{WE.Dec}(\cdot, \mathsf{WE.ct}_\alpha), \mathsf{m}, \alpha), (\mathsf{dk}_\rho, \mathsf{dk}_{\mathbb{S}})) = \mathsf{m}$ である．

3.3 安全性証明

次にこの章では，3.1節で構成した ME について，Mismatch Condition における適応的 Privacy を証明する．

定理 1. Soundness を満たす NP に対する WE，NP に対する Statistically Sound な NIZK 証明システム，Statistically Binding なコミットメント，そして回路に対する関数タグシステムを仮定すると，Mismatch Condition において適応的 Privacy を満たす回路に対する ME が存在する．

Proof. ここでは，$\mathcal{A}$ を ME の Mismatch Condition の適応的 Privacy を破ろうとする攻撃者，$\mathcal{B}$ を各ゲーム間の仮定を破ろうとする帰着アルゴリズム，$\mathcal{C}$ を ME の適応的 Privacy に対する挑戦者とする．

まず，以下のようなゲーム列を定義する：

– $\mathrm{PMEGame}_0^b(\lambda)$：攻撃者 $\mathcal{A}$ と挑戦者間の適応的 Privacy ゲーム $\mathsf{Expt}^{\mathsf{privacy}}_{\mathcal{A},\mathsf{ME}}(\lambda)$ そのもの．

– $\mathrm{PMEGame}_1^b(\lambda)$：$\mathrm{PMEGame}_0^b(\lambda)$ から以下のようにゲームを修正する．まず，ゲームの最初に $\mathcal{C}$ が”スマートタグ鍵”$\mathsf{tsk} \leftarrow \mathsf{KGen}(1^\lambda)$ を選ぶ．次に $\mathcal{C}$ はポリシー生成クエリへの返答で，$\mathsf{tag}_{\mathbb{S}}$ について，ダミータグの代わりにスマートタグ $\mathsf{tag}_{\mathbb{S}} \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, \mathbb{S})$ を生成して，$\mathsf{dk}_{\mathbb{S}} := (\mathbb{S}, \mathsf{tag}_{\mathbb{S}}, \pi_1)$ で返答する．また $\mathcal{C}$ は暗号化クエリへの返答で，tag_{σ_b} について，ダミータグの代わりにスマートタグ $\mathsf{tag}_{\sigma_b} \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, \sigma_b)$ を生成して，これを用いてチャレンジ暗号文を生成する．

– $\mathrm{PMEGame}_2^b(\lambda)$：$\mathrm{PMEGame}_1^b(\lambda)$ から以下のようにゲームを修正する．まず $\mathcal{C}$ は復号鍵生成クエリへの返答で，tag_{f_ρ} について，ダミータグの代わりにスマートタグ $\mathsf{tag}_{f_\rho} \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f_\rho)$ を生成して，$\mathsf{dk}_{f_\rho} := (f_\rho, \mathsf{tag}_{f_\rho}, \pi_0)$ で返答する．また $\mathcal{C}$ は暗号化クエリへの返答で，$\mathsf{tag}_{\mathbb{R}_b}$ について，ダミータグの代わりにスマートタグ $\mathsf{tag}_{\mathbb{R}_b} \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, \mathbb{R}_b)$ を生成して，これを用いてチャレンジ暗号文を生成する．

– $\mathrm{PMEGame}_3^b(\lambda)$：$\mathrm{PMEGame}_2^b(\lambda)$ から以下のようにゲームを修正する．$\mathcal{C}$ が $\mathsf{mpk} = (\mathsf{Com.crs}, \mathsf{NIZK.crs}_0, \mathsf{NIZK.crs}_1, \mathsf{com}_0, \mathsf{com}_1)$ を選ぶ際，$\mathsf{com}_1 := \mathsf{Commit}_{\mathsf{Com.crs}}(\mathsf{tsk}; r_1)$ のように，$0^{d(\lambda)}$ ではなく tsk に対するコミットメントとする．

– $\mathrm{PMEGame}_4^b(\lambda)$：$\mathrm{PMEGame}_3^b(\lambda)$ から以下のようにゲームを修正する．$\mathcal{C}$ の復号鍵生成クエリへの返答で，証明 π_0 の生成について，$w_0 = r_0$ の代わりに $w_0 = (\mathsf{tsk}, r_1, r_2)$ を使って証明 $\pi_0 \leftarrow \mathsf{Prove}_{\mathsf{NIZK.crs}_0}(x_0 = (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, \rho, \mathsf{tag}_{f_\rho}), w_0 = (\mathsf{tsk}; r_1, r_2))$ を生成する．ただし，r_2 は $\mathsf{tag}_{f_\rho} \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f_\rho; r_2)$ の生成に使われた乱数である．

– $\mathrm{PMEGame}_5^b(\lambda)$：$\mathrm{PMEGame}_4^b(\lambda)$ から以下のようにゲームを修正する．$\mathcal{C}$ のポリシー生成クエリへの返答で，証明 π_1 の生成について，$w_1 = r_0$ の代わりに $w_1 = (\mathsf{tsk}, r_1, r_2)$ を使って証明 $\pi_1 \leftarrow \mathsf{Prove}_{\mathsf{NIZK.crs}_1}(x_1 = (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, \mathbb{S}, \mathsf{tag}_{\mathbb{S}}), w_1 = (\mathsf{tsk}; r_1, r_2))$ を生成する．ただし，r_2 は $\mathsf{tag}_{\mathbb{S}} \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, \mathbb{S}; r_2)$ の生成に使われた乱数である．

– $\mathrm{PMEGame}_6^b(\lambda)$：$\mathrm{PMEGame}_5^b(\lambda)$ から以下のようにゲームを修正する．$\mathcal{C}$ が $\mathsf{mpk} = (\mathsf{Com.crs}, \mathsf{NIZK.crs}_0, \mathsf{NIZK.crs}_1, \mathsf{com}_0, \mathsf{com}_1)$ を選ぶ際，$\mathsf{com}_0 \coloneqq \mathsf{Commit}_{\mathsf{Com.crs}}(1; r_0)$ のように，0 ではなく 1 に対するコミットメントとする．

– $\mathrm{PMEGame}_7^b(\lambda)$：$\mathrm{PMEGame}_6^b(\lambda)$ から以下のようにゲームを修正する．$\mathcal{C}$ が暗号化クエリに答える際，$\mathsf{WE.ct}$ の計算について $\alpha \leftarrow\!\$\ \{0,1\}^{d(\lambda)}$ の代わりに $\alpha = 0^{d(\lambda)}$ を用いて $\mathsf{WE.ct} \leftarrow \mathsf{WE.Enc}(1^\lambda, x_{WE}, \alpha = 0^{d(\lambda)})$ を計算する．

– $\mathrm{PMEGame}_8^b(\lambda)$：$\mathrm{PMEGame}_7^b(\lambda)$ から以下のようにゲームを修正する．$\mathcal{C}$ が暗号化クエリに答える際，チャレンジ暗号文 ct の生成において，$\mathsf{ct} \leftarrow \mathsf{Obf}(1^\lambda, \mathsf{ME.Dec}(\cdot, \mathsf{WE.ct}_\alpha, \mathsf{m}_b, \alpha))$ の代わりに，シミュレータ Sim を用いて $\mathsf{ct} \coloneqq \tilde{P} \leftarrow \mathsf{Sim}(1^\lambda, 1^{|\mathsf{ME.Dec}(\cdot,\cdot)|}, 1^{|\mathsf{m}|})$ を計算する．

次に，$Adv^{\mathsf{privacy}}_{ME,\mathcal{A}}(1^\lambda)$ を $\mathcal{A}$ の ME の適応的 Privacy に対する優位性，ϵ_i を $\mathcal{A}$ が $\mathrm{PMEGame}_i^b(1^\lambda)$ においてチャレンジビット b を推測できる確率とする．このとき，$Adv^{\mathsf{privacy}}_{\mathsf{ME},\mathcal{A}}(1^\lambda) = \left|\epsilon_0 - \frac{1}{2}\right|$ であり，

$$Adv^{\mathsf{privacy}}_{\mathsf{ME},\mathcal{A}}(1^\lambda) = \left|\epsilon_0 - \frac{1}{2}\right| \leq \sum_{i=0,\cdots,7} |\epsilon_i - \epsilon_{i+1}| + \left|\epsilon_8(\mathcal{A}) - \frac{1}{2}\right|$$

が成立するので，$|\epsilon_i - \epsilon_{i+1}|\ (for\ i = 0,\cdots,7) \leq \mathsf{negl}(\lambda)$，$\left|\epsilon_8 - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)$ を示せばよい．

さらに，2.1の Privacy の定義より，Mismatch Condition において，攻撃者に許されるクエリは以下の４パターンが存在する：

$$① \mathbb{R}_0(\rho) = \mathbb{R}_1(\rho) = 0 \ \vee\ ② \mathbb{S}(\sigma_0) = \mathbb{S}(\sigma_1) = 0 \vee$$
$$③ \mathbb{R}_0(\rho) = \mathbb{S}(\sigma_1) = 0 \ \vee\ ④ \mathbb{S}(\sigma_0) = \mathbb{R}_1(\rho) = 0$$

以降，Mismatch Condition のパターン ① における Privacy について考える．つまり，攻撃者がするクエリは全て $\mathbb{R}_0(\rho) = \mathbb{R}_1(\rho) = 0$ を満たすものとする．

補題 1. FTS が安全であれば，全ての PPT 攻撃者 $\mathcal{A}$ に対し $|\epsilon_0 - \epsilon_1| = \mathsf{negl}(\lambda)$ および $|\epsilon_1 - \epsilon_2| = \mathsf{negl}(\lambda)$ が成立する．

Proof. スペースの都合上，詳細は割愛するが，$\mathrm{PMEGame}_1^b(\lambda)$ および $\mathrm{PMEGame}_2^b(\lambda)$ の変更点は，FTS のタグの作り方の変更のみであるため，ME の適応的 Privacy に対する攻撃者 $\mathcal{A}$ を用いて，関数タグシステムの安全性を破ろうとする帰着アルゴリズム $\mathcal{B}$ を構築することで証明可能である． □

補題 2. COM が Hiding を満たすと仮定すれば，全ての PPT 攻撃者 $\mathcal{A}$ に対し $\epsilon_2 \overset{c}{\approx} \epsilon_3$ および $\epsilon_5 \overset{c}{\approx} \epsilon_6$ が成立する．

Proof. これらのゲーム間の差異はコミットメントの生成方法のみである．コミットメントの Hiding を仮定すれば，攻撃者 $\mathcal{A}$ はこの差異を識別できないため，$\epsilon_2 \overset{c}{\approx} \epsilon_3$ および $\epsilon_5 \overset{c}{\approx} \epsilon_6$ が成り立つ． □

補題 3. NIZK が Witness Indistinguishability を満たすと仮定すると，全ての PPT 攻撃者 $\mathcal{A}$ に対し $|\epsilon_3 - \epsilon_4| = \mathsf{negl}(\lambda)$ および $|\epsilon_4 - \epsilon_5| = \mathsf{negl}(\lambda)$ が成立する．

Proof. これらのゲーム間の差異は NIZK の証明の生成に利用した witness のみである．NIZK の Witness Indistinguishability を仮定すれば，攻撃者 $\mathcal{A}$ はこの差異を識別できないため，$|\epsilon_3 - \epsilon_4| = \mathsf{negl}(\lambda)$ および $|\epsilon_4 - \epsilon_5| = \mathsf{negl}(\lambda)$ が成り立つ． □

補題 4. WE が Soundness を満たし，かつ，NIZK が Statistical Soundness を満たすと仮定すると，全ての PPT 攻撃者 $\mathcal{A}$ に対し $|\epsilon_6 - \epsilon_7| = \mathsf{negl}(\lambda)$ が成立する．

Proof. まず，Com.crs が Statistical Binding で $\mathsf{NIZK.crs}_0, \mathsf{NIZK.crs}_1$ が Statistical Sound であれば，NP ステートメント x_{WE} は誤りであることを証明する．

まず，ある $w_{WE} = (\rho, \mathsf{tag}_{f_\rho}, \mathbb{S}, \mathsf{tag}_{\mathbb{S}}, \pi_0, \pi_1)$ に対し $(x_{WE}, w_{WE}) \in \mathsf{WE}.R$ を仮定する．このとき，$x_0 = (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, \rho, \mathsf{tag}_{f_\rho})$，$x_1 = (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, \mathbb{S}, \mathsf{tag}_{\mathbb{S}})$ として，$\mathsf{Verify}_{\mathsf{NIZK.crs}_0}(x_0, \pi_0) = 1 \ \wedge\ \mathsf{Verify}_{\mathsf{NIZK.crs}_1}(x_1, \pi_1) = 1 \ \wedge\ \mathbb{S}(\sigma) = 1 \ \wedge\ \mathbb{R}(\rho) = 1 \ \wedge\ \mathsf{Trriger}(\mathsf{tag}_{\mathbb{S}}, \mathsf{tag}_\sigma) = 0 \ \wedge\ \mathsf{Trriger}(\mathsf{tag}_{f_\rho}, \mathsf{tag}_{\mathbb{R}}) = 0$ の条件が同時に成立しなければならない．いま，$\mathsf{com}_0 = \mathsf{Commit}_{\mathsf{Com.crs}}(1; r_0)$，$\mathsf{com}_1 =$

$\mathsf{Commit}_{\mathsf{Com.crs}}(\mathsf{tsk}; r_1)$ で，ある r_2 に対し $\mathsf{tag}_{f_\rho} = \mathsf{SFunctionTag}(\mathsf{tsk}, f_\rho; r_2)$，$\mathsf{tag}_{\mathbb{S}} = \mathsf{SFunctionTag}(\mathsf{tsk}, \mathbb{S}; r_2)$ であり，また $\mathsf{tag}_{\mathbb{R}} = \mathsf{SInputTag}(\mathsf{tsk}, \mathbb{R})$，$\mathsf{tag}_{\sigma} = \mathsf{SInputTag}(\mathsf{tsk}, \sigma)$ である．よって，$\mathbb{S}(\sigma) = 1 \wedge \mathbb{R}(\rho) = 1$ と関数タグシステムのスマート正当性から $\mathsf{Trriger}(\mathsf{tag}_{\mathbb{S}}, \mathsf{tag}_{\sigma}) = 1 \wedge \mathsf{Trriger}(\mathsf{tag}_{f_\rho}, \mathsf{tag}_{\mathbb{R}}) = 1$ が成立しなければならない．これは上記の条件に矛盾するので，Com.crs が Statistical Binding で $\mathsf{NIZK.crs}_0, \mathsf{NIZK.crs}_1$ が Statistical Sound より，NP ステートメント x_{WE} は誤りである．以上から，攻撃者 $\mathcal{A}$ は誤ったステートメント x_{WE} に対し無視できない確率で $\mathsf{WE.Enc}(1^\lambda, x_{WE}, \alpha)$ と $\mathsf{WE.Enc}(1^\lambda, x_{WE}, 0^{d(\lambda)})$ を見分けなければならない．これに対し WE の Soundness を仮定すると，これらの識別は不可能であり $|\epsilon_6 - \epsilon_7| = \mathsf{negl}(\lambda)$ が成立する． □

補題 5. LO が安全であると仮定すると，全ての PPT 攻撃者 $\mathcal{A}$ に対し $|\epsilon_7 - \epsilon_8| = \mathsf{negl}(\lambda)$ が成立する．

Proof. スペースの都合上詳細は割愛するが，これらのゲームの差異は，難読化回路の生成方法の変更のみであるため，ME の適応的 Privacy に対する攻撃者 $\mathcal{A}$ を用いて，LO の安全性を破ろうとする帰着アルゴリズム $\mathcal{B}$ を構成し，LO の安全性ゲームをシミュレートすることで証明できる． □

また $\left|\epsilon_8 - \frac{1}{2}\right|$ については以下のように変形できる：

$$\begin{aligned}\left|\epsilon_8 - \frac{1}{2}\right| &= \left|\Pr\left[\mathrm{PMEGame}_8^{\mathrm{b}}(1^\lambda) = 1\right] - \frac{1}{2}\right| \\ &= \frac{1}{2}\Big|\Pr\left[\mathrm{PMEGame}_8^{0}(1^\lambda) = 1\right] \\ &\qquad - \Pr\left[\mathrm{PMEGame}_8^{1}(1^\lambda) = 1\right]\Big|.\end{aligned}$$

$\mathrm{PMEGame}_8^{b}(1^\lambda)$ はビット b によらないため，$\epsilon_8 = \frac{1}{2}$ である．したがって，$\left|\epsilon_8 - \frac{1}{2}\right| = 0$ である．

以上より，$|\epsilon_i - \epsilon_{i+1}|$ $(for \quad i = 0, \cdots, 7) = \mathsf{negl}(\lambda)$，$\left|\epsilon_8 - \frac{1}{2}\right| = \mathsf{negl}(\lambda)$ が示されたので，パターン ① に対する Mismatch Condition の Privacy が示された．

なお，Mismatch Condition Privacy の他のパターン ②, ③, ④ について考えると，① 同様攻撃者 $\mathcal{A}$ が Admissible であり，またクエリした復号鍵・ポリシーがいずれも暗号文に対して無効になるため，① と同様に Privacy を証明できる．以上から，我々の ME が Mismath Condition における適応的 Privacy を満たすことが証明された． □

4. まとめ

本稿では，WE，コミットメント，NIZK，関数タグシステム，LO から新たな ME を構成した．また構成した ME について，Mismatch Condition における適応的 Privacy を証明した．今後の課題についてだが，まず Match Condition における Privacy の証明が挙げられる．前述の通り，ME の Privacy は Mismatch Condition と Match Condition の両方が求められるため，Match Condition Privacy の証明は解決の必要がある課題である．また，今回の構成で採用した技術については，基本的に WE から作れることが知られる一方，LO については WE から作れるかどうかが未知である．したがって，LO を WE から作れるような技術に置き換えることで，ME を WE のみから作れることを示すことも，今後の課題である．

謝辞 本研究の一部は JSPS 科研費 JP23K24846, JP24K20776, JP25KJ1319, JST CREST JPMJCR22M1 の助成を受けたものです．また，本研究の一部は，JST 経済安全保障重要技術育成プログラム【JPMJKP24U2】の支援を受けたものです．

参考文献

[1] G. Ateniese, D. Francati, D. Nuñez, and D. Venturi. Match me if you can: Matchmaking encryption and its applications. CRYPTO 2019, Part II, LNCS 11693, pp. 701–731.
[2] G. Ateniese, D. Francati, D. Nuñez, and D. Venturi. Match me if you can: Matchmaking encryption and its applications. Journal of Cryptology, 34(3):16.
[3] U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In 31st FOCS, pp. 308–317.
[4] D. Francati, D. Friolo, G. Malavolta, and D. Venturi. Multi-key and multi-input predicate encryption (for conjunctions) from learning with errors. Journal of Cryptology, 37(3):24.
[5] S. Garg, C. Gentry, A. Sahai, and B. Waters. Witness encryption and its applications. 45th ACM STOC, pp. 467–476.
[6] R. Goyal, V. Koppula, and B. Waters. Lockable obfuscation. 58th FOCS, pp. 612–621.
[7] J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for NP. EUROCRYPT 2006, LNCS 4004, pp. 339–358.
[8] M. Naor. Bit commitment using pseudorandomness. Journal of Cryptology Volume 4.
[9] B. Waters and D. Wichs. Adaptively secure attribute-based encryption from witness encryption. TCC 2024, Part III, LNCS 15366, pp. 65–90.
[10] D. Wichs and G. Zirdelis. Obfuscating compute-and-compare programs under LWE. 58th FOCS, pp. 600–611.