1X-07

# 鍵生成センタに対して安全なIDベース暗号の実装評価

佐藤 裕太 †　江村 恵太 ‡　大東 俊博 §
東海大学 †　情報通信研究機構 ‡　東海大学/情報通信研究機構 §

## 1 はじめに

ID ベース暗号 (Identity-Based Encryption，IBE)[1] は公開鍵暗号の 1 種であり，公開鍵として任意の値 (ID) を設定できる．鍵生成センタ (Key Generartion Center，KGC) はマスター公開鍵 mpk とマスター秘密鍵 msk を管理しており，ID が正当であることを確認した後 msk を使用して ID に対応する秘密鍵 $\mathrm{sk}_{\mathrm{ID}}$ を発行する．一度 mpk を入手すれば，後は ID を知ったタイミングでいつでも暗号化を行うことができる．しかしながら，ID ベース暗号において KGC は全ての暗号文を復号できる強い権限があることが課題とされている．この課題に対し，江村ら [2] は KGC に対して安全な ID ベース暗号を提案している．本稿では，江村らのペアリングベースの方式 [3] を実装し，秘密鍵生成時のサーバとの通信時間を含めて性能評価を行った結果を報告する．

## 2 江村らの方式

本章では，江村らの KGC に対して安全な ID ベース暗号を紹介する．なお [3] では対称ペアリングを用いた方式が提案されていたが，効率性の観点から一般的に非対称ペアリングが用いられる．そこで本稿では非対称ペアリングを用いた方式を新たに構成した．江村らの方式では，KGC が行っていた ID 確認機能を ID 認証局 (Identity-Certifying Authority，ICA) に分離している (図 1)．さらに鍵生成にブラインド署名技術を応用することで，KGC は ID に関する情報を得ることなく秘密鍵を生成し，かつ生成された秘密鍵はユーザのみが得られる仕組みを Boneh-Franklin ID ベース暗号 [4] に付与している．KGC と ICA は結託しない，かつ ID はランダムに選ばれるという仮定の下，KGC は暗号文から平文に関する情報を得ることはできないことが示されている．

### 2.1 準備

$\mathbb{G}_1$，$\mathbb{G}_2$，$\mathbb{G}_T$ を素数位数 $p$ の巡回群，$g_1 \in \mathbb{G}_1$，$g_2 \in \mathbb{G}_2$ を生成元とし，双線形写像 $e$ を $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ と定義する．$\mathbb{G}_1 \neq \mathbb{G}_2$ の場合を非対称ペアリング，$\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$ の場合を対称ペアリングと定義する．

### 2.2 江村らの KGC に対して安全な ID ベース暗号

本節では非対称ペアリングを用いた場合のアルゴリズムを記述する．以下 (**Sig.KeyGen**, **Sig.Sign**, **Sig.Verify**) を署名アルゴリズムとする．

Setup($1^\lambda$)： $g_1 \leftarrow \mathbb{G}_1$，$g_2 \leftarrow \mathbb{G}_2$ を選び，params = ($\mathbb{G}_1$, $\mathbb{G}_2$, $g_1$, $g_2$, H) を出力する．ここで $\mathrm{H} : \{0, 1\}^* \to \mathbb{G}_1$ はハッシュ関数である．

KGC.KeyGen(params)： $x \leftarrow \mathbb{Z}_p$ を選び，$Y_1 = {g_1}^x$，$Y_2 = {g_2}^x$ を計算する．マスター公開鍵 mpk $\leftarrow (Y_1, Y_2)$ とマスター秘密鍵 msk $\leftarrow x$ を出力する．

ICA.KeyGen(params)：
$(\mathrm{vk}_{\mathrm{Sig}}, \mathrm{sk}_{\mathrm{Sig}}) \leftarrow$ **Sig.KeyGen**($1^\lambda$) を実行し，証明書検証鍵 vk $\leftarrow \mathrm{vk}_{\mathrm{sig}}$ と証明書生成鍵 ik $\leftarrow \mathrm{ik}_{\mathrm{sig}}$ を出力する．

ICA.Cert(vk, ik, ID)： $u_{\mathrm{ID}} = \mathrm{H}(\mathrm{ID})$ を計算する．次に $y_{\mathrm{ID},1} \leftarrow \mathbb{Z}_p$ を選び，$u_{\mathrm{ID},1} = {g_1}^{y_{\mathrm{ID},1}}$ と $u_{\mathrm{ID},2} = u_{\mathrm{ID}} u_{\mathrm{ID},1} \in \mathbb{G}_1$ を計算する．さらに $u_{\mathrm{ID},2}$ に対し ik $= \mathrm{sk}_{\mathrm{Sig}}$ を用いて $\sigma_{\mathrm{Sig}} \leftarrow$ **Sig.Sign**($\mathrm{sk}_{\mathrm{Sig}}$, $u_{\mathrm{ID},2}$) を計算する．最後に証明書 cert $= (u_{\mathrm{ID},2}, \sigma_{\mathrm{Sig}})$ とトラップドア情報 td $= y_{\mathrm{ID},1}$ を出力する．

IBE.Enc(mpk, ID, M)： $\mathrm{M} \in \mathbb{G}_T$ とする． $u_{\mathrm{ID}} = \mathrm{H}(\mathrm{ID})$ を計算し，$s \leftarrow \mathbb{Z}_p$ を選び，$\mathrm{c}_0 = {g_2}^s$ および $\mathrm{c}_1 = \mathrm{M} \cdot e(u_{\mathrm{ID}}, Y_2)^s$ を計算する．最後に暗号文 ct $= (\mathrm{c}_0, \mathrm{c}_1)$ を出力する．

IBE.Dec(mpk, $\mathrm{sk}_{\mathrm{ID}}$, ct)： ct $= (\mathrm{c}_0, \mathrm{c}_1)$ に対し，$\mathrm{M} = \mathrm{c}_1 / e(\mathrm{sk}_{\mathrm{ID}}, \mathrm{c}_0)$ を計算し，平文 M を出力する．

⟨Key-issuing Protocol⟩：
ObtainKey(mpk, ID, cert, td), IssueKey(mpk, msk, vk): ユーザと KGC 間で実行されるインタラクティブ鍵生成プロトコルは 2 つのアルゴリズム ObtainKey と IssueKey から成る．ユーザは ObtainKey アルゴリズムを, KGC は IssueKey アルゴリズムをそれぞれ実行する．

User : (mpk, ID, cert, td) を入力とし，第一ラウンドメッセージ $\mathrm{M}_{\mathrm{user}} = (u_{\mathrm{ID},2}, \sigma_{\mathrm{Sig}})$ を KGC に送信する．

KGC： (mpk, msk, vk) およびユーザから送信された $\mathrm{M}_{\mathrm{user}}$ を入力とし，

Implementation and Evaluation on an Identity-Based Encryption Scheme with Security against the KGC

† Sato Yuta, Tokai University
‡ Emura Keita, National Institute of Information and Communications Technology
§ Ohigashi Toshihiro, Tokai University/National Institute of Information and Communications Technology

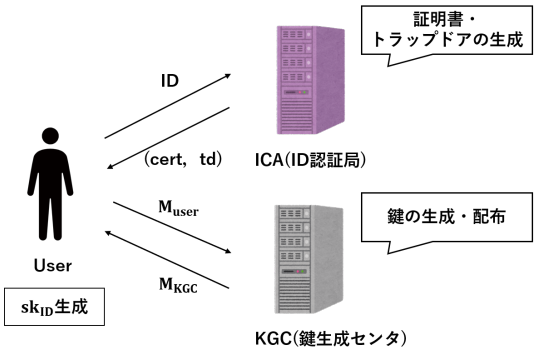


図 1　鍵生成センタに対して安全な ID ベース暗号

vk = $\mathrm{vk}_{\mathrm{Sig}}$，$\mathrm{M}_{\mathrm{user}} = (u_{\mathrm{ID},2}, \sigma_{\mathrm{Sig}})$ に対し，**Sig.Verify**$(\mathrm{vk}_{sig}, u_{\mathrm{ID},2}, \sigma_{sig}) = \perp$ の場合，$\mathrm{M}_{\mathrm{KGC}} = \perp$ とし，$\mathrm{M}_{\mathrm{KGC}}$ をユーザに送信する．$\perp$ でない場合，mpk = $(Y_1, Y_2)$ および msk = $x$ を用いて $y_{\mathrm{ID},2} = {u_{\mathrm{ID},2}}^x$ を計算する．最後に第二ラウンドメッセージ $\mathrm{M}_{\mathrm{KGC}} = y_{\mathrm{ID},2}$ をユーザに送信する．
User： $\mathrm{M}_{\mathrm{KGC}} = \perp$ の場合，$\perp$ を出力する．$\perp$ でない場合は，td $= y_{\mathrm{ID},1}$ および $\mathrm{M}_{\mathrm{KGC}} = y_{\mathrm{ID},2}$ を用いて，$\mathrm{sk}_{\mathrm{ID}} = y_{\mathrm{ID},2} \cdot {Y_1}^{-y_{\mathrm{ID},1}}$ を計算する．ここで $\mathrm{sk}_{\mathrm{ID}} = \mathrm{H}(\mathrm{ID})^x$ が成り立つ．

## 3　実装評価

本実装では，対称/非対称ペアリングをサポートしている C 言語ライブラリ PBC[*1](ver pbc-0.5.14) を用いた．jPBC[5] にてサポートされている `pairingparametergenerator` API を用いて 128 ビット安全性を満たすパラメータを生成した．対称ペアリング (TypeA) の場合は楕円曲線の位数を 256 ビット，拡大体のサイズは 3072 ビットとした．非対称ペアリング (TypeF) の場合は楕円曲線の位数を 462 ビット，拡大体のサイズは 5544 ビットとした．また署名方式 (**Sig.KeyGen**，**Sig.Sign**，**Sig.Verify**) として，OpenSSL ライブラリ[*2](ver OpenSSL-1.0.1t) で提供されている NID_X9_62_prime256v1 曲線を使用した ECDSA を利用した．

実験環境は ICA，KGC として CPU : Intel(R) Core(TM) i7-6950X @ 3.00GHz，RAM : 64GB，User が CPU : Intel(R) Core(TM) i9-9920X @ 3.50GHz，RAM : 64GB を用い，ネットワークとして LAN 内に User，ICA，KGC を配置した．

本実験では対称ペアリングを用いた場合と非対称ペアリングを用いた場合のそれぞれの各行程を 100 回実行し，処理時間の平均値を求めた結果を表 1 に示す．結果から通信時間はおよそ 138 ミリ秒，計算は各工程が 160 ミリ秒未満，全工程を処理しても 800 ミリ秒かからないことがわかった．

表 1　評価実験の結果 [msec]

| | | 対称ペアリング | 非対称ペアリング | 計算/通信 |
|---|---|---|---|---|
| ICA.Cert | ID 送信，(cert,td) 取得 | 135.64 | 137.38 | 通信 |
| | (cert,td) 生成 | 158.34 | 117.02 | 計算 |
| Key-issuing protocol | $\mathrm{M}_{\mathrm{user}}$生成 | 0.22 | 0.23 | 計算 |
| | $\mathrm{M}_{\mathrm{user}}$ 送信，$\mathrm{M}_{\mathrm{KGC}}$取得 | 137.97 | 137.72 | 通信 |
| | $\mathrm{M}_{\mathrm{KGC}}$生成 | 102.17 | 117.57 | 計算 |
| | $\mathrm{sk}_{\mathrm{ID}}$ 生成 | 42.01 | 47.92 | 計算 |
| Encrypt /Decrypt | 暗号化 | 81.07 | 140.88 | 計算 |
| | 復号 | 37.03 | 97.20 | 計算 |

## 4　まとめ

江村らが提案した KGC に対して安全な ID ベース暗号について，ペアリングベースの方式を実装し，ICA や KGC の通信時間を含めて性能評価を行った．表 1 より，暗号的処理よりも通信により時間を要することがわかった．なおこれら通信を要する手順は秘密鍵生成に関するもののみであり，これらは一度実行すればよいことに注意されたい．そのため，Boneh-Franklin ID ベース暗号に KGC に対する安全性を付与することへの効率性の影響は小さいと結論付けられる．また，本実験では通常効率的なはずの非対称ペアリングを用いた方式の方が非効率であるという結果となった．これは mcl ライブラリ[*3]で使用されている楕円曲線のように最適化されたものではないため遅くなっていると考えられる．そのため，mcl ライブラリでの実装評価を今後の課題とする．また江村らの方式は，送信者は暗号化のために ID を知る必要がある一方で KGC は ID を知らず，かつ ID がランダムに選ばれるという仮定の下で安全性が証明されている．そのため，この仮定に即した具体的な応用の提案も今後の課題とする．

## 参考文献

[1] A. Shamir, “Identity-Based Cryptosystems and Signature Schemes,” In CRYPTO 1984, pp. 47–53, 1984.
[2] K. Emura, S. Katsumata, and Y. Watanabe, “Identity-Based Encryption with Security Against the KGC: A Formal Model and Its Instantiation from Lattices,” In ESORICS 2019, pp. 113–133, 2019.
[3] K. Emura, S. Katsumata, and Y. Watanabe, “Identity-Based Encryption with Security against the KGC: A Formal Model and Its Instantiations,” Cryptology ePrint Archive, Report 2019/1384, 32 pages, 2019.
[4] D. Boneh, M. Franklin, “Identity-Based Encryption from the Weil Pairing,” In CRYPTO 2001, pp. 213–229, 2001.
[5] A. D. Caro and V. Iovino, “jpbc: Java pairing based cryptography,” In IEEE ISCC 2011, pp. 850–855, 2011.

[*1] `https://crypto.stanford.edu/pbc/`
[*2] `https://www.openssl.org/`
[*3] `https://github.com/herumi/mcl/`