属性ベース認証方式の準適応的中間者攻撃に対する安全性の提案とGarbled Circuitsを用いた一般的構成

知念 広太郎[1,a)]　穴田 啓晃[2,b)]

概要：属性ベース認証が Anada ら (ACISP2013) により提案されている．また，属性ベース暗号の準適応的攻撃モデルの安全性が Chen-Wee ら (SCN2014) により提案されている．本稿で我々は，属性ベース認証に対する準適応的中間者攻撃モデルの安全性を提案する．我々は準適応的攻撃に対し安全な属性ベース暗号をチャレンジ＆レスポンスに用いた属性ベース認証が準適応的中間者攻撃に対し安全であることを証明する．一方，Goyal ら（TCC2016-B）が提案した，選択的攻撃に対し安全な関数型暗号から準適応的攻撃に対し安全な関数型暗号への Garbled Circuits と公開鍵暗号を用いた一般的変換を使用し，選択的攻撃に対し安全な属性ベース鍵カプセル化機構から準適応的中間者攻撃に対し安全な属性ベース認証への一般的構成を書き下す．最後に，Garbled Circuits を用いない準適応的中間者攻撃に対し安全な属性ベース認証との性能比較について述べる．

キーワード：属性ベース認証，準適応的攻撃安全性，Garbled Circuits

Proposal of Security of Attribute-Based Authentication Schemes against Semi-adaptive Man-in-the-Middle Attacks and Generic Construction Using Garbled Circuits

Kotaro Chinen[1,a)]　Hiroaki Anada[2,b)]

Abstract: Attribute-based authentication was proposed by Anada et al.(ACISP 2013). Also, semi-adaptive security of attribute-based encryption was proposed by Chen-Wee et al.(SCN2014). In this paper, we propose semi-adaptive man-in-the-middle attack security against attribute-based authentication. We prove that attribute-based authentication constructed by applying attribute-based encryption that is secure against semi-adaptive attacks to challenge-response is secure against semi-adaptive man-in-the-middle attacks. Goyal et al. (TCC2016-B) proposed a general transformation using Garbled Circuits and public key encryption from functional encryption that is secure against selective attacks to functional encryption that is secure against semi-adaptive attacks. We use this general transformation to write down the general configuration from an attribute-based key encapsulation mechanism that is secure against selective attacks to an attribute-based authentication that is secure against semi-adaptive man-in-the-middle attacks. Finally, we describe a performance comparison with attribute-based authentication that is secure against semi-adaptive man-in-the-middle attacks that do not use Garbled Circuits.

Keywords: Attribute-based Authentication, semi-Adaptive secuirty, Garbled circuits

[1] 長崎県立大学 地域創生研究科　情報工学専攻
,University of Nagasaki
[2] 長崎県立大学
,University of Nagasaki
[a)] mc120002@sun.ac.jp
[b)] anada@sun.ac.jp

1. 研究の背景

認証はアクセス制御の基本的なプロセスである．基本的な認証方式の一つに，公開鍵暗号スキームを使用するチャレンジ&レスポンス認証方式がある．

属性ベース暗号 (Attribute-Based Encryption,ABE) は公開鍵暗号を一般化したものである．属性ベース暗号は，属性とアクセス構造と呼ばれる概念を用いる．属性とは所属，資格，職位などの個人へ付与される概念のことである．アクセス構造とは，それらの属性をブール式に書き下したものである．平文を暗号化する時，属性セットの情報を含ませる．暗号文を復号する時，暗号文に含まれる属性セットが秘密鍵に関連づけられているアクセス構造のアサイメントパタンを満足した場合，復号に成功する．

属性ベース認証 (Attribute-Based Authentication,ABAuth) とは，公開鍵暗号スキームに属性ベース暗号スキームを採用したチャレンジ&レスポンス認証方式の事である．属性ベース認証では，検証者は属性セットを関連付けたチャレンジを証明者へ送る．証明者はアクセス構造が関連付けられた秘密鍵を使用してレスポンスを検証者へ返す．認証が成功する条件は，チャレンジに付与された属性セットが秘密鍵のアクセス構造を満足するアサイメントパタンの場合である．

Ostrovsky,Sahai,Waters [8] により、属性ベース鍵カプセル化機構 (Attribute-Based Key-Encapsulation Mechanism, ABKEM) が提案 (OSW-ABKEM) された．穴田ら [1] により ABKEM を使用した ABAuth が提案された．穴田ら [2] により，選択暗号文攻撃安全 (Chosen-Ciphertext Attack,CCA) な ABKEM を使用した ABAuth は，中間者攻撃安全であることが証明された．これまで属性ベース認証の暗号学的な取り扱いにおいては，攻撃者が標的属性セットを指定するタイミングは，選択的（selective）な安全性モデルにおいては公開鍵を得る前かつ各種クエリの前の段階，適応的（adaptive）な安全性モデルにおいては公開鍵を得た後かつ各種クエリの後の段階であった．しかしながら，現実には公開鍵を得た後かつ各種クエリの前の段階で攻撃者が標的属性セットを指定する攻撃パタンが自然であると考えられる．Chen ら [3] により，ABE の準適応的 (semi-adaptive) な安全性モデルが提案された．このモデルは攻撃者が標的属性セットを指定するタイミングは，公開鍵を得た後かつ各種クエリの前の段階である．従って，ABAuth に適応的な安全性モデルを要求することは過剰であり，効率性を考慮した準適応的な安全性モデルで十分である．

1.1 我々の貢献

本稿では，ABKEM を用いたチャレンジ&レスポンス認証スキーム ABAuth に対する準適応的中間者攻撃とその安全性を提案する．また，Goyal ら [5] が提案した選択的識別不可能選択平文攻撃安全な（selectively IND-CPA secure, sel-IND-CPA）関数型暗号から準適応的識別不可能選択平文攻撃安全な（semi-adaptively IND-CPA secure, sa-IND-CPA）関数型暗号への一般的変換を用い，選択的識別不可能選択平文攻撃安全な（sel-IND-CPA）ABKEM から準適応的識別不可能選択平文攻撃安全な（sa-IND-CPA）ABKEM への一般的変換を示す．我々の一般的変換は ABKEM，Yao の Garbled circuits [11] 及び IND-CPA 安全な公開鍵暗号スキーム（Public-Key Encryption,PKE）から ABKEM を構成するものである．次いで，選択的 CPA 安全な ABKEM が public verifiability を有するならば，Yamada ら [10] の CPA-to-CCA 変換（PKC2011）と Goyal ら [5] の一般的変換を使用して，得られた ABKEM から構成した ABAuth が準適応的中間者攻撃安全であることを述べる．

また，具体例に，Ostrovsky ら [8] の KP-ABKEM を非対称ペアリングかつ CCA-secure にした KP-ABKEM [4] に Goyal ら [5] の一般的変換を適用した KP-ABKEM を使用し．Garbled Circuits を使用しない ABKEM との ABAuth を構成した場合の性能比較をする．

2. 準備

本節では，用語についての解説を示す．$\mathbb{N}$ は自然数の集合である．λ はセキュリティパラメータで，$\lambda \in \mathbb{N}$ である．n は属性ユニバースである．n のサイズは，λ の多項式で抑えられる small universe である．

2.1 属性ベース鍵カプセル化機構（ABKEM）

ABKEM には秘密鍵にアクセス構造を関連付ける鍵ポリシー (key-policy,KP) モデルと暗号文にアクセス構造を関連付ける暗号文ポリシー (ciphertext-policy,CP) モデルがある．本稿では，鍵ポリシーモデルの ABKEM(KP-ABKEM) で記述する．ABKEM は 4 つの確率的多項式時間アルゴリズム $(\mathsf{Setup}_{\mathsf{ABKEM}}, \mathsf{KeyGen}_{\mathsf{ABKEM}}, \mathsf{Encap}_{\mathsf{ABKEM}}, \mathsf{Decap}_{\mathsf{ABKEM}})$ で構成される．

$\mathsf{Setup}_{\mathsf{ABKEM}}(1^\lambda, n) \to (\mathrm{PK}, \mathrm{MSK})$. $\mathsf{Setup}_{\mathsf{ABKEM}}$ アルゴリズムは，引数にセキュリティパラメータ 1^λ と属性ユニバース n を受け取り，公開鍵 PK とマスタ秘密鍵 MSK を出力する．

$\mathsf{KeyGen}_{\mathsf{ABKEM}}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}) \to \mathrm{SK}_{\mathbb{A}}$. $\mathsf{KeyGen}_{\mathsf{ABKEM}}$ アルゴリズムは，引数に公開鍵 PK, マスタ秘密鍵 MSK, アクセス構造 $\mathbb{A} \in 2^{\{n\}} \backslash \{\phi\}$ を受け取り，秘密鍵 $\mathrm{SK}_{\mathbb{A}}$ を出力する．

$\mathsf{Encap}_{\mathsf{ABKEM}}(\mathrm{PK}, S) \to (\kappa, \psi)$. $\mathsf{Encap}_{\mathsf{ABKEM}}$ アルゴリズムは，引数に公開鍵 PK と属性セット S を受け取り，ランダム KEM 鍵 κ とそのカプセル化 ψ を出力する．

$\mathsf{Decap}_{\mathsf{ABKEM}}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}, \psi) \to \hat{\kappa}$. $\mathsf{Decap}_{\mathsf{ABKEM}}$ アルゴリズム

は，引数に公開鍵 PK, 秘密鍵 $\mathrm{SK}_{\mathbb{A}}$, カプセル化 KEM 鍵 ψ を受け取り，復号した KEM 鍵 $\hat{\kappa}$ を出力する．

2.2 属性ベース認証 (ABAuth)

ABAuth には，証明者がアクセス構造を持つ証明者ポリシーモデル (Prover policy,PP) と検証者がアクセス構造を持つ検証者ポリシーモデル (Verifier policy,VP) モデルがある．本稿では，証明者ポリシーモデルを ABAuth(PP-ABAuth) で記述する．PP-ABauth は 4 つの多項式時間アルゴリズム $(\mathsf{Setup}_{\mathsf{ABAuth}}, \mathsf{KeyGen}_{\mathsf{ABAuth}}, \mathsf{P}, \mathsf{V})$ で構成される．

$\mathsf{Setup}_{\mathsf{ABAuth}}(1^\lambda, n) \to (\mathrm{PK}, \mathrm{MSK})$. Setup アルゴリズムは，引数にセキュリティパラメータ λ, 属性ユニバース n を受け取り，公開鍵 PK, マスタ秘密鍵 MSK を出力する．

$\mathsf{KeyGen}_{\mathsf{ABAuth}}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}) \to \mathrm{SK}_{\mathbb{A}}$. KeyGen アルゴリズムは，引数に公開鍵 PK, マスタ秘密鍵 MSK, アクセス構造 $\mathbb{A}$ を受け取り，アクセス構造 $\mathbb{A}$ に対応する秘密鍵 $\mathrm{SK}_{\mathbb{A}}$ を出力する．

$\mathsf{P}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}), \mathsf{V}(\mathrm{PK}, S)$. P は証明者，V は検証者に対応するアルゴリズムである．P アルゴリズムは，引数に公開鍵 PK, 秘密鍵 $\mathrm{SK}_{\mathbb{A}}$ を受け取る．V アルゴリズムは，引数に公開鍵 PK, 属性セット S を受け取る．P と V はいくらかの対話を行う．最後に V は，決定ビット b を出力する．$b = 1$ の場合，V は P を受け入れる．$b = 0$ の場合，V は P を排斥する．b が 1 を返す条件は，V が持つ属性セット S に対して，P は $S \in \mathbb{A}$ を満たす秘密鍵 $\mathrm{SK}_{\mathbb{A}}$ を所持していることである．

PP-ABauth は，任意の $1^\lambda, n$ に対して，$S \in \mathbb{A}$ を満足するならば，以下の性質を満たす．

$Pr[\mathsf{Setup}(1^\lambda, n) \to (\mathrm{PK}, \mathrm{MSK}); \mathsf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}) \to \mathrm{SK}_{\mathbb{A}}; b \leftarrow \langle P(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}), V(\mathrm{PK}, S)\rangle : b = 1] = 1$

2.3 公開鍵暗号 (PKE)

メッセージ空間 M_λ に対する公開鍵暗号スキームは 3 つのアルゴリズム $(\mathsf{Setup}_{\mathsf{PKE}}, \mathsf{Enc}_{\mathsf{PKE}}, \mathsf{Dec}_{\mathsf{PKE}})$ で構成される．

$\mathsf{Setup}_{\mathsf{PKE}}(1^\lambda) \to (\mathrm{PK}_{\mathsf{PKE}}, \mathrm{SK}_{\mathsf{PKE}})$. $\mathsf{Setup}_{\mathsf{PKE}}$ アルゴリズムは，引数にセキュリティパラメータ 1^λ を受け取り，公開鍵 $\mathrm{PK}_{\mathsf{PKE}}$ と秘密鍵 $\mathrm{SK}_{\mathsf{PKE}}$ を出力する．

$\mathsf{Enc}_{\mathsf{PKE}}(\mathrm{PK}, m \in M_\lambda) \to \mathsf{c}$. $\mathsf{Enc}_{\mathsf{PKE}}$ アルゴリズムは，引数に公開鍵 $\mathrm{PK}_{\mathsf{PKE}}$ とメッセージ空間 M_λ 上のメッセージ m を受け取り，暗号文 c を出力する．

$\mathsf{Dec}_{\mathsf{PKE}}(\mathrm{SK}, \mathsf{c}) \to M_\lambda$. $\mathsf{Dec}_{\mathsf{PKE}}$ アルゴリズムは，引数に秘密鍵 $\mathrm{SK}_{\mathsf{PKE}}$ と暗号文 c を受け取り，メッセージ m を出力する．

2.4 Garbled circuits [6]

Garbling scheme は 2 つの多項式時間アルゴリズム $(\mathsf{Garble}, \mathsf{Eval})$ で構成される．

$\mathsf{Garble}(\mathsf{C} \in C_n, 1^\lambda) \to (C, \{\omega_{i,b}\}_{i \le n, b \in \{0,1\}})$. Garble アルゴリズムは，引数にセキュリティパラメータ λ と回路 $\mathsf{C} \in C_n$ を受け取り，Garbled 回路 C と対応する $2n$ 個のワイヤタグ $\omega_{i,b}$ を出力する．

$\mathsf{Eval}(\mathsf{C}, \{\omega_i\}_{i \le n}) \to y \in \{0,1\}$. Eval アルゴリズムは，引数に Garled 回路 G と n 個のワイヤタグ ω_i を受け取り，$y \in \{0,1\}$ を出力する．

2.5 ABKEM の選択的安全性 [8] と準適応的安全性 [3]

ABKEM に対して以下のような実験を定義する．

$\mathsf{Exp}^{\text{ow-x-cca}}_{\mathcal{A},\text{KP-ABKEM}}(1^\lambda, n)$:
$(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathsf{Setup}(\lambda, n)$
$S^* \leftarrow \mathcal{A}(x, n)$
$(\kappa^*, \psi^*) \leftarrow \mathsf{Encap}(\mathrm{PK}, S^*)$
$\hat{\kappa^*} \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot), \mathcal{DEC}(\mathrm{PK},\mathrm{SK}_\cdot,\cdot)}(\mathrm{PK}, \psi^*)$
If $\hat{\kappa^*} = \kappa^*$ then Return Win else Return Lose

$x = 1^\lambda$: 選択的安全性　$x = \mathrm{PK}$: 準適応的安全性

実験において，攻撃者 $\mathcal{A}$ は入力に (x, n) を受け取り，標的とする属性セット S^* を宣言する．もし $\mathcal{A}$ が受け取る入力が $(1^\lambda, n)$ である場合，選択的な攻撃であるという．もし $\mathcal{A}$ が受け取る入力が (PK, n) である場合，準適応的な攻撃であるという．$\mathcal{A}$ は鍵生成オラクル $\mathcal{KG}$ と脱カプセル化オラクル $\mathcal{DEC}$ へアクセスすることが可能である．$\mathcal{KG}$ は $\mathbb{A}_i$ でクエリを受け取ると，秘密鍵 $\mathrm{SK}_{\mathbb{A}_i}$ を返す．$\mathcal{DEC}$ は $(\mathbb{A}_j, \psi_j)$ でクエリを受け取ると，KEM 鍵 $\hat{\kappa}_j$ を返す．

3. 準適応的中間者攻撃安全な属性ベース認証スキーム

本節では, 準適応的中間者攻撃安全の定義を記述する．

$Expr^{\text{sa-cmim}}_{\mathcal{A},\text{PP-ABAuth}}(\lambda, n)$:
$(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathsf{Setup}(\lambda, n)$
$S^* \leftarrow \mathcal{A}(\mathrm{PK}, n)$
$b \leftarrow \langle \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot), \mathsf{P}_j(\mathrm{PK},\mathrm{SK}_\cdot)|_{j=1}^{q_\mathrm{p}}}(\mathrm{PK}), \mathsf{V}(\mathrm{PK}, S^*)\rangle$
If $b = 1$ then Return Win else Return Lose

準適応的中間者攻撃の実験では，$\mathcal{A}$ は公開鍵 PK を受け取った後に標的属性セット S^* を指定する．$\mathcal{A}$ は $\mathcal{KG}$ と証明者 P_j へ問い合わせることができる．$\mathcal{A}$ はアクセス構造 $\mathbb{A}_i$ を与えられると，$\mathcal{KG}(\mathrm{PK}, \mathrm{MSK}\cdot)$ へ鍵抽出クエリを発行し秘密鍵 $\mathrm{SK}_{\mathbb{A}_i}$ を得る．さらに $\mathcal{A}$ はアクセス構造 $\mathbb{A}_j (j = 1, \cdots, q_p)$ を選択し，$\mathsf{P}_j(\mathrm{PK}, \mathrm{SK}_\cdot)$ を呼び出す．$\mathcal{A}$ は属性セット S_j を持つ検証者として各々の P_j と対話する．

ただし，$\mathcal{A}$ は S^* に関して二つの制約が課される．鍵発行クエリの $\mathbb{A}_i$ は $S^* \notin \mathbb{A}_i$ を満たさなければならな

い．各証明者との対話において $S^* \notin \mathbb{A}_j$ を満たし，証明者 $\mathsf{P}_j(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}_j})$ との対話のやり取りの記録と検証者 $V(\mathrm{PK}, S^*)$ との対話のやり取りの記録が等しくあってはならない．

$$Adv^{\text{sa-cmim}}_{\mathcal{A},\mathsf{PP\text{-}ABAuth}}(\lambda) \stackrel{\text{def}}{=} Pr[Expr^{\text{sa-cmim}}_{\mathcal{A},\mathsf{PP\text{-}ABAuth}}(\lambda, n) \ \ returns \ \ \text{WIN}]$$

4. 選択的安全な ABKEM から準適応的安全な ABKEM への一般的変換 [5]

本節では, 選択的安全な ABKEM から準適応的安全な ABKEM への一般的変換を示す．Goyal-Koppula-Waters [5] により一般的な関数型暗号（functional encryption）について選択的安全ななスキームから準適応的安全なスキームへの一般的変換が示されている．[5] に従い, 我々は ABKEM の場合について同様の一般的変換を書き下す．

変換で得られる ABKEM は 4 つの多項式時間アルゴリズム $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ で構成される．

$\mathsf{Setup}(1^\lambda, n) \rightarrow (\mathrm{PK}, \mathrm{MSK})$. Setup アルゴリズムは，セキュリティパラメータ 1^λ と属性ユニバース n を受け取り，$\mathsf{Setup}_{\mathsf{ABKEM}}(1^\lambda, n)$ と $\mathsf{Setup}_{\mathsf{PKE}}(1^\lambda)$ を実行する．実行の結果として $(\mathrm{PK}_{\mathsf{ABKEM}}, \mathrm{MSK}_{\mathsf{ABKEM}})$ と $(\mathrm{PK}_{\mathsf{PKE}_{i,b}}, \mathrm{SK}_{\mathsf{PKE}_{i,b}})_{b=0,1}^{0\le i\le l}$ を得る．最後に公開鍵 $\mathrm{PK} = (\mathrm{PK}_{\mathsf{PKE}_{i,b}})$ とマスタ秘密鍵 $\mathrm{MSK} = (\mathrm{PK}_{\mathsf{ABKEM}}, \mathrm{MSK}_{\mathsf{ABKEM}}, \mathrm{SK}_{\mathsf{PKE}_{i,b}})$ を出力する．

$\mathsf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}) \rightarrow \mathrm{SK}_{\mathbb{A}}$. KeyGen アルゴリズムは，公開鍵 PK，マスタ秘密鍵 MSK，アクセス構造 $\mathbb{A}$ を受け取り，$\mathsf{KeyGen}_{\mathsf{ABKEM}}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A})$ を実行する．実行の結果として，$\mathrm{SK}_{\mathsf{ABKEM}_{\mathbb{A}}}$ を得る．最後に秘密鍵 $\mathrm{SK}_{\mathbb{A}} = (\mathrm{PK}_{\mathsf{ABKEM}}, \mathrm{SK}_{\mathsf{ABKEM},\mathbb{A}}, \mathrm{SK}_{\mathsf{PKE}_{i,\mathrm{PK}_{\mathsf{ABKEM}[i]}}})$ を出力する．

$\mathsf{Encap}(\mathrm{PK}, S) \rightarrow ct$. Encap アルゴリズムは，公開鍵 PK と属性セット S を受け取り，$\mathsf{Garble}(\mathsf{Encap}_{\mathsf{ABKEM}}(\mathrm{PK}, S))$ を実行する．実行の結果，Garbled 回路 C とワイヤ $\omega_{i,b}$ を得る．次に $\mathsf{Encap}_{\mathsf{PKE}}(\omega_{i,b}, \mathrm{PK}_{\mathsf{PKE}_{i,b}})$ を実行し，ワイヤの暗号化 $c_{i,b}$ を得る．最後に暗号文 $ct = (C, c_{i,b})$ を出力する．

$\mathsf{Decap}(\mathrm{SK}_{\mathbb{A}}, \mathrm{PK}, ct) \rightarrow \mathsf{Decap}_{\mathsf{ABKEM}}$. Decap アルゴリズムは，秘密鍵 $\mathrm{SK}_{\mathbb{A}}$，公開鍵 PK，暗号文 ct を受け取り，$\mathsf{Decap}_{\mathsf{PKE}}(\mathrm{SK}_{\mathsf{PKE}_{i,\mathrm{PK}_{\mathsf{ABKEM}[i]}}}, ct_{i,\mathrm{PK}_{\mathsf{ABKEM}}[i]})$ を実行する．実行の結果，復号されたワイヤ $\hat{\omega}_i$ を得る．次に $\mathsf{Eval}(C, \hat{\omega}_i)$ を実行し，KEM 鍵の暗号化 ψ を得る．最後に $\mathsf{Decap}(\mathrm{PK}_{\mathsf{ABKEM}}, \mathrm{SK}_{\mathsf{ABKEM}}, \psi)$ を実行する．

Theorem 1 ([5]). *ABKEM* が選択的安全ならば，変換後の *ABKEM* は準適応的安全である．

Algorithm 1 $\mathsf{Setup}(1^\lambda, n)$

$(\mathrm{PK}_{\mathsf{ABKEM}}, \mathrm{MSK}_{\mathsf{ABKEM}}) \leftarrow \mathsf{Setup}_{\mathsf{ABKEM}}(1^\lambda, n)$
for $i = 1 \ \ to \ \ l$: **do**
  for $b = 0, 1$: **do**
    $(\mathrm{PK}_{\mathsf{PKE}_{i,b}}, \mathrm{SK}_{\mathsf{PKE}_{i,b}}) \leftarrow \mathsf{Setup}_{\mathsf{PKE}}(1^\lambda)$
  end for
end for
$\mathrm{PK} = \mathrm{PK}_{\mathsf{PKE}_{i,b}}, \mathrm{MSK} = (\mathrm{PK}_{\mathsf{ABKEM}}, \mathrm{MSK}_{\mathsf{ABKEM}}, \mathrm{SK}_{\mathsf{PKE}_{i,b}})$
return $(\mathrm{PK}, \mathrm{MSK})$

Algorithm 2 $\mathsf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A} = (M, \rho))$

$\mathrm{SK}_{\mathsf{ABKEM},\mathbb{A}} \leftarrow \mathsf{KeyGen}_{\mathsf{ABKEM}}(\mathrm{PK}_{\mathsf{ABKEM}}, \mathrm{MSK}_{\mathsf{ABKEM}}, \mathbb{A})$
$\mathrm{SK}_{\mathbb{A}} = (\mathrm{PK}_{\mathsf{ABKEM}}, \mathrm{SK}_{\mathsf{ABKEM},\mathbb{A}}, \{\mathrm{SK}_{\mathsf{PKE}_{i,\mathrm{PK}_{\mathsf{ABKEM}}[i]}}\}_{i\le l})$
return $\mathrm{SK}_{\mathbb{A}}$

Algorithm 3 $\mathsf{Encap}(\mathrm{PK}, S)$

$(C, \omega_{i,b}) \leftarrow \mathsf{Garble}(\mathsf{Encap}_{\mathsf{ABKEM}}(\mathrm{PK}, S; r), 1^\lambda)$
for $i = 1 \ \ to \ \ l$: **do**
  for $b = 0, 1$: **do**
    $(c_{i,b}) \leftarrow \mathsf{Encap}_{\mathsf{PKE}}(\omega_{i,b}, \mathrm{PK}_{\mathsf{PKE}_{i,b}})$
  end for
end for
$\mathsf{c}_{i,b} = (c_{i,b})$
$ct = \{C, \mathsf{c}_{i,b}\}$
return ct

Algorithm 4 $\mathsf{Decap}(\mathrm{SK}_{\mathbb{A}}, \mathrm{PK}, ct)$

for $i = 1 \ \ to \ \ l$: **do**
  $\hat{\omega}_i \leftarrow \mathsf{Decap}_{\mathsf{PKE}}(\mathrm{SK}_{\mathsf{PKE}_{i,\mathrm{PK}_{\mathsf{ABKEM}}[i]}}, ct_{i,\mathrm{PK}_{\mathsf{ABKEM}}[i]})$
end for
$\psi \leftarrow \mathsf{Eval}(C, \hat{\omega}_i)$
return $\mathsf{Decap}_{\mathsf{ABKEM}}(\mathrm{PK}_{\mathsf{ABKEM}}, \mathrm{SK}_{\mathsf{ABKEM},\mathbb{A}}, \psi)$

5. 選択的選択平文攻撃安全な ABKEM から準適応的中間者攻撃安全な ABAUTH への一般的変換

本節では，選択的選択平文攻撃 (chosen-plaintext attack、CPA) 安全な ABKEM から準適応的中間者攻撃安全な ABAUTH の一般的変換を示す．変換は以下の 3 つの段階で行われる．

(1) 選択的 CPA 安全な ABKEM から選択的 CCA 安全な ABKEM への変換.
(2) 選択的 CCA 安全な ABKEM から準適応的 CCA 安全な ABKEM への変換.
(3) 準適応的 CCA 安全な ABKEM から準適応的中間者攻撃安全な ABAuth への変換.

(1) : 選択的 CPA 安全な ABKEM が $\mathsf{Verify}(\mathrm{PK}, ct, S, S') \rightarrow \{0, 1, \perp\}$ アルゴリズムを内在する Public verifiablity の性質を持つと仮定すると，Yamada ら [10] が提案した CPA-to-CCA コンパイラの適用が可能となる．Yamada らのコンパイラではワンタイム署名スキームを使用する．ワンタイム署名スキームは三つ

の多項式時間アルゴリズム $(\mathsf{SGK}, \mathsf{Sign}, \mathsf{Vrfy})$ で構成される．

$\mathsf{SGK}(1^\lambda) \to (\mathrm{vk}, \mathrm{sgk})$. SGK アルゴリズムは，引数にセキュリティパラメータ 1^λ を受け取り，検証鍵 vk と署名鍵 sgk を出力する．

Sign と Vrfy は，それぞれ署名アルゴリズムと検証アルゴリズムである．

また，変換では属性ユニバース n とは独立する以下の性質を満たすダミーユニバース W を利用する．

- $t := poly(1^\lambda)$
- $W = \{P_{1,0}, P_{1,1}, P_{2,0}, P_{2,1}, \cdots, P_{t,0}, P_{t,1}\}$

[10] の CPA-to-CCA 変換を使用し，選択的 CPA 安全な ABKEM から選択的 CCA 安全な ABKEM への変換を行う．

Theorem 2 ([10]). *ABKEM が CPA 安全かつ Public verifiablity の性質を持つならば，変換後の ABKEM は CCA 安全である．*

(2) : 変換で得られた選択的 CCA 安全な ABKEM に Goyal ら [5] の一般的変換を適用し，選択的 CCA 安全な ABKEM から準適応的 CCA 安全な ABKEM への変換を行う．変換方法は Section4 に記述している．

(3) : 変換で得られた準適応的 CCA 安全な ABKEM を準適応的中間者攻撃安全な ABAuth への一般的変換を示す．

変換で得られた ABAuth は 4 つの多項式時間アルゴリズム $(\mathsf{Setup}_{\mathsf{ABAuth}}, \mathsf{KeyGen}_{\mathsf{ABAuth}}, \mathsf{V}, \mathsf{P})$ で構成される．

$\mathsf{Setup}_{\mathsf{ABAuth}}(1^\lambda, n) \to (\mathrm{PK}, \mathrm{MSK})$. $\mathsf{Setup}_{\mathsf{ABAuth}}$ アルゴリズムは，引数にセキュリティパラメータ 1^λ と属性ユニバース n を受け取り，$\mathsf{Setup}(1^\lambda, n \cup W)$ を実行する．実行の結果，公開鍵 PK とマスタ秘密鍵 MSK を得る．最後に PK と MSK を出力する．

$\mathsf{KeyGen}_{\mathsf{ABAuth}}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}) \to (\mathrm{SK}_{\mathbb{A}})$. $\mathsf{KeyGen}_{\mathsf{ABAuth}}$ は，引数に公開鍵 PK，マスタ秘密鍵 MSK，アクセス構造 $\mathbb{A}$ を受け取り，$\mathsf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A})$ を実行する．実行の結果，秘密鍵 $\mathrm{SK}_{\mathbb{A}}$ を得る．最後に $\mathrm{SK}_{\mathbb{A}}$ を出力する．

$\mathsf{V}(\mathrm{PK}, S), \mathsf{P}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}})$. V アルゴリズムは，引数に公開鍵 PK と属性セット S を受け取り，$\mathsf{SGK}(1^\lambda)$ を実行する．実行の結果，検証鍵 vk と署名鍵 sgk を得る．次にダミーユニバース W の属性セット S_{vk} を指定し，$\mathsf{Encap}(\mathrm{PK}, S \cup S_{\mathrm{vk}})$ を実行し暗号文 ct を得る．得られた暗号文に $\mathsf{Sign}(\mathrm{sgk}, ct)$ を実行し，署名を行う．そして暗号文 $ct' = (\mathrm{vk}, ct, \sigma)$ を証明者へ送る．

P アルゴリズムは暗号文 ct' を受け取り，分析して $\mathsf{Vrfy}(\mathrm{vk}, ct, \sigma)$ を実行する．Vrfy が 0 or $\perp$ を返した場合，アボートする．Vrfy が 1 を返した場合，$\mathsf{Verify}(\mathrm{PK}, ct, \mathbb{A}, \wedge_{S_{\mathrm{vk}}} P)$ を実行する．Verify が 0 or $\perp$ を返した場合，アボートする．Verify が 1 を返した場合，$\mathsf{Decap}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}, ct)$ を実行する．実行の結果，KEM 鍵 $\hat{\kappa}$ を得る．そして $\hat{\kappa}$ を検証者へ送る．

$\hat{\kappa}$ を受け取った V は，KEM 鍵 κ との比較を行う．$\kappa = \hat{\kappa}$ である場合，$d = 1$ を出力する．それ以外の場合，$d = 0$ を出力する．

Algorithm 5 $\mathsf{Setup}_{\mathsf{ABAuth}}(1^\lambda, n)$

$(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathsf{Setup}(1^\lambda, n \cup W)$
return $(\mathrm{PK}, \mathrm{MSK})$

Algorithm 6 $\mathsf{KeyGen}_{\mathsf{ABAuth}}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A})$

$\mathrm{SK}_{\mathbb{A}} \leftarrow \mathsf{KeyGen}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A})$
return $\mathrm{SK}_{\mathbb{A}}$

Algorithm 7 $\mathsf{V}(\mathrm{PK}, S)$

$\mathsf{SGK}(1^\lambda) \to (\mathrm{vk}, \mathrm{sgk})$
$S_{\mathrm{vk}} = \{P_{1,\mathrm{vk}_1}, P_{2,\mathrm{vk}_2}, \cdots, P_{t,\mathrm{vk}_t}\} \subset W$
$ct \leftarrow \mathsf{Encap}(\mathrm{PK}, S \cup S_{\mathrm{vk}})$
$\mathsf{Sign}(\mathrm{sgk}, ct) \to \sigma$
$ct' = (ct, \mathrm{vk}, \sigma)$
Send ct' to P
Receive $\hat{\kappa}$ to P
if $\hat{\kappa} = \kappa$ **then**
  $d = 1$
else
  $d = 0$
end if
return d

Algorithm 8 $\mathsf{P}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}})$

Receive ct' to V
if $\mathsf{Vrfy}(ct, \mathrm{vk}, \sigma) = 0$ *or* $\perp$ **then**
  return $\perp$
else
  if $\mathsf{Verify}(\mathrm{PK}, ct, \mathbb{A}, \wedge_{P \in S_{\mathrm{vk}}} P) = 0$ *or* $\perp$ **then**
    return $\perp$
  else
    $\hat{\kappa} \leftarrow \mathsf{Decap}(\mathrm{PK}, \mathrm{SK}_{\mathbb{A}}, ct)$
  end if
end if
Send $\hat{\kappa}$ to V

6. 安全性証明

本節では，選択的 CPA 安全な ABKEM を適切に変換して得られる ABAuth が準適応的中間者攻撃安全を満たすことを証明する．

Theorem2 より，選択的 CPA 安全な ABKEM を [10] の CPA-to-CCA 変換を使用し，得られた ABKEM は選択的 CCA 安全を満たす．Theorem1 より，選択的 CCA 攻撃安全な ABKEM を [5] の一般的変換を使用し，得られた

ABKEM は準適応的 CCA 安全を満たす．

準適応的 CCA 安全な ABKEM を使用した ABAuth が準適応的中間者攻撃安全であることを示す．

Theorem 3. *ABKEM* が準適応的 *CCA* 安全ならば変換で得られた *ABAUTH* は準適応的中間者攻撃安全である．

$$Adv^{sa-cmim}_{\mathcal{A},ABauth}(1^\lambda) \leq Adv^{sa-cca}_{\mathcal{B},ABKEM}(1^\lambda)$$

Proof. ABAuth に対する任意の多項式時間の準適応的中間者攻撃者 $\mathcal{A}$ を内部に雇うことで，ABKEM に対する多項式時間の準適応的 CCA 攻撃者 $\mathcal{B}$ を構築できることを示す．

$\mathcal{B}(\mathrm{PK}, n)$:

Setup :

内部初期化, $\mathcal{A}(\mathrm{PK}, n)$ の呼び出し

Ansering($\mathcal{A}$ のクエリ) :

$\mathcal{A}$ が標的属性セット S^* を出力した時

・S^* を $\mathcal{B}$ の目標属性セットとして出力

・チャレンジ ψ^* を受け取る

$\mathcal{A}$ が鍵抽出クエリ $(\mathbb{A})$ を発行した時

$\mathrm{SK}_\mathbb{A} \leftarrow \mathcal{KG}(\mathrm{PK}, \mathrm{MSK}, \mathbb{A}), \mathrm{SK}_\mathbb{A}$ を $\mathcal{A}$ に返す

$\mathcal{A}$ が P へのチャレンジ $(\mathbb{A}, \psi)$ を発行した時

$\hat{\kappa} \leftarrow \mathcal{DEC}(\mathrm{PK}, \mathrm{SK}_\mathbb{A}, \psi)$, レスポンス $\hat{\kappa}$ を $\mathcal{A}$ に返す

$\mathcal{A}$ が V へチャレンジをクエリした時

チャレンジ ψ^* を $\mathcal{A}$ へ送る

$\mathcal{A}$ が V へレスポンス $\hat{\kappa^*}$ を送る時

レスポンス $\hat{\kappa^*}$ を出力

$\mathcal{B}$ は引数に (PK, n) を受け取り，内部初期化と (PK, n) で $\mathcal{A}$ を呼び出す．$\mathcal{A}$ が標的属性セット S^* を出力した時，S^* を $\mathcal{B}$ の標的属性セットとして出力する．その時，$\mathcal{B}$ は V からチャレンジ暗号文 ψ^* を受け取る．$\mathcal{A}$ が $\mathbb{A}$ で鍵抽出クエリを発行した時，$\mathcal{B}$ は鍵生成オラクル $\mathcal{KG}(\mathrm{PK}, \mathrm{MSK}, \cdot)$ にクエリを行い，結果 $\mathrm{SK}_\mathbb{A}$ を $\mathcal{A}$ へを返す．$\mathcal{A}$ が $(\mathbb{A}, \psi)$ を証明者 P へ送る時，$\mathcal{B}$ は脱カプセル化オラクル $\mathcal{DEC}(\mathrm{PK}, \mathrm{SK}., \cdot)$ へクエリを行い，結果 $\hat{\kappa}$ を $\mathcal{A}$ へ返す．$\mathcal{A}$ が検証者 V へチャレンジをクエリする時，$\mathcal{B}$ はチャレンジとして ψ^* を返す．$\mathcal{A}$ が V へレスポンス $\hat{\kappa^*}$ を送るとき，$\mathcal{B}$ は推測として $\hat{\kappa^*}$ を出力する．

$\mathcal{B}$ の内部にある $\mathcal{A}$ は，実際の $\mathcal{A}$ の見ているものと同じである．もし $\mathcal{A}$ が攻撃に成功するならば，$\mathcal{B}$ も攻撃に成功する．よって，Theorem3 の不等式は成り立つ． □

7. 性能の漸近的特性の考察

本節では，Goyal ら [5] の一般的変換を使用した ABKEM で構築した ABAuth の性能の漸近的特性を考察する．そのため，Garbled Circuits を用いない ABKEM を使用した ABAuth の性能の比較する．具体的に比較する ABKEM は，以下の 5 つである．

- Ostrovsky らの選択的安全な ABKEM [8] を Goyal らの変換方式 [5] で変換し，得られた準適応的安全な ABKEM
- Chen らの準適応的安全な ABKEM(合成位数の双線形群) [3]
- Chen らの準適応的安全な ABKEM(素数位数の双線形群) [3]
- Okamato らの適応的安全な ABKEM [7]
- Tkashima の準適応的安全な ABKEM [9]

比較する項目は ABAuth に実装した場合の以下の 5 つである，

- 秘密鍵長:|SK|
- 公開鍵長:|PK|
- チャレンジ文長:$|cha|$
- 証明者の計算時間:P time
- 検証者の計算時間:V time

これらの比較の結果を表 1 に示している．

Goyal ら [5] の一般的変換により得られた準適応的攻撃安全な ABKEM は，ABKEM の暗号化アルゴリズムを，Garbled Circuits で表現し，Garbled Circuits の各ワイヤを公開鍵暗号スキームで暗号化するという，2 重の暗号化構造になっているため，暗号化の計算時間は長くなる．また，復号アルゴリズムにおいても同様に，2 重の暗号化構造を復号をする必要があるため，復号の計算時間も長くなる．また，Garbled Circuits の各ワイヤに対して，公開鍵暗号スキームの公開鍵 $\mathrm{PK}_{\mathsf{PKE}}$ と秘密鍵 $\mathrm{SK}_{\mathsf{PKE}}$ を生成する必要があるため，鍵の量も膨大となる．そして変換後の ABKEM は，ABKEM が生成する公開鍵 $\mathrm{PK}_{\mathsf{ABKEM}}$ 及び秘密鍵 $\mathrm{SK}_{\mathsf{ABKEM}}$ と公開鍵暗号スキームが生成する秘密鍵 $\mathrm{SK}_{\mathsf{PKE}}$ を合わせ，秘密鍵 $\mathrm{SK} := (\mathrm{PK}_{\mathsf{ABKEM}}, \mathrm{SK}_{\mathsf{ABKEM}}, \mathrm{SK}_{\mathsf{PKE}})$ として出力するため，秘密鍵長も長くなる．さらに，Garbled Circuits と各ワイヤを暗号文として出力するため，暗号文長も長くなる．

[5] の一般的変換で得られた準適応的攻撃安全な ABKEM から構成した ABAuth は，他の ABAuth と比較してチャレンジ文長，検証者の計算時間，証明者の計算時間の項目で最も長い．また，秘密鍵長と公開鍵長も他と比較して短いとは言えない．

具体例に選択的攻撃安全な OSW-ABKEM [8] を使用したが，比較項目に Garbled Circuits のワイヤ数が影響しているため，他の選択的攻撃安全な ABKEM を使用したとしても，大きな変化はないと思われる．

得られた結果から，[5] の一般的変換をチャレンジ&レスポンス認証スキームの構成に用いることは，効率が良いと

は言えない．

[5] の一般的変換の利点は，Garbled Circuits と公開鍵暗号スキームがあれば，任意の選択的攻撃安全な関数型暗号を準適応的攻撃安全な関数型暗号に変換することが可能な汎用性の高さである．しかしながら，チャレンジ&レスポンス認証スキームの構成で用いる場合には適していないと思われる．

8. まとめと今後の課題

我々は属性ベース認証における準適応的中間者攻撃の提案と安全性を証明した．また，Goyal ら [5] が提案した一般的変換を使用し，選択的攻撃安全な ABKEM から準適応的中間者攻撃安全な ABAuth への一般的構成を示した．さらに，Goyal の [5] の一般的変換を使用する ABAuth と使用しない ABAuth との性能を比較し，[5] の一般的変換を認証スキームの構成に使用するには，適していないという結論に至った．

属性ベース暗号で構成する属性ベース認証が中間者攻撃安全であるためには，属性ベース暗号が CCA 安全である必要がある．性能比較で参考とした ABKEM は，CCA 安全性が証明されていないものも含まれている．鍵長や計算時間などの定量的な評価を行ったが，参考にした一部の ABKEM は CCA 安全性への変換が可能であることが証明されていないため，定性的な評価が不十分である．

今後の課題として，参考にした ABKEM の CCA 安全性の証明をし，定性的な評価を踏まえ，属性ベース認証の構成に適した属性ベース暗号の研究をする．

謝辞

著者らは本稿に関連する研究 [4] の準適応的攻撃に対する安全性を検討すべきことをコメント下さった江村恵太様に深謝致します．

参考文献

[1] H. Anada, S. Arita, S. Handa, and Y. Iwabuchi.
Attribute-based identification: Definitions and efficient constructions.
In *Information Security and Privacy - 18th Australasian Conference, ACISP 2013, Brisbane, Australia, July 1-3, 2013. Proceedings*, pages 168–186, 2013.

[2] H. Anada, S. Arita, S. Handa, and Y. Iwabuchi.
Attribute-based identification: Definitions and efficient constructions.
IEICE Trans. Fundam. Electron. Commun. Comput. Sci., 97-A(5):1086–1102, 2014.

[3] J. Chen and H. Wee.
Semi-adaptive attribute-based encryption and improved delegation for boolean formula.
In M. Abdalla and R. D. Prisco, editors, *Security and Cryptography for Networks - 9th International Conference, SCN 2014, Amalfi, Italy, September 3-5, 2014. Proceedings*, volume 8642 of *Lecture Notes in Computer Science*, pages 277–297. Springer, 2014.

[4] K. Chinen and H. Anada.
Construction and evaluation of attribute-based challenge-and-response authentication on asymmetric bilinear map.
In *Seventh International Symposium on Computing and Networking Workshops, CANDAR 2019 Workshops, Nagasaki, Japan, November 26-29, 2019*, pages 320–326. IEEE, 2019.

[5] R. Goyal, V. Koppula, and B. Waters.
Semi-adaptive security and bundling functionalities made generic and easy.
In M. Hirt and A. D. Smith, editors, *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, volume 9986 of *Lecture Notes in Computer Science*, pages 361–388, 2016.

[6] Y. Lindell, editor.
Tutorials on the Foundations of Cryptography.
Springer International Publishing, 2017.

[7] T. Okamoto and K. Takashima.
Fully secure functional encryption with general relations from the decisional linear assumption.
IACR Cryptol. ePrint Arch., 2010:563, 2010.

[8] R. Ostrovsky, A. Sahai, and B. Waters.
Attribute-based encryption with non-monotonic access structures.
In *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007*, pages 195–203, 2007.

[9] K. Takashima.
Expressive attribute-based encryption with constant-size ciphertexts from the decisional linear assumption.
IEICE Trans. Fundam. Electron. Commun. Comput. Sci., 103-A(1):74–106, 2020.

[10] S. Yamada, N. Attrapadung, G. Hanaoka, and N. Kunihiro.
Generic constructions for chosen-ciphertext secure attribute based encryption.
In D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, editors, *Public Key Cryptography - PKC 2011 - 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings*, volume 6571 of *Lecture Notes in Computer Science*, pages 71–89. Springer, 2011.

[11] A. C. Yao.
How to generate and exchange secrets (extended abstract).
In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167. IEEE Computer Society, 1986.

表 1 AB-Auth 性能比較：準適応的安全 対 適応的安全.
n は属性ユニバースの位数，l はシェア生成行列の行数．M_{V} は V で使用される属性の最大数，M_{P} は P で使用される属性の最大数，v は Garbled circuits のワイヤ数，CW14-1 は合成位数の双線形群ベースの準適応的安全スキーム，CW14-2 は素数位数の双線形群ベースの準適応的安全スキーム．|SK| は SK における群の元の個数．|PK| は PK における群の元の個数 time は群の元のべき乗の回数，あるいは，2 元の乗算の回数で評価.

AB-Auth スキーム	security	|SK|	|PK|	$\|cha\|$	P time	V time
OSW07 [8]+GKW16 [5]	semi-adaptive	$O(n + M_{\mathsf{P}} + v)$	$O(v)$	$O(M_{\mathsf{V}} + v)$	$O(M_{\mathsf{P}} + v)$	$O(M_{\mathsf{V}} + v)$
CW14-1 [3]	semi-adaptive	$O(nl)$	$O(n)$	$O(1)$	$O(M_{\mathsf{P}})$	$O(1)$
CW14-2 [3]	semi-adaptive	$O(l)$	$O(n)$	$O(n)$	$O(M_{\mathsf{P}})$	$O(M_{\mathsf{V}})$
OT10 [7]	adaptive	$O(M_{\mathsf{P}} \cdot l)$	$O(n)$	$O(M_{\mathsf{V}})$	$O(M_{\mathsf{P}})$	$O(M_{\mathsf{V}})$
T20 [9]	semi-adaptive	$O(l)$	$O(n)$	$O(1)$	$O(n)$	$O(n)$