Computer Security Symposium 2020
26 - 29 October 2020

集団検査機能を有する準同型認証暗号

佐藤 慎悟[1,a)]

概要：準同型認証暗号 (Homomorphic Authenticated Encryption, HAE) は, データを暗号化したまま演算処理が可能な共通鍵暗号であり, 安全性として秘匿性と完全性を同時に達成する. しかし, 既存の HAE 方式では, 複数の暗号文を演算処理して復号が失敗した場合に, どの暗号文が改ざんされたのか検出することができない. 本稿では, 改ざんされた暗号文を検出可能な, 集団検査機能を有する準同型認証暗号 (Group-Testing Homomorphic Authenticated Encryption, GT-HAE) を導入する. GT-HAE のモデルと安全性を定義し, 定式化した安全性を満たす一般的構成を提案する.

キーワード：共通鍵暗号, 準同型認証暗号, 集団検査機能

Group-Testing Homomorphic Authenticated Encryption

Shingo Sato[1,a)]

Abstract: Homomorphic authenticated encryption (HAE) is symmetric key encryption which performs computations on encrypted data, and guarantees both confidentiality and integrity. However, existing HAE schemes cannot detect invalid encrypted data in the case where decryption rejects a ciphertext generated by the computation on encrypted one. In this paper, we introduce the concept of group-testing homomorphic authenticated encryption (GT-HAE) which can detect invalid ciphertexts. We define the model and the security of GT-HAE, and propose generic constructions which satisfy the formalized security.

Keywords: symmetric key cryptography, homomorphic authenticated encryption, group-testing

1. はじめに

1.1 背景

準同型認証暗号 (Homomorphic Authenticated Encryption, HAE) [5] は, データを暗号化したまま演算処理が可能な共通鍵暗号であり, 安全性として秘匿性と完全性の両方を満たす. データの秘匿性あるいは偽造不可能性を保証したまま演算処理できる暗号技術は, クラウド計算やビッグデータ分析など多くの応用が考えられ, 研究が盛んに行われている. このような機能をもつ暗号技術の 1 つである HAE は有用であり, 本稿では, これに焦点を当てる.

既存の HAE 方式として算術回路を演算処理する方式 [5] や任意の回路を演算処理できる Fully HAE 方式 [2] がある. また, [1] による認証暗号の一般的構成に適切なプリミティブを適用することで HAE 方式を構成することが可能である. しかし, これらの方式では, 改ざんされた暗号文を含む複数の暗号文を演算処理した場合に, 復号処理したユーザは演算処理された暗号文が不正であることしか知らされず, 演算前の暗号文の中のどれが改ざんされたのかを特定することができない.

一方, 集団検査 (Group-Testing, GT) プロトコルは, 陽性か陰性に分類されるアイテムの集合の中から陽性アイテムを検出するプロトコルであり, 個々に検査する場合と比べて少ない検査回数で検出することができる. この技術には様々な分野への応用があり, 暗号技術への応用としては認証技術などへの適用に関して既存研究 [3], [4], [6], [7] がある.

[1] 国立研究開発法人情報通信研究機構
National Institute of Information and Communications Technology (NICT).
[a)] shingo-sato@nict.go.jp

1.2 本研究の貢献

本稿では, 改ざんされた暗号文を検出できる, 集団検査機能を有する準同型認証暗号 (Group-Testing Homomorphic Authenticated Encryption, GT-HAE) を導入する. このような機能を実現することにより, 演算処理で得られた暗号文の復号が失敗した場合に, 改ざんされた不正な暗号文を特定し正当な暗号文のみを演算処理することが可能である.

本研究において, GT-HAE のモデルと安全性を定義し, この定式化した安全性を満たす GT-HAE の一般的構成を 2 つ提案する. 具体的な貢献は以下の通りである.

- GT-HAE のモデルを定義し, 秘匿性, 完全性, identifiablity の安全性概念を考慮する. 秘匿性として IND-GT-CCA 安全性, 完全性として UF-GT-CCA 安全性と, これよりも強い安全性 sUF-GT-CCA 安全性を定式化する. 正当な暗号文か不正な暗号文かを正しく識別する安全性 identifiability について, ident-completeness と ident-(weak-)soundness を定式化する.
- 以下の GT-HAE の一般的構成を提案する.
  - GTHAE_1: HAE 方式と GT プロトコルから構成され, 次の安全性を満たす:
    IND-GT-CCA 安全性, sUF-GT-CCA 安全性, ident-completeness と ident-weak-soundness.
  - GTHAE_2: HAE 方式, GT プロトコル, 簡潔な非対話ゼロ知識アーギュメント (Zero-Knowledge Succinct Non-interactive Argument, zk-SNARG) から構成され, 次の安全性を満たす:
    IND-GT-CCA 安全性, UF-GT-CCA 安全性, ident-completeness と ident-soundness.

これらの構成の違いとして, GTHAE_1 は完全性について sUF-GT-CCA 安全性を達成するのに対して, GTHAE_2 はそれよりも弱い UF-GT-CCA 安全性を満たす. 一方で, identifiability については GTHAE_1 は ident-weak-soundness を満たすのに対して, GTHAE_2 はこれよりも強い安全性 ident-soundness を達成する.

なお, これらの一般的構成に適切な既存方式を適用することにより具体的な GT-HAE 方式が得られる.

2. 準備

本稿において次の記法を用いる: 正の整数 n に対して, $[n] := \{1, \ldots, n\}$ とする. n 個の値 $x_1, \ldots, x_n$ と, インデックスの部分集合 $I \subseteq [n]$ に対して, $(x_i)_{i \in I}$ をインデックス $i \in [n]$ が I に含まれる値 x_i の列とする. ベクトル $\boldsymbol{x}$ の第 i 成分を x_i とする. 行列 $\boldsymbol{X}$ の (i, j) 成分を $x_{i,j}$ とする. 関数 $f : \mathbb{N} \to \mathbb{R}$ に対して, 任意の定数 $c > 0$ と十分大きい $\lambda \in \mathbb{N}$ に対して $f(\lambda) = o(\lambda^{-c})$ が成り立つとき, f は λ において無視できるほど小さいといい, $f(\lambda) = \mathsf{negl}(\lambda)$ と記述する. ある確率が $1 - \mathsf{negl}(\lambda)$ のとき圧倒的確率という. "確率的多項式時間"を PPT と記述する. 確率的アルゴリズム A に対して, $y \leftarrow \mathsf{A}(x; r)$ は, x を入力として乱数 r を用いて y を出力することを意味する.

さらに, HAE と GT-HAE を定義するにあたって, [2] で導入された概念 labeled-program を定義する. ラベル空間 $\mathcal{T}$ とメッセージ空間 $\mathcal{M}$ に対して, labeled-program $\mathcal{P}$ は $(f, \tau_1, \ldots, \tau_\ell)$ で構成され, $f : \mathcal{M}^\ell \to \mathcal{M}$ は回路であり, $\tau_1, \ldots, \tau_\ell \in \mathcal{T}$ は f の入力に対するラベルである. $\mathsf{m}_i \in \mathcal{M}$ が τ_i において暗号化されるならば ($i \in [\ell]$), labeled-program $\mathcal{P} = (f, \tau_1, \ldots, \tau_\ell)$ の評価は $f(\mathsf{m}_1, \ldots, \mathsf{m}_\ell)$ である. また, ラベル τ における identity labeled-program を $\mathcal{I}_\tau = (\mathsf{idf}, \tau)$ とする ($\mathsf{idf} : \mathcal{M} \to \mathcal{M}$ は恒等写像).

2.1 準同型認証暗号 (HAE)

HAE 方式は 4 つの多項式時間アルゴリズム (KGen, Enc, Eval, Dec) で構成される: セキュリティパラメータ λ に対して, $\mathcal{M} = \mathcal{M}(\lambda)$ をメッセージ空間, $\mathcal{T} = \mathcal{T}(\lambda)$ をラベル空間, $\mathcal{CT} = \mathcal{CT}(\lambda)$ を暗号文空間, 許容されるゲートで構成される回路の空間を $\mathcal{F}$ とする.

- $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda)$: セキュリティパラメータ 1^λ を入力とし, 秘密鍵 sk と評価鍵 ek を出力する.
- $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau, \mathsf{m})$: 秘密鍵 sk, ラベル $\tau \in \mathcal{T}$, メッセージ $\mathsf{m} \in \mathcal{M}$ を入力とし, 暗号文 $\mathsf{ct} \in \mathcal{CT}$ を出力する.
- $\widehat{\mathsf{ct}} \leftarrow \mathsf{Eval}(\mathsf{ek}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$: 評価鍵 ek, 回路 $f \in \mathcal{F}$, 暗号文の組 $(\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$ を入力とし, 新しい暗号文 $\widehat{\mathsf{ct}} \in \mathcal{CT}$ を出力する.
- $\mathsf{m}/\perp \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathcal{P}, \widehat{\mathsf{ct}})$: 秘密鍵 sk, labeled-program $\mathcal{P} = (f, \tau_1, \ldots, \tau_\ell)$, 暗号文 $\widehat{\mathsf{ct}} \in \mathcal{CT}$ を入力とし, メッセージ $\mathsf{m} \in \mathcal{M}$ または拒否シンボル $\perp$ を出力する.

ここで, KGen, Enc は確率的アルゴリズム, Dec は決定性アルゴリズムである. また, HAE 方式は correctness と compactness を満たすことが要求される.

定義 1 (Correctness)**.** 次が成り立つとき, *HAE* 方式 $\mathsf{HAE} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ は *correctness* を満たす*:*

- 全ての $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda)$, 全ての $\tau \in \mathcal{T}$, 全ての $\mathsf{m} \in \mathcal{M}$ に対して, $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau, \mathsf{m})$ ならば, 圧倒的確率で $\mathsf{Dec}(\mathsf{sk}, \mathcal{I}_\tau, \mathsf{ct}) = \mathsf{m}$ が成り立つ.
- 全ての $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda)$, $i \in [\ell]$ における全ての $\tau_i \in \mathcal{T}$ と全ての $\mathsf{m}_i \in \mathcal{M}$ に対して, $i \in [\ell]$ と任意の $f \in \mathcal{F}$ において $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau_i, \mathsf{m}_i)$, $\widehat{\mathsf{ct}} \leftarrow \mathsf{Eval}(\mathsf{sk}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$, $\mathcal{P} = (f, \tau_1, \ldots, \tau_\ell)$ ならば, 圧倒的確率で $\mathsf{Dec}(\mathsf{sk}, \mathcal{P}, \widehat{\mathsf{ct}}) = f(\mathsf{m}_1, \ldots, \mathsf{m}_\ell)$ が成り立つ.

定義 2 (Compactness)**.** 次が成り立つとき, *HAE* 方式 $\mathsf{HAE} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ は *compactness* を満たす*:* 全てのセキュリティパラメータ λ に対して, ある多項式 poly が存在して $\mathsf{Eval}(\mathsf{ek}, \cdot, \cdot)$ と $\mathsf{Enc}(\mathsf{sk}, \cdot, \cdot)$ の出力サイズが高々 $\mathsf{poly}(\lambda)$ である.

さらに, 関数 $\widetilde{f}_{(\mathsf{m}_i)_{i \in I}}$, $\widetilde{\mathsf{ct}}_{f, (\mathsf{ct}_i)_{i \in I}}$ を定義する. HAE 方式

(KGen, Enc, Eval, Dec), 回路 $f \in \mathcal{F}$, 部分集合 $I \subseteq [\ell]$, メッセージの組 $(\mathsf{m}_i)_{i\in I} \in \mathcal{M}^{|I|}$, 暗号文の組 $(\mathsf{ct}_i)_{i\in I}$ (各 $i \in I$ に対して $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau_i, \mathsf{m}_i)$) に対して,

$$\tilde{f}_{(\mathsf{m}_i)_{i\in I}} = f((\mathsf{m}_i)_{i\in I}),$$
$$\widetilde{\mathsf{ct}}_{f,(\mathsf{ct}_i)_{i\in I}} = \mathsf{Eval}(\mathsf{ek}, f, (\mathsf{ct}_i)_{i\in I}).$$

とする. $(\mathsf{m}_j)_{j\notin I} \in \mathcal{M}^{\ell-|I|}$ と $(\mathsf{ct}_j)_{j\notin I} \in \mathcal{CT}^{\ell-|I|}$ に対して, $\tilde{f}_{(\mathsf{m}_i)_{i\in I}}(\mathsf{m}_j)_{j\notin I} = f(\mathsf{m}_1, \ldots, \mathsf{m}_\ell)$ と $\widetilde{\mathsf{ct}}_{f,(\mathsf{ct}_i)_{i\in I}}(\mathsf{ct}_j)_{j\notin I} = \mathsf{Eval}(\mathsf{ek}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$ であることに注意する.

HAE において *ciphertext constant-testability* (CCT) を満たすことが望ましい.

定義 3 (CCT)**.** 次が成り立つとき, *HAE* 方式 $\mathsf{HAE} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ は CCT を満たす*:* 全ての $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda)$, 全ての $f \in \mathcal{F}$, 全ての $I \subseteq [\ell]$, 全ての $(\mathsf{ct}_i)_{i\in I} \in \mathcal{CT}^{|I|}$ に対して, 圧倒的確率で $\widetilde{\mathsf{ct}}_{f,(\mathsf{ct}_i)_{i\in I}}$ が *constant* か *not-constant* か判定する *PPT* アルゴリズムが存在する.

HAE 方式の安全性として秘匿性と完全性について定義する. まず, 暗号化オラクル ENC を定義する: ENC は, 暗号化クエリ $(\tau, \mathsf{m}) \in \mathcal{T} \times \mathcal{M}$ を受け取る. $(\tau, \cdot, \cdot) \in \mathcal{Q}$ ならば, $\perp$ を返す. そうでなければ, $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau, \mathsf{m})$ を返して, $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\tau, \mathsf{m}, \mathsf{ct})\}$ とする. ここで, $\mathcal{Q}$ は ENC オラクルに対するクエリとその応答のリストである.

秘匿性として IND-CPA 安全性と IND-CCA 安全性が定義される. 本稿では IND-CPA 安全性のみを定義する (IND-CCA 安全性の定義は [5] を参照せよ).

定義 4 (IND-CPA 安全性)**.** 次が成り立つとき, *HAE* 方式 $\mathsf{HAE} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ は IND-CPA 安全性を満たす*:* 任意の *PPT* 攻撃者 $\mathsf{A} = (\mathsf{A}_0, \mathsf{A}_1)$ に対して, アドバンテージ

$$\mathsf{Adv}^{\text{ind-cpa}}_{\mathsf{HAE},\mathsf{A}}(\lambda) := \left| \Pr\left[b = b' \;\middle|\; \begin{array}{l} (\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda); \mathcal{Q} \leftarrow \emptyset; \\ (\tau^*, \mathsf{m}_0, \mathsf{m}_1, \mathsf{st}) \leftarrow \mathsf{A}_0^{\mathsf{ENC}}(\mathsf{ek}); \\ b \xleftarrow{\$} \{0,1\};\ \mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau^*, \mathsf{m}_b); \\ b' \leftarrow \mathsf{A}_1^{\mathsf{ENC}}(\mathsf{ek}, \mathsf{ct}^*, \mathsf{st}) \end{array} \right] - \frac{1}{2} \right|$$

が λ において無視できるほど小さい. ここで, st は状態情報であり, A は $(\tau^*, \cdot, \cdot) \in \mathcal{Q}$ を満たす *challenge* $(\tau^*, \mathsf{m}_0, \mathsf{m}_1)$ を発行することが許されない.

完全性として UF-CPA 安全性, UF-CCA 安全性, sUF-CPA 安全性, sUF-CCA 安全性が定義される. 本稿では sUF-CPA 安全性のみ定義する (UF-CPA 安全性, UF-CCA 安全性, sUF-CCA 安全性の定義は [5] を参照せよ).

定義 5 (sUF-CPA 安全性)**.** 次が成り立つとき, *HAE* 方式 $\mathsf{HAE} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ は sUF-CPA 安全性を満たす*:* 任意の *PPT* 攻撃者 A に対して, アドバンテージ

$$\mathsf{Adv}^{\text{suf-cpa}}_{\mathsf{HAE},\mathsf{A}}(\lambda) := \Pr\left[\mathsf{A}\ \textit{wins} \;\middle|\; \begin{array}{l} (\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda); \mathcal{Q} \leftarrow \emptyset; \\ (\mathcal{P}^*, \widehat{\mathsf{ct}}^*) \leftarrow \mathsf{A}^{\mathsf{ENC}}(\mathsf{ek}) \end{array} \right]$$

が λ において無視できるほど小さい. ここで [A *wins*] を, 次が成り立つ事象とする*:*

- $\mathsf{Dec}(\mathsf{sk}, \mathcal{P}^*, \widehat{\mathsf{ct}}^*) \neq \perp$, かつ
- 次のいずれかが成り立つ*:*
  - $\widetilde{\mathsf{ct}}$ が *not-constant* である.
  - $\widetilde{\mathsf{ct}}$ が *constant* かつ $\widehat{\mathsf{ct}}^* \neq \widetilde{\mathsf{ct}}$ である.

ただし, $\widetilde{\mathsf{ct}} = \widetilde{\mathsf{ct}}_{f^*,(\mathsf{ct}_i)_{i\in I}}$, $\mathcal{P}^* = (f^*, \tau_1^*, \ldots, \tau_\ell^*)$, $I = \{i \in [\ell] \mid (\tau_i^*, \mathsf{m}_i, \mathsf{ct}_i) \in \mathcal{Q} \wedge \mathsf{m}_i \in \mathcal{M} \wedge \mathsf{ct}_i \in \mathcal{CT}\}$ とする.

2.2 集団検査 (GT) プロトコル

GT プロトコルの処理の主な流れとしては, 与えられたアイテムの集合に対して複数の部分集合を選び, 各部分集合に対して含まれるアイテムを混合してテストする. これらのテスト結果に応じて陽性アイテムを特定する. 仮定として, アイテムの (部分) 集合のテストにおいて, 陽性アイテムが 1 つ以上あればテスト結果は陽性を示し, 全て陰性アイテムであればテスト結果は陰性を示すとする. 本稿では d-disjunct 行列を用いる GT プロトコルを考える.

定義 6 (d-disjunct 行列)**.** 次が成り立つとき, 行列 $\boldsymbol{G} = [\boldsymbol{g}_1, \ldots, \boldsymbol{g}_\ell] \in \{0,1\}^{u\times\ell}$ は *d-disjunct* である *(*各 $i \in [\ell]$ に対して $\boldsymbol{g}_i \in \{0,1\}^u$*):* 任意の d 個の列ベクトル $\boldsymbol{g}_{s_1}, \ldots, \boldsymbol{g}_{s_d}$ と任意の $\bar{\boldsymbol{g}} \in \{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_\ell\} \backslash \{\boldsymbol{g}_{s_1}, \ldots, \boldsymbol{g}_{s_d}\}$ に対して *($s_1, \ldots, s_d \in [\ell]$)*, $\boldsymbol{g}^{(uni)} = \bigvee_{i=1}^d \boldsymbol{g}_{s_i}$ として, ある $z \in [u]$ において $g_z^{(uni)} < \bar{g}_z$ が成り立つ. ここで, $\bigvee$ はビット演算の論理和とする.

d-disjunct 行列を用いることによって, ℓ 個のアイテムが与えられて高々 d 個の陽性アイテムを検出することが可能である. d-disjunct 行列 $\boldsymbol{G} \in \{0,1\}^{u\times\ell}$ を用いた GT プロトコルの処理の流れを記述する: ℓ 個のアイテムにインデックス $1, \ldots, \ell$ を割り当て, インデックスの部分集合 $S_i(\boldsymbol{G}) = \{j \mid j \in [\ell] \wedge g_{i,j} = 1\}$ を定義する.

(1) $J \leftarrow \{1, \ldots, \ell\}$.

(2) 各 $i \in [u]$ に対して, $S_i(\boldsymbol{G})$ に属するインデックスをもつアイテムを混合してテストを行う. テスト結果が陰性ならば, $J \leftarrow J \backslash \{k\}_{k\in S_i(\boldsymbol{G})}$ とする.

(3) J を出力する.

このとき, $\boldsymbol{G}$ が d-disjunct ならば, 出力された J は陽性アイテムのインデックスの集合である.

2.3 簡潔な非対話ゼロ知識アーギュメント (zk-SNARG)

本稿では公開検証可能な zk-SNARG のみを考える. 関係 $R \subseteq \{0,1\}^* \times \{0,1\}^*$ に対する zk-SNARG は 3 つの多項式時間アルゴリズム (Gen, P, V) で構成される: R で定義される言語を $L(R) = \{x \mid \exists w \text{ s.t. } (x, w) \in R\}$ とする.

- $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)$: セキュリティパラメータ 1^λ を入力とし, common reference string crs を出力する.
- $\pi \leftarrow \mathsf{P}(\mathsf{crs}, x, w)$: crs, ステートメント x, 証拠 w を入力とし, 証明 π を出力する.

- $1/0 \leftarrow \mathsf{V}(\mathsf{crs}, x, \pi)$: crs, ステートメント x, 証明 π を入力とし, 1 または 0 を出力する.

ここで, Gen, P は確率的アルゴリズム, V は決定性アルゴリズムである. zk-SNARG は以下の 4 つの性質を満たすことが要求される:

Completeness 全ての $(x, w) \in R$ に対して, 次が成り立つ:

$$\Pr\left[\mathsf{V}(\mathsf{crs}, x, \pi) = 1 \;\middle|\; \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda); \\ \pi \leftarrow \mathsf{P}(\mathsf{crs}, x, w) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

Soundness 任意の PPT アルゴリズム A に対して, 次が成り立つ:

$$\Pr\left[\begin{array}{c} \mathsf{V}(\mathsf{crs}, x, \pi) = 1 \wedge \\ x \notin L(R) \end{array} \;\middle|\; \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda); \\ (x, \pi) \leftarrow \mathsf{A}(\mathsf{crs}) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Zero-Knowledge PPT シミュレータ $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ が存在し, 任意の PPT アルゴリズム A に対して次が成り立つ:

$$\begin{aligned} &\left|\Pr\left[1 \leftarrow \mathsf{A}^{\mathsf{P}^*}(\mathsf{crs}) \mid \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)\right]\right. \\ &\quad \left. - \Pr\left[1 \leftarrow \mathsf{A}^{\mathsf{Sim}^*}(\mathsf{crs}) \mid (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Sim}_0(1^\lambda)\right]\right| \\ &\leq \mathsf{negl}(\lambda). \end{aligned}$$

ここで, $\mathsf{Sim}_0(1^\lambda)$ は crs とトラップドア td を生成し, $\mathsf{Sim}_1(\mathsf{td}, x)$ は証明 π を生成する. また, オラクル $\mathsf{P}^*(x, w)$ は $\pi \leftarrow \mathsf{P}(\mathsf{crs}, x, w)$ を返す. オラクル $\mathsf{Sim}^*(x, w)$ は, $(x, w) \notin R$ ならば $\perp$ を返し, そうでなければ $\pi \leftarrow \mathsf{Sim}_1(\mathsf{td}, x)$ を返す.

Succinctness ある多項式 poly が存在し, 証明サイズと Gen, V の時間計算量が高々 $\mathsf{poly}(\lambda + |x|)$ である.

3. 集団検査機能を有する準同型認証暗号 (GT-HAE)

GT-HAE 方式は 6 つの多項式時間アルゴリズム (KGen, Enc, Eval, Dec, GTctxt, GTident) で構成される: セキュリティパラメータ λ に対して, $\mathcal{M} = \mathcal{M}(\lambda)$ をメッセージ空間, $\mathcal{CT} = \mathcal{CT}(\lambda)$ を暗号文空間, $\mathcal{T} = \mathcal{T}(\lambda)$ をラベル空間, 許容されるゲートで構成される回路の空間を $\mathcal{F}$ とする.

- $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda)$:セキュリティパラメータ 1^λ を入力とし, 秘密鍵 sk と評価鍵 ek を出力する.
- $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau, \mathsf{m})$: 秘密鍵 sk, ラベル $\tau \in \mathcal{T}$, メッセージ $\mathsf{m} \in \mathcal{M}$ を入力とし, 暗号文 $\mathsf{ct} \in \mathcal{CT}$ を出力する.
- $\widehat{\mathsf{ct}} \leftarrow \mathsf{Eval}(\mathsf{ek}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$: 評価鍵 ek, 回路 $f \in \mathcal{F}$, ℓ 個の暗号文の組 $(\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$ を入力とし, 新しい暗号文 $\widehat{\mathsf{ct}} \in \mathcal{CT}$ を出力する.
- $\mathsf{m}/\perp \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathcal{P}, \mathsf{ct})$: 秘密鍵 sk, labeled-program $\mathcal{P} = (f, \tau_1, \ldots, \tau_\ell)$, 暗号文 $\widehat{\mathsf{ct}} \in \mathcal{CT}$ を入力とし, メッセージ $\mathsf{m} \in \mathcal{M}$ か拒否シンボル $\perp$ を出力する.
- $(\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u) \leftarrow \mathsf{GTctxt}(\mathsf{ek}, \boldsymbol{G}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$: 評価鍵 ek, d-disjunct 行列 $\boldsymbol{G} \in \{0,1\}^{u \times \ell}$, 回路 $f \in \mathcal{F}$, ℓ 個の暗号文の組 $(\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$ を入力とし, 新しい u 個の暗号文の組 $(\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u)$ を出力する.
- $J \leftarrow \mathsf{GTident}(\mathsf{sk}, \boldsymbol{G}, \mathcal{P}, (\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u))$: 秘密鍵 sk, d-disjunct 行列 $\boldsymbol{G} \in \{0,1\}^{u \times \ell}$, labeled-program $\mathcal{P} = (f, \tau_1, \ldots, \tau_\ell)$, u 個の暗号文の組 $(\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u)$ を入力とし, ラベルの集合 J を出力する.

ここで, Gen, Enc は確率的アルゴリズムであり, Dec, GTident は決定性アルゴリズムである.

定義 7 (Correctness)**.** *定義 1 の条件に加えて次が成り立つとき, GT-HAE 方式* $\mathsf{GTHAE} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{GTctxt}, \mathsf{GTident})$ *は correctness を満たす: 全ての* $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda)$, *全ての* $f \in \mathcal{F}$, *全ての d-disjunct 行列* $\boldsymbol{G} \in \{0,1\}^{u \times \ell}$, *全ての* $(\tau_1, \ldots, \tau_\ell) \in \mathcal{T}^\ell$, *全ての* $(\mathsf{m}_1, \ldots, \mathsf{m}_\ell) \in \mathcal{M}^\ell$ *に対して,* $\mathsf{GTident}(\mathsf{sk}, \boldsymbol{G}, \mathcal{P}, (\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u)) = \emptyset$ *が成り立つ. ここで,* $\mathcal{P} = (f, \tau_1, \ldots, \tau_\ell)$, *各* $i \in [\ell]$ *に対して* $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau_i, \mathsf{m}_i)$, $(\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u) \leftarrow \mathsf{GTctxt}(\mathsf{ek}, \boldsymbol{G}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$ *とする.*

GT-HAE の compactness と CCT はそれぞれ定義 2 と定義 3 と同様に定義される.

GT-HAE の安全性として, 秘匿性, 完全性, identifiablity を定式化する. まず, 以下のオラクルを定義する:

- ENC: 暗号化オラクル ENC は, 暗号化クエリ $(\tau, \mathsf{m}) \in \mathcal{T} \times \mathcal{M}$ を受け取る. $(\tau, \cdot, \cdot) \in \mathcal{Q}$ ならば, $\perp$ を返す. そうでなければ, $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau, \mathsf{m})$ を返して, $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\tau, \mathsf{m}, \mathsf{ct})\}$ とする. ここで, $\mathcal{Q}$ は ENC オラクルに対するクエリとその応答のリストである.
- DEC: 復号オラクル DEC は, 復号クエリ $(\mathcal{P}, \widehat{\mathsf{ct}})$ を受け取り, $\mathsf{m}/\perp \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathcal{P}, \widehat{\mathsf{ct}})$ を返す.
- GTIDENT: GT オラクル GTIDENT は, GT クエリ $(\boldsymbol{G}, \mathcal{P}, (\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u))$ を受け取り, $J \leftarrow \mathsf{GTident}(\mathsf{sk}, \boldsymbol{G}, \mathcal{P}, (\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u))$ を返す.

秘匿性として IND-GT-CCA 安全性, 完全性として UF-GT-CCA 安全性と sUF-GT-CCA 安全性を定義する.

定義 8 (IND-GT-CCA 安全性)**.** *次が成り立つとき, GT-HAE 方式* GTHAE = (KGen, Enc, Eval, Dec, GTctxt, GTident) *は IND-GT-CCA 安全性を満たす: 任意の PPT 攻撃者* $\mathsf{A} = (\mathsf{A}_0, \mathsf{A}_1)$ *に対して, アドバンテージ*

$$\mathsf{Adv}^{\text{ind-gt-cca}}_{\mathsf{GTHAE},\mathsf{A}}(\lambda) := \left|\Pr\left[b = b' \;\middle|\; \begin{array}{l} (\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda); \mathcal{Q} \leftarrow \emptyset; \\ (\tau^*, \mathsf{m}_0, \mathsf{m}_1, \mathsf{st}) \\ \quad \leftarrow \mathsf{A}_0^{\mathsf{ENC},\mathsf{DEC},\mathsf{GTIDENT}}(\mathsf{ek}); \\ b \xleftarrow{\$} \{0,1\};\ \mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau^*, \mathsf{m}_b); \\ b' \leftarrow \mathsf{A}_1^{\mathsf{ENC},\mathsf{DEC},\mathsf{GTIDENT}}(\mathsf{ek}, \mathsf{ct}^*, \mathsf{st}) \end{array}\right] - \frac{1}{2}\right|$$

が λ において無視できるほど小さい．ただし，A は以下のことが禁止される：

- $(\tau^*, \cdot, \cdot) \in \mathcal{Q}$ である *challenge* $(\tau^*, \mathsf{m}_0, \mathsf{m}_1)$ を発行する．
- DEC あるいは GTIDENT に対して *illegal labeled-program* $\mathcal{P}$ を含むクエリを発行する．

ここで，st は状態情報とし，次を満たすとき *labeled-program* $\mathcal{P} = (f, \tau_1, \ldots, \tau_\ell)$ は *illegal* である：ある $i^* \in I = \{i \in [\ell] \mid (\tau_i, \mathsf{m}_i, \mathsf{ct}_i) \in \mathcal{Q}\}$ が存在し，$\tau_{i^*} = \tau^*$ かつ $\widetilde{f}_0 \neq \widetilde{f}_1$ が成り立つ．$\mathsf{m}_{i^*} = \mathsf{m}_0$ に対して $\widetilde{f}_0 = f_{(\mathsf{m}_i)_{i\in I}}$，$\mathsf{m}_{i^*} = \mathsf{m}_1$ に対して $\widetilde{f}_1 = f_{(\mathsf{m}_i)_{i\in I}}$ とする．

定義 9 (UF-GT-CCA/sUF-GT-CCA 安全性)**.** 次が成り立つとき，*GT-HAE* 方式 GTHAE = (KGen, Enc, Eval, Dec, GTctxt, GTident) は UF-GT-CCA 安全性を満たす：任意の *PPT* 攻撃者 A に対して，アドバンテージ

$$\mathsf{Adv}^{\text{uf-gt-cca}}_{\mathsf{GTHAE},\mathsf{A}}(\lambda) := \Pr\left[\mathsf{A}\ \textit{wins} \;\middle|\; \begin{array}{l}(\mathsf{sk},\mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda); \mathcal{Q} \leftarrow \emptyset;\\ (\mathcal{P}^*, \widehat{\mathsf{ct}}^*) \leftarrow \mathsf{A}^{\mathsf{ENC},\mathsf{DEC},\mathsf{GTIDENT}}(\mathsf{ek})\end{array}\right]$$

が λ において無視できるほど小さい．ここで，$\mathcal{P}^* = (f^*, \tau_1^*, \ldots, \tau_\ell^*)$ とし，次が成り立つ事象を [A *wins*] とする：

- $\mathsf{Dec}(\mathsf{sk}, \mathcal{P}^*, \widehat{\mathsf{ct}}^*) \neq \perp$，かつ
- 以下のいずれかが成り立つ：
  - $\widetilde{f}$ が *not-constant* である．
  - $\widetilde{f}$ が *constant* かつ $\mathsf{Dec}(\mathsf{sk}, \mathcal{P}^*, \widehat{\mathsf{ct}}^*) \neq \widetilde{f}$ である．
    ここで $\widetilde{f} = \widetilde{f}_{(\mathsf{m}_i)_{i\in I}}$，$I = \{i \in [\ell] \mid (\tau_i^*, \mathsf{m}_i, \mathsf{ct}_i) \in \mathcal{Q} \wedge \mathsf{m}_i \in \mathcal{M} \wedge \mathsf{ct}_i \in \mathcal{CT}\}$ とする．

さらに，sUF-GT-CCA 安全性は，事象 [A *wins*] を除いて UF-GT-CCA 安全性と同様に定義される．次が成り立つ事象を $\mathsf{Adv}^{\text{suf-gt-cca}}_{\mathsf{GTHAE},\mathsf{A}}(\lambda)$ における [A *wins*] とする：

- $\mathsf{Dec}(\mathsf{sk}, \mathcal{P}^*, \widehat{\mathsf{ct}}^*) \neq \perp$，かつ
- 以下のいずれかが成り立つ：
  - $\widetilde{\mathsf{ct}}$ が *not-constant* である．
  - $\widetilde{\mathsf{ct}}$ が *constant* かつ $\widehat{\mathsf{ct}}^* \neq \widetilde{\mathsf{ct}}$ である．
    ただし，$\widetilde{\mathsf{ct}} = \widetilde{\mathsf{ct}}_{f^*,(\mathsf{ct}_i)_{i\in I}}$，$I = \{i \in [\ell] \mid (\tau_i^*, \mathsf{m}_i, \mathsf{ct}_i) \in \mathcal{Q} \wedge \mathsf{m}_i \in \mathcal{M} \wedge \mathsf{ct}_i \in \mathcal{CT}\}$ とする．

Identifiablity として ident-completeness と ident-soundness を定義する．ident-completeness は全ての正当なラベルを GTident の出力 J に含めない安全性に対し，ident-soundness は不正なラベルを全て検出する安全性である ([4] を参考)．

定義 10 (Identifiability)**.** 次が成り立つとき，*GT-HAE* 方式 GTHAE = (KGen, Enc, Eval, Dec, GTctxt, GTident) は ident-completeness を満たす：任意の *PPT* 攻撃者 A に対して，アドバンテージ

$$\mathsf{Adv}^{\text{complete}}_{\mathsf{GTHAE},\mathsf{A}}(\lambda) := \Pr\left[\mathsf{A}\ \textit{wins} \;\middle|\; \begin{array}{l}(\mathsf{sk},\mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda); \mathcal{Q} \leftarrow \emptyset;\\ (\boldsymbol{G}^*, \mathcal{P}^*, \mathsf{ct}_1^*, \ldots, \mathsf{ct}_\ell^*) \leftarrow \mathsf{A}^{\mathsf{ENC},\mathsf{DEC},\mathsf{GTIDENT}}(\mathsf{ek})\end{array}\right]$$

が λ において無視できるほど小さい．ここで，[A *wins*] = $[\bar{D} \cap J \neq \perp]$ とし，$\bar{D}$ と J は以下のように定義される：

- $(\widehat{\mathsf{ct}}_1^*, \ldots, \widehat{\mathsf{ct}}_u^*) \leftarrow \mathsf{GTctxt}(\mathsf{ek}, \boldsymbol{G}^*, f^*, (\mathsf{ct}_1^*, \ldots, \mathsf{ct}_\ell^*))$，
- $J \leftarrow \mathsf{GTident}(\mathsf{sk}, \boldsymbol{G}^*, \mathcal{P}^*, (\widehat{\mathsf{ct}}_1^*, \ldots, \widehat{\mathsf{ct}}_u^*))$，
- $D = \{\tau_i^* \mid i \in [\ell] \wedge \mathsf{Dec}(\mathsf{sk}, \mathcal{I}_{\tau_i^*}, \mathsf{ct}_i^*) = \perp\}$，
- $\bar{D} = \{\tau_i^* \mid i \in [\ell] \wedge \mathsf{Dec}(\mathsf{sk}, \mathcal{I}_{\tau_i^*}, \mathsf{ct}_i^*) \neq \perp\}$．

また，任意の *PPT* 攻撃者 A に対して $\mathsf{Adv}^{\text{sound}}_{\mathsf{GTHAE},\mathsf{A}}(\lambda) \leq \mathsf{negl}(\lambda)$ であるとき GTHAE は ident-soundness を満たす：$\mathsf{Adv}^{\text{sound}}_{\mathsf{GTHAE},\mathsf{A}}(\lambda)$ は，[A *wins*] = $[D \backslash J \neq \perp]$ であることを除いて前述の $\mathsf{Adv}^{\text{complete}}_{\mathsf{GTHAE},\mathsf{A}}(\lambda)$ と同様に定義される．

さらに，ident-soundness の弱い安全性 ident-weak-soundness は，$D = \{\tau_i^* \mid i \in [\ell] \wedge \mathsf{Dec}(\mathsf{sk}, \mathcal{I}_{\tau_i^*}, \mathsf{ct}_i^*) = \perp \wedge \mathsf{E}_i\}$ であることを除いて ident-soundness と同様に定義される．ここで，E_i を次が成り立つ事象とする：ある $j \in [u]$ が存在して $i \in S_j(\boldsymbol{G})$ であり，任意の $\overline{\mathsf{ct}} \in \mathsf{Enc}(\mathsf{sk}, \mathcal{T}, \mathcal{M})$ に対して以下のいずれかが成り立つ：

- $\widetilde{\mathsf{ct}}_j$ が *not-constant* である．
- $\widetilde{\mathsf{ct}}_j$ が *constant* かつ $\widetilde{\mathsf{ct}}_j \neq \widehat{\mathsf{ct}}_j^*$．

$\widetilde{\mathsf{ct}}_j = \widetilde{\mathsf{ct}}_{f,(\mathsf{ct}_k)_{k\in I}}((\overline{\mathsf{ct}})_{k\notin S_j(\boldsymbol{G})})$，$I = \{i \in [\ell] \mid (\tau_i^*, \mathsf{m}_i, \mathsf{ct}_i) \in \mathcal{Q} \wedge \mathsf{m}_i \in \mathcal{M} \wedge \mathsf{ct}_i \in \mathcal{CT}\}$ とする．

4. GT-HAE の一般的構成

4.1 提案方式 1

この構成では，HAE 方式 HAE = (KGen^{hae}, Enc^{hae}, Eval^{hae}, Dec^{hae}) と d-disjunct 行列を用いる．GT プロトコルを GTctxt と GTident に適用することによってメッセージが改ざんされた暗号文を検出することが可能である．

提案方式 GTHAE_1 = (KGen, Enc, Eval, Dec, GTctxt, GTident) は次の通りである：セキュリティパラメータ λ に対して，ラベル $\tau_c \in \mathcal{T}(\lambda)$ と $\mathsf{m}_c \in \mathcal{M}(\lambda)$ を適切に選び，ラベル空間を $\mathcal{T} = \mathcal{T}(\lambda)\backslash\{\tau_c\}$，メッセージ空間を $\mathcal{M} = \mathcal{M}(\lambda)\backslash\{\mathsf{m}_c\}$ とする．また，$\boldsymbol{G} \in \{0,1\}^{u\times\ell}$ と $i \in [u]$ に対して，$S_i(\boldsymbol{G}) = \{j \mid j \in [\ell] \wedge g_{i,j} = 1\}$ とする．$\boldsymbol{G} \in \{0,1\}^{u\times\ell}$ と $i \in [u]$ に対する labeled-program を $\mathcal{P}_i = (f, \tau_1', \ldots, \tau_\ell')$ とする ($k \in [\ell]$ に対して，$k \in S_i(\boldsymbol{G})$ ならば $\tau_k' \leftarrow \tau_i$，$k \notin S_i(\boldsymbol{G})$ ならば $\tau_i' \leftarrow \tau_c$ とする)．

- $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda)$:
  (1) $(\mathsf{sk}^{hae}, \mathsf{ek}^{hae}) \leftarrow \mathsf{KGen}^{hae}(1^\lambda)$.
  (2) $\mathsf{ct}_c \leftarrow \mathsf{Enc}^{hae}(\mathsf{sk}, \tau_c, \mathsf{m}_c)$.
  (3) $\mathsf{sk} = \mathsf{sk}^{hae}$ と $\mathsf{ek} = (\mathsf{ek}^{hae}, \mathsf{ct}_c)$ を出力する．
- $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau, \mathsf{m})$:
  $\mathsf{ct} \leftarrow \mathsf{Enc}^{hae}(\mathsf{sk}^{hae}, \tau, \mathsf{m})$ を出力する．
- $\widehat{\mathsf{ct}} \leftarrow \mathsf{Eval}(\mathsf{ek}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$:
  $\widehat{\mathsf{ct}} \leftarrow \mathsf{Eval}^{hae}(\mathsf{ek}^{hae}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$ を出力する．
- $\mathsf{m}/\perp \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathcal{P}, \widehat{\mathsf{ct}})$:
  $\mathsf{m}/\perp \leftarrow \mathsf{Dec}^{hae}(\mathsf{sk}^{hae}, \mathcal{P}, \widehat{\mathsf{ct}})$ を出力する．
- $(\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u) \leftarrow \mathsf{GTctxt}(\mathsf{ek}, \boldsymbol{G}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$:

(1) 各 $i \in [u]$ に対して, 以下の処理を行う:
  (a) 関数 $\widetilde{\mathsf{ct}}_i = \widetilde{\mathsf{ct}}_{f,(\mathsf{ct}_k)_{k\in S_i(\boldsymbol{G})}}$ を生成する.
  (b) 各 $k \notin S_i(\boldsymbol{G})$ に対して, $\mathsf{ct}_k \leftarrow \mathsf{ct}_c$.
  (c) $\widehat{\mathsf{ct}}_i \leftarrow \widetilde{\mathsf{ct}}_i((\mathsf{ct}_k)_{k\notin S_i(\boldsymbol{G})})$.
(2) $(\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u)$ を出力する.

- $J \leftarrow \mathsf{GTident}(\mathsf{sk}, \boldsymbol{G}, \mathcal{P}, (\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u))$:
  (1) $J \leftarrow \{\tau_1, \ldots, \tau_\ell\}$.
  (2) 各 $i \in [u]$ に対して, $\mathsf{Dec}^{hae}(\mathsf{sk}^{hae}, \mathcal{P}_i, \widehat{\mathsf{ct}}_i) \neq \perp$ ならば, $J \leftarrow J \backslash \{\tau_k\}_{k\in S_i(\boldsymbol{G})}$ とする.
  (3) J を出力する.

HAE が correctness, compactness, CCT を満たすならば, GTHAE_1 もこれらを満たすことは明らかである. GTHAE_1 の安全性について定理 1, 2, 3 が成り立つ.

定理 1. HAE が IND-CPA 安全性と sUF-CPA 安全性を満たすならば, 提案方式 GTHAE_1 は IND-GT-CCA 安全性を満たす.

定理 2. HAE が sUF-CPA 安全性を満たすならば, 提案方式 GTHAE_1 は sUF-GT-CCA 安全性を満たす.

定理 3. 提案方式 GTHAE_1 は以下の identifiability を満たす:

- $\boldsymbol{G}$ が *d-disjunct* 行列ならば, GTHAE_1 は ident-completeness を満たす.
- $\boldsymbol{G}$ が *d-disjunct* 行列であり HAE が sUF-CPA 安全性を満たすならば, GTHAE_1 は ident-weak-soundness を満たす.

これらの証明において, A を GTHAE_1 に対する PPT 攻撃者, q を DEC と GTIDENT に対して発行されたクエリの総数とする.

4.1.1 定理 1 の証明

この証明では [5] の CPA 安全性から CCA 安全性への帰着証明の手法を利用する. 次の安全性ゲームを考える: Game_0 を通常の IND-GT-CCA 安全性ゲーム, Game_1 を Game_0 において DEC を DEC_1 に変更した安全性ゲームとする. DEC_1 は復号クエリ $(\mathcal{P}, \widehat{\mathsf{ct}})$ を受け取り, 以下の処理を行う:

(1) $\mathcal{P}$ が illegal ならば $\perp$ を返す.
(2) $I \leftarrow \emptyset$ とし, 各 $i \in [\ell]$ に対して $(\tau_i, \mathsf{m}, \mathsf{ct}) \in \mathcal{Q}$ ならば, $I \leftarrow I \cup \{i\}$, $\mathsf{m}_i \leftarrow \mathsf{m}$, $\mathsf{ct}_i \leftarrow \mathsf{ct}$ とする.
(3) $\widetilde{\mathsf{ct}} \leftarrow \widetilde{\mathsf{ct}}_{f,(\mathsf{ct}_i)_{i\in I}}$, $\tilde{f} \leftarrow \tilde{f}_{(\mathsf{m}_i)_{i\in I}}$ とする.
(4) $\widetilde{\mathsf{ct}}$ が constant かつ $\widetilde{\mathsf{ct}} = \widehat{\mathsf{ct}}$ ならば, $\tilde{f}$ を返す. そうでなければ $\perp$ を返す.

Game_0 と Game_1 の識別不可能性を示す. DEC_1 の処理において以下のいずれかの事象が起こらなければ, Game_1 は Game_0 と等価である:

Bad_1: $(\widetilde{\mathsf{ct}}$ が constant かつ $\widetilde{\mathsf{ct}} = \widehat{\mathsf{ct}}) \wedge \mathsf{Dec}(\mathsf{sk}, \mathcal{P}, \widehat{\mathsf{ct}}) = \perp$.

Bad_2: $(\widetilde{\mathsf{ct}}$ が not-constant, または $\widetilde{\mathsf{ct}}$ が constant かつ $\widetilde{\mathsf{ct}} = \widehat{\mathsf{ct}}) \wedge \mathsf{Dec}(\mathsf{sk}, \mathcal{P}, \widehat{\mathsf{ct}}) \neq \perp$.

Bad_1 について, $\widetilde{\mathsf{ct}}$ が constant かつ $\widetilde{\mathsf{ct}} = \widehat{\mathsf{ct}}$ のとき, HAE の correctness により $\mathsf{Dec}(\mathsf{sk}, \mathcal{P}, \widehat{\mathsf{ct}}) = \tilde{f}$ であり Bad_1 が起こる確率は無視できるほど小さい. よって, Bad_2 が起こるときのみ, DEC_1 のシミュレーションは失敗する. Bad_2 におけるクエリ $(\mathcal{P}, \widehat{\mathsf{ct}})$ は明らかに GTHAE_1 の偽造であり, sUF-CPA 安全性を破る PPT アルゴリズム $\mathsf{F}_1^{suf(i)}$ が次のように構成される ($i \in [q]$): 評価鍵 ek^{hae} を入力とする. $(\tau_c, \mathsf{m}_c) \in \mathcal{T}(\lambda) \times \mathcal{M}(\lambda)$ を選び, sUF-CPA 安全性ゲームにおける暗号化オラクル ENC^{cpa} に発行して ct_c を得る. $\mathcal{Q} \leftarrow \{(\mathsf{m}_c, \tau_c, \mathsf{ct}_c)\}$ とし, A に $\mathsf{ek} = (\mathsf{ek}^{hae}, \mathsf{ct}_c)$ を与える. オラクル ENC, DEC, GTIDENT を以下のようにシミュレートする:

- ENC: 暗号化クエリ (τ, m) を受け取ったら, (τ, m) を ENC^{cpa} に発行して ct を得る. ct を返し, $\mathsf{ct} \neq \perp$ ならば $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\tau, \mathsf{m}, \mathsf{ct})\}$ とする.
- DEC: i 番目までのクエリに対して DEC_1 の処理を行う. i 番目のクエリ $(\mathcal{P}, \widehat{\mathsf{ct}})$ において, Bad_2 の条件を満たす偽造として $(\mathcal{P}, \widehat{\mathsf{ct}})$ を出力して停止する.
- GTIDENT: GT クエリ $(\boldsymbol{G}, \mathcal{P}, (\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u))$ を受け取る. $J \leftarrow \{\tau_1, \ldots, \tau_\ell\}$ とし, 各 $i \in [u]$ に対して, 上記の DEC オラクルに対して $(\mathcal{P}_i, \widehat{\mathsf{ct}}_i)$ を発行し, その応答 m_i を受け取る. $\mathsf{m}_i \neq \perp$ ならば $J \leftarrow J \backslash \{\tau_k\}_{k\in S_i(\boldsymbol{G})}$ とする. J を返す.

また, A が $(\tau^*, \mathsf{m}_0, \mathsf{m}_1)$ を発行したら, 以下の処理を行う:

(1) $(\tau^*, \cdot, \cdot) \in \mathcal{Q}$ ならば拒否する.
(2) (τ^*, m_b) を ENC^{cpa} に発行して ct^* を得る ($b \xleftarrow{\$} \{0,1\}$).
(3) ct^* を返し, $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\tau, \mathsf{m}_b, \mathsf{ct}^*)\}$ とする.

$\mathsf{F}_1^{suf(i)}$ の出力が GTHAE_1 の偽造であることは前述の通りである. よって, $i \in [q]$ 上の union-bound により 2 つのゲームを識別できる確率は高々 $q \cdot \mathsf{Adv}^{\text{suf-cpa}}_{\mathsf{HAE},\mathsf{F}_1^{suf}}(\lambda)$ である.

次に, Game_1 において $|\Pr[b = b'] - 1/2| \leq \mathsf{Adv}^{\text{ind-cpa}}_{\mathsf{HAE},\mathsf{D}_1^{cpa}}(\lambda)$ であることを示す. IND-CPA 安全性を破る PPT アルゴリズム D_1^{cpa} を構成する. D_1^{cpa} は, 以下の点を除いて $\mathsf{F}_1^{suf(q)}$ と同様である:

- A が $(\tau^*, \mathsf{m}_0, \mathsf{m}_1)$ を発行したら, challenge として, これを発行して ct^* を受け取る. ct^* を A に返し, $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\tau^*, \mathsf{m}_0, \mathsf{ct}^*)\}$ とする.
- A が $b' \in \{0,1\}$ を出力したら D_1^{cpa} も b' を出力する.

Challenge の処理について, 本来の Game_1 では $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\tau^*, \mathsf{m}_b, \mathsf{ct}^*)\}$ とするのに対して, D_1^{cpa} では $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\tau^*, \mathsf{m}_0, \mathsf{ct}^*)\}$ としている. (τ^*, ct^*) を含む準同型演算が施された復号クエリ $(\mathcal{P}', \widehat{\mathsf{ct}}')$ が発行されたとき, シミュレーションは失敗する. しかし, このような $\mathcal{P}'$ は illegal であるため, $(\mathcal{P}', \widehat{\mathsf{ct}}')$ の発行は禁止されている. よって, D_1^{cpa} は Game_1 を正しくシミュレートしている. また, A が $b = b'$ を満たす b' を出力したら D_1^{cpa} も IND-CPA 安全性ゲームにおいて勝利することは明らかである.

以上により, $\mathsf{Adv}^{\text{ind-gt-cca}}_{\mathsf{GTHAE}_1,\mathsf{A}}(\lambda) \leq q \cdot \mathsf{Adv}^{\text{suf-cpa}}_{\mathsf{HAE},\mathsf{F}_1^{suf}}(\lambda) + \mathsf{Adv}^{\text{ind-cpa}}_{\mathsf{HAE},\mathsf{D}_1^{cpa}}(\lambda)$ が得られる. □

4.1.2 定理 2 の証明

定理 1 の証明と同様にして, DEC を DEC_1 に変更する安全性ゲームを考える. このとき, $i \in [q]$ に対して, challenge の処理を除いて $\mathsf{F}_1^{suf(i)}$ と同様にして sUF-CPA 安全性を破る PPT アルゴリズム $\mathsf{F}_2^{suf(i)}$ が構成される. 変更されたゲームで sUF-CPA 安全性に帰着できることは明らかである. $i \in [q]$ の union-bound により, $\mathsf{Adv}^{\text{suf-gt-cca}}_{\mathsf{GTHAE}_1,\mathsf{A}}(\lambda) \leq (q+1) \cdot \mathsf{Adv}^{\text{suf-cpa}}_{\mathsf{HAE},\mathsf{F}_2^{suf}}(\lambda)$ を得る. □

4.1.3 定理 3 の証明

GTHAE_1 が ident-completeness を満たすことを示す. A の出力を $(\boldsymbol{G}^*, (f^*, \tau_1^*, \ldots, \tau_\ell^*), (\mathsf{ct}_1^*, \ldots, \mathsf{ct}_\ell^*))$ とする. $\mathsf{Dec}(\mathsf{sk}, \mathcal{I}_{\tau_i^*}, \mathsf{ct}_i^*) \neq \perp$ かつ $(\tau_i^*, \mathsf{ct}_i^*) \in J$ を満たす $(\tau_i^*, \mathsf{ct}_i^*)$ を考える $(i \in [\ell])$. $(\tau_i^*, \mathsf{ct}_i^*) \in J$ により, 全ての $i \in S_j(\boldsymbol{G}^*)$ を満たす $j \in [u]$ において, $\mathsf{Dec}(\mathsf{sk}, \mathcal{P}_j^*, \widehat{\mathsf{ct}}_j^*) = \perp$ である. しかし, $\boldsymbol{G}^*$ の d-disjunct 性と HAE の correctness により, $\mathsf{Dec}(\mathsf{sk}, \mathcal{P}_j^*, \widehat{\mathsf{ct}}_j^*) = \perp$ である確率は無視できるほど小さい. よって, $\mathsf{Adv}^{\text{complete}}_{\mathsf{GTHAE}_1,\mathsf{A}}(\lambda) \leq \mathsf{negl}(\lambda)$ が得られる.

GTHAE_1 が ident-weak-soundness を満たすことを示す. DEC を定理 1 の証明における DEC_1 に変更する安全性ゲームを考える. この変更は sUF-CPA 安全性により識別不可能であることは定理 1, 2 の証明で示した通りである. この変更において sUF-CPA 安全性を破る PPT アルゴリズム $\mathsf{F}_3^{suf(j)}$ を次のように構成する $(j \in [u])$: 定理 1 の証明における $\mathsf{F}_1^{suf(q)}$ と同様にしてオラクルをシミュレートする. A が $(\boldsymbol{G}^*, (f^*, \tau_1^*, \ldots, \tau_\ell^*), (\mathsf{ct}_1^*, \ldots, \mathsf{ct}_\ell^*))$ を出力したとき, 定義に従って $(\widehat{\mathsf{ct}}_1^*, \ldots, \widehat{\mathsf{ct}}_u^*)$ を計算し, $(\mathcal{P}_j^*, \widehat{\mathsf{ct}}_j^*)$ を出力する. このとき, A の勝利条件により, ある $i \in [\ell]$ が存在して $i \in S_j(\boldsymbol{G}^*)$, $\mathsf{Dec}(\mathsf{sk}, \mathcal{I}_{\tau_i^*}, \mathsf{ct}_i^*) = \perp$, $\tau_i^* \notin J$ であり, $\widetilde{\mathsf{ct}}_j$ について以下のいずれかが成り立つ:

- $\widetilde{\mathsf{ct}}_j$ が not-constant である.
- $\widetilde{\mathsf{ct}}_j$ が constant かつ $\widetilde{\mathsf{ct}}_j \neq \widehat{\mathsf{ct}}_j^*$ である.

また, $\tau_i^* \notin J$ により $\mathsf{Dec}^{hae}(\mathsf{sk}^{hae}, \mathcal{P}_j^*, \widehat{\mathsf{ct}}_j^*) \neq \perp$ であり, $(\mathcal{P}_j^*, \widehat{\mathsf{ct}}_j^*)$ は sUF-CPA 安全性ゲームにおける偽造である.

以上により, $\mathsf{Adv}^{\text{w-sound}}_{\mathsf{GTHAE}_1,\mathsf{A}}(\lambda) \leq (u+q) \cdot \mathsf{Adv}^{\text{suf-cpa}}_{\mathsf{GTHAE}_1,\mathsf{F}_3^{suf}}(\lambda)$ を得る. □

4.2 提案方式 2

ident-soundness を満たす一般的構成 GTHAE_2 を示す. この構成では, HAE 方式 $\mathsf{HAE} = (\mathsf{KGen}^{hae}, \mathsf{Enc}^{hae}, \mathsf{Dec}^{hae}, \mathsf{Eval}^{hae})$, 以下の関係 $(R_{enc} \vee R_{eval})$ に対する公開検証可能な zk-SNARG $\Pi_{\mathsf{SNARG}} = (\mathsf{Gen}, \mathsf{P}, \mathsf{V})$ を利用する.

- 関係 R_{enc}:
  - Identity labeled-program $\mathcal{I}_\tau$ と HAE の暗号文 ct^{hae} の組をステートメントとする.
  - HAE の秘密鍵 sk^{hae}, メッセージ m, Enc^{hae} による m の暗号化で用いられる乱数 r の組を証拠とする.
  - $\mathsf{ct}^{hae} = \mathsf{Enc}^{hae}(\mathsf{sk}^{hae}, \tau, \mathsf{m}; r)$ が成り立つとき関係 R_{enc} を満たす.
- 関係 R_{eval}:
  - Labeled-program $\mathcal{P}$ と Eval^{hae} で生成された暗号文 $\widehat{\mathsf{ct}}^{hae}$ の組をステートメントとする.
  - 暗号文の組 $(\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$ を証拠する ($i \in [\ell]$ に対して $\mathsf{ct}_i = (\mathcal{I}_{\tau_i}, \mathsf{ct}_i^{hae}, \pi_i)$, π_i は R_{enc} に対する証明).
  - $\widehat{\mathsf{ct}}^{hae} = \mathsf{Eval}^{hae}(\mathsf{ek}^{hae}, f, (\mathsf{ct}_1^{hae}, \ldots, \mathsf{ct}_\ell^{hae}))$ かつ, 全ての $i \in [\ell]$ に対して $\mathsf{V}(\mathsf{crs}, (\mathcal{I}_{\tau_i}, \mathsf{ct}_i^{hae}), \pi_i) = 1$ が成り立つとき関係 R_{gt} を満たす.

(GTctxt, GTident) に対して上記の Π_{SNARG} を適用することによって, $\mathsf{V}(\mathsf{crs}, (\mathcal{I}_{\tau_i}, \mathsf{ct}_i^{hae}), \pi_i) = 0$ である不正な暗号文 $(\mathcal{I}_{\tau_i}, \mathsf{ct}_i^{hae}, \pi_i)$ が存在すれば, Π_{SNARG} の soundness により検出される, というのが ident-soundness を達成するためのアイディアである. また, GT-HAE の compactness を達成するために Π_{SNARG} の succinctness が要求される.

提案方式 $\mathsf{GTHAE}_2 = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{GTctxt}, \mathsf{GTident})$ は次のように構成される: $\mathcal{T} = \mathcal{T}(\lambda) \backslash \{\tau_c\}$, $\mathcal{M} = \mathcal{M}(\lambda) \backslash \{\mathsf{m}_c\}$, $S_i(\boldsymbol{G})$, $\mathcal{P}_i = (f, \tau_1', \ldots, \tau_\ell')$ を GTHAE_1 と同様に定義する.

- $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{KGen}(1^\lambda)$:
  (1) $(\mathsf{sk}^{hae}, \mathsf{ek}^{hae}) \leftarrow \mathsf{KGen}^{hae}(1^\lambda)$, $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)$.
  (2) $\mathsf{ct}_c^{hae} \leftarrow \mathsf{Enc}^{hae}(\mathsf{sk}^{hae}, \tau_c, \mathsf{m}_c)$.
  (3) $\mathsf{sk} = (\mathsf{sk}^{hae}, \mathsf{crs})$ と $\mathsf{ek} = (\mathsf{ek}^{hae}, \mathsf{crs}, \mathsf{ct}_c^{hae})$ を出力する.
- $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, \tau, \mathsf{m})$:
  (1) $\mathsf{ct}^{hae} \leftarrow \mathsf{Enc}^{hae}(\mathsf{sk}^{hae}, \tau, \mathsf{m}; r)$.
  (2) $\pi \leftarrow \mathsf{P}(\mathsf{crs}, (\mathcal{I}_\tau, \mathsf{ct}^{hae}), (\mathsf{sk}^{hae}, \mathsf{m}, r))$.
  (3) $\mathsf{ct} = (\mathcal{I}_\tau, \mathsf{ct}^{hae}, \pi)$ を出力する.
- $\widehat{\mathsf{ct}} \leftarrow \mathsf{Eval}(\mathsf{ek}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$: $\mathsf{ct}_i = (\mathcal{I}_{\tau_i}, \mathsf{ct}_i^{hae}, \pi_i)$ $(i \in [\ell])$, $\mathcal{P} = (f, \tau_1, \ldots, \tau_\ell)$ として以下の処理を行う:
  (1) $\widehat{\mathsf{ct}}^{hae} \leftarrow \mathsf{Eval}^{hae}(\mathsf{ek}^{hae}, f, (\mathsf{ct}_1^{hae}, \ldots, \mathsf{ct}_\ell^{hae}))$.
  (2) $\widehat{\pi} \leftarrow \mathsf{P}(\mathsf{crs}, (\mathcal{P}, \widehat{\mathsf{ct}}^{hae}), (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$.
  (3) $\widehat{\mathsf{ct}} = (\mathcal{P}, \widehat{\mathsf{ct}}^{hae}, \widehat{\pi})$ を出力する.
- $\mathsf{m}/\perp \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathcal{P}, \widehat{\mathsf{ct}})$: $\widehat{\mathsf{ct}} = (\mathcal{P}, \widehat{\mathsf{ct}}^{hae}, \widehat{\pi})$ に対して, $\mathsf{V}(\mathsf{crs}, (\mathcal{P}, \widehat{\mathsf{ct}}^{hae}), \widehat{\pi}) = 1$ ならば $\mathsf{m}/\perp \leftarrow \mathsf{Dec}^{hae}(\mathsf{sk}^{hae}, \mathcal{P}, \widehat{\mathsf{ct}}^{hae})$ を出力する. そうでなければ $\perp$ を出力する.
- $(\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u) \leftarrow \mathsf{GTctxt}(\mathsf{ek}, \boldsymbol{G}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$:
  (1) 各 $i \in [u]$ に対して, 次の処理を行う:
    (a) $\widehat{\mathsf{ct}}_i^{hae} \leftarrow \widetilde{\mathsf{ct}}_{f, (\mathsf{ct}_k^{hae})_{k \in S_i(\boldsymbol{G})}}((\mathsf{ct}_c^{hae})_{k \notin S_i(\boldsymbol{G})})$.
    (b) $k \in [\ell]$ に対して, $k \in S_i(\boldsymbol{G})$ ならば $\mathsf{ct}_k' \leftarrow \mathsf{ct}_k$, $k \notin S_i(\boldsymbol{G})$ ならば $\mathsf{ct}_k' \leftarrow \mathsf{ct}_c$ とする.
    (c) $\widehat{\pi}_i \leftarrow \mathsf{P}(\mathsf{crs}, (\mathcal{P}_i, \widehat{\mathsf{ct}}_i^{hae}), (\mathsf{ct}_1', \ldots, \mathsf{ct}_\ell'))$.
    (d) $\widehat{\mathsf{ct}}_i = (\mathcal{P}_i, \widehat{\mathsf{ct}}_i^{hae}, \widehat{\pi}_i)$ とする.
  (2) $(\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u)$ を出力する.
- $J \leftarrow \mathsf{GTident}(\mathsf{sk}, \boldsymbol{G}, \mathcal{P}, (\widehat{\mathsf{ct}}_1, \ldots, \widehat{\mathsf{ct}}_u))$:
  (1) $J \leftarrow \{\tau_1, \ldots, \tau_\ell\}$.
  (2) 各 $i \in [u]$ に対して, $\mathsf{V}(\mathsf{crs}, (\mathcal{P}_i, \widehat{\mathsf{ct}}_i^{hae}), \widehat{\pi}_i) = 1$ かつ $\mathsf{Dec}^{hae}(\mathsf{sk}^{hae}, \mathcal{P}_i, \widehat{\mathsf{ct}}_i^{hae}) \neq \perp$ ならば, $J \leftarrow$

$J\backslash\{\tau_k\}_{k\in S_i(\boldsymbol{G})}$ とする.

(3) J を出力する.

GTHAE_2 の安全性について定理 4, 5, 6 が成り立つ. ページの都合上, これらの証明は概要のみ記述する.

定理 4. HAE が IND-CPA 安全性と sUF-CPA 安全性を満たし Π_{SNARG} が zero-knowledge を満たすならば, 提案方式 GTHAE_2 は IND-GT-CCA 安全性を満たす.

定理 5. HAE が sUF-CPA 安全性を満たし Π_{SNARG} が zero-knowledge を満たすならば, 提案方式 GTHAE_2 は UF-GT-CCA 安全性を満たす.

定理 6. 提案方式 GTHAE_2 は次の identifiability を満たす:

- $\boldsymbol{G}$ が *d-disjunct* 行列ならば, GTHAE_2 は ident-completeness を満たす.
- $\boldsymbol{G}$ が *d-disjunct* 行列であり Π_{SNARG} が soundness を満たすならば, GTHAE_2 は ident-soundness を満たす.

4.2.1 定理 4 の証明の概要

A を GTHAE_2 に対する PPT 攻撃者とする. 以下の安全性ゲームを考える:

- Game_0: 通常の IND-GT-CCA 安全性ゲーム.
- Game_1: Game_0 において, ct^* の生成において Π_{SNARG} のシミュレータ $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ を用いて証明 π^* を計算するように変更した安全性ゲーム.
- Game_2: Game_1 において, DEC オラクルを次の DEC_1 に変更した安全性ゲーム:
  (1) 復号クエリ $(\mathcal{P}, \widehat{\mathsf{ct}})$ を受け取る $(\widehat{\mathsf{ct}} = (\mathcal{P}, \widehat{\mathsf{ct}}^{hae}, \widehat{\pi}))$.
  (2) $I \leftarrow \emptyset$ とし, 各 $i \in [\ell]$ に対して, $(\tau_i, \mathsf{m}, \mathsf{ct}) \in \mathcal{Q}$ ならば, $I \leftarrow I \cup \{i\}$, $\mathsf{m}_i \leftarrow \mathsf{m}$, $\mathsf{ct}_i \leftarrow \mathsf{ct}$ とする.
  (3) $\widetilde{\mathsf{ct}}^{hae} \leftarrow \widetilde{\mathsf{ct}}_{f,(\mathsf{ct}_i^{hae})_{i\in I}}$, $\widetilde{f} \leftarrow \widetilde{f}_{(\mathsf{m}_i)_{i\in I}}$ とする.
  (4) $\widehat{\mathsf{ct}}^{hae} = \widetilde{\mathsf{ct}}^{hae}$ かつ $\mathsf{V}(\mathsf{crs}, (\mathcal{P}, \widehat{\mathsf{ct}}^{hae}), \widehat{\pi}) = 1$ ならば $\widetilde{f}$ を返す. そうでなければ $\perp$ を返す.

Game_0 と Game_1 の識別不可能性は Π_{SNARG} の zero-knowledge によって保証される. このとき, ct^* の生成において Sim_1 に対して言語 $L(R_{enc} \vee R_{eval})$ に含まれない $(\mathcal{I}_{\tau^*}, \mathsf{ct}^{hae*})$ を入力とする確率は, HAE の correctness により無視できるほど小さいことに注意する.

Game_1 と Game_2 の識別不可能性は HAE の sUF-CPA 安全性によって保証される. なぜなら, 定理 1 の証明における Game_0 と Game_1 の識別不可能性の解析と同様にして, Π_{SNARG} の処理を追加する点を除いて sUF-CPA 安全性を破る PPT アルゴリズムを構成できるためである.

Game_2 において $|\Pr[b = b'] - 1/2| \leq \mathsf{Adv}^{\text{ind-cpa}}_{\mathsf{HAE},\mathsf{D}_2^{cpa}}(\lambda)$ が成り立つ. これは, Π_{SNARG} の処理を追加する点を除いて定理 1 の証明の D_1^{cpa} と同様にして, HAE の IND-CPA 安全性を破る PPT アルゴリズム D_2^{cpa} を構成できるためである.

以上により, 定理 4 が示された. □

4.2.2 定理 5 の証明の概要

UF-GT-CCA 安全性ゲームにおいてアルゴリズム P の代わりに Π_{SNARG} のシミュレータ Sim を用いるゲームに変更する. この変更は Π_{SNARG} の zero-knowledge により保証される. 変更した安全性ゲームにおいて, Π_{SNARG} の処理を追加する点を除いて定理 2 の証明と同様にして sUF-CPA 安全性を破る PPT アルゴリズムが構成される. [5] の定理 2 と本稿の定理 2 の証明により, UF-GT-CCA 安全性が破られれば, sUF-CPA 安全性も破られる. □

4.2.3 定理 6 の証明の概要

定理 3 の証明と同様にして GTHAE_2 は ident-completeness を満たすことが示される.

GTHAE_2 が ident-soundness を満たすことを示す. UF-GT-CCA 安全性ゲームにおいて, GTident の処理で V が受理するならば $J \leftarrow J\backslash\{\tau_k\}_{k\in S_j(\boldsymbol{G})}$ $(j \in [u])$ とする安全性ゲームに変更する. このとき, $\mathsf{V}(\mathsf{crs}, (\mathcal{P}_i, \widehat{\mathsf{ct}}_i^{hae}), \widehat{\pi}_i) = 1$ かつ $\mathsf{Dec}^{hae}(\mathsf{sk}^{hae}, \mathcal{P}_i, \widehat{\mathsf{ct}}_i^{hae}) = \perp$ が起こらなければ, この変更は識別されない. この事象が起こる確率は Π_{SNARG} の soundness により無視できるほど小さい.

この安全性ゲームにおいて, Π_{SNARG} の soundness を破る PPT アルゴリズム F^{snd} は次のように構成される: crs を入力とし, $(\mathsf{sk}, \mathsf{ek})$ を生成する. 生成した sk を用いてオラクルをシミュレートする. A が $(\boldsymbol{G}^*, \mathcal{P}^*, (\mathsf{ct}_1^*, \ldots, \mathsf{ct}_\ell^*))$ を出力したら, $(\widehat{\mathsf{ct}}_1^*, \ldots, \widehat{\mathsf{ct}}_u^*)$ と J を計算する. A の勝利条件により, ある $(i, j) \in [\ell] \times [u]$ が存在して $i \in S_j(\boldsymbol{G}^*)$, $\mathsf{Dec}^{hae}(\mathsf{sk}^{hae}, \mathcal{I}_{\tau_i^*}, \mathsf{ct}_i^*) = 0$, $\mathsf{V}(\mathsf{crs}, (\mathcal{P}_j^*, \widehat{\mathsf{ct}}_j^{hae*}), \widehat{\pi}_j^*) = 1$ が成り立つ. F^{snd} は $((\mathcal{P}_j^*, \widehat{\mathsf{ct}}_j^{hae*}), \widehat{\pi}_j^*)$ を出力する. この出力は言語 $L(R_{enc} \vee R_{eval})$ に明らかに属さないにもかかわらず V によって受理されるため, soundness が破られる. □

参考文献

[1] M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *J. Cryptology*, 21(4):469–491, 2008.

[2] R. Gennaro and D. Wichs. Fully homomorphic message authenticators. In *ASIACRYPT (2)*, volume 8270 of *LNCS*, pages 301–320. Springer, 2013.

[3] G. Hartung, B. Kaidel, A. Koch, J. Koch, and A. Rupp. Fault-tolerant aggregate signatures. In *Public Key Cryptography (1)*, volume 9614 of *LNCS*, pages 331–356. Springer, 2016.

[4] S. Hirose and J. Shikata. Non-adaptive group-testing aggregate MAC scheme. In *ISPEC*, volume 11125 of *LNCS*, pages 357–372. Springer, 2018.

[5] C. Joo and A. Yun. Homomorphic authenticated encryption secure against chosen-ciphertext attack. In *ASIACRYPT (2)*, volume 8874 of *LNCS*, pages 173–192. Springer, 2014.

[6] K. Minematsu. Efficient message authentication codes with combinatorial group testing. In *ESORICS (1)*, volume 9326 of *LNCS*, pages 185–202. Springer, 2015.

[7] K. Minematsu and N. Kamiya. Symmetric-key corruption detection: When xor-macs meet combinatorial group testing. In *ESORICS (1)*, volume 11735 of *LNCS*, pages 595–615. Springer, 2019.