Implementation of a Strongly Robust Identity-Based Encryption Scheme over Type-3 Pairings[*]

Hiroshi Okano[¶], Keita Emura[§], Takuya Ishibashi[¶, §], Toshihiro Ohigashi[¶, §], and Tatsuya Suzuki[†]

[¶]Tokai University, Japan.
[§]National Institute of Information and Communications Technology (NICT), Japan.
[†]University of Tsukuba, Japan.

April 28, 2020

Abstract

Identity-based encryption (IBE) is a powerful mechanism for maintaining security. However, systems based on IBE are unpopular when compared with those of the public-key encryption (PKE). In our opinion, one of the reasons is a gap between theory and practice. For example, a generic transformation of weakly/strongly robust IBE from any IBE has been proposed by Abdalla et al., no robust IBE scheme is explicitly given. This means that, theoretically, anyone can construct a weakly/strongly robust IBE scheme by employing this transformation. However, this seems not easily applicable to non-cryptographers. In this paper, we first introduce the Gentry IBE scheme constructed over Type-3 pairings by employing the transformation proposed by Abe et al., and second we explicitly give strongly/weakly robust Gentry IBE schemes by employing the Abdalla et al. transformation. Finally, we show its implementation result and show that we can add strong robustness to the Gentry IBE scheme with a very few additional costs. We employ the mcl library to support a Barreto-Naehrig curve defined over the 462-bit prime. The encryption requires about 5 ms, whereas the decryption requires about 9 ms.

1 Introduction

1.1 Background

Public-key encryption (PKE) is an important mechanism for maintaining security in several organizations. Identity-based encryption (IBE) [52] is an extension of PKE in which an arbitrary string,

[*]An extended abstract appears in the Seventh International Symposium on Computing and Networking (CANDAR 2019) [45]. After publishing the conference version, the mcl library v1.00 (released on September 30, 2019) supports functions for computing multi-scalar multiplications (`mclBnG1_mulVec`, `mclBnG2_mulVec`, and `mclBnGT_powVec`) that were not employed in our implementation. In this version, we employed these functions and re-implement IBE schemes (Section 5). In addition to this, we found that our encodings given in [45] do not work well (See Appendix), and thus we employ the KEM/DEM framework where KEM stands for key encapsulation mechanism and DEM stands for data encapsulation mechanism. For the DEM part, we employ AES-GCM and gave new implementation results due to this modification. Moreover, we add an application of robust IBEs to searchable encryption (Section 6). This work was done when the fifth author, Tatsuya Suzuki, was a master student at the Tokai University, Japan, and was a research assistant at the National Institute of Information and Communications Technology (NICT), Japan. He is supported by a JSPS Research Fellowship for Young Scientists.

such as email, location information, and biometrics information, is considered to be a public key, and is standardized by the ISO/IEC 18033-5 and IEEE P1363.3. The conventional PKE requires public-key infrastructure systems, whereas IBE does not require such systems. However, IBE can offer secure systems, which are not achievable using PKE. For example, IBE can maintain IoT security as revealed by [2, 8, 40, 50]. Micro Focus [41] launched a secure email service based on IBE and insisted that "*IBE can be used to build security systems that are more dynamic, lightweight, and scalable*".

Further, the security systems based on IBE are not popular when compared with PKE-based systems. A gap between theory and practice is one of the reasons for the non-universal applicability of IBE. Regardless, several IBE security definitions are proposed, which include indistinguishability against plaintext/ciphertext attacks (IND-CPA/CCA) [17], selective/adaptive security [15], anonymity against chosen plaintext/ciphertext attacks (ANON-CPA/CCA) [18], and weak/strong robustness [4, 5]. However, software engineers face challenges while implementing the IBE schemes integrated into the security systems since these security properties are not easily selected according to the functionalities or models of the systems if engineers are not familiar with cryptography.

For example, Abdalla et al. [4, 5] introduced robustness which seems attractive to be employed in such systems. Informally, robustness guarantees that a ciphertext C of a plaintext M encrypted by an identity ID is not decrypted by a secret key of a different identity ID' ($\mathsf{ID} \neq \mathsf{ID}'$). This causes a problem when the underlying IBE scheme is anonymous. Here, anonymity indicates that the ID information is not revealed from C. Although a receiver's privacy is protected in an anonymous IBE, nobody can recognize the decryption keys that are to be used, even the corresponding receiver. We remark that, "C is not decrypted" contains the situation that the decryption algorithm outputs a plaintext M' where $M \neq M'$. Thus, if a random string such as password is encrypted, then one may receive M' as own password. If the underlying IBE scheme is robust, then the decryption algorithm outputs a special rejection symbol $\perp$ if an inaccurate secret key is used. Abdalla et al. defined two notions of robustness, weak robustness and strong robustness. Refer to Section 4 for details. Although robustness seems naturally holds in usual IBE schemes, this it not true, and is not guaranteed in usual IBE schemes in general. Thus, Abdalla et al. proposed the generic transformation of weakly/strongly robust IBE from any IBE. That is, anyone can *theoretically* construct a weakly/strongly robust IBE scheme by employing this transformation. However, this seems not easily applicable to non-cryptographers. In addition, no robust IBE scheme is explicitly given in their paper.[1]

Moreover, it is difficult to select which parameters, e.g., elliptic curves, should be used for implementation based on a state-of-the-art method for solving the discrete logarithm (DL) problem [12, 39] or based on the type of pairings. These situations may prevent the applicability of IBE.[2]

1.2 Our Contribution

In this paper, we focus on the robustness [4, 5] because no concrete IBE scheme offers robustness. We explicitly present strongly/weakly robust IBE schemes. Moreover, inspired by [49], we present the implementation results.

We employ the Gentry IBE scheme [32] as the underlying IBE, which is adaptively IND/ANON-CCA secure. Because the Abdalla et al. transformation preserves CCA security and anonymity, we can add weak/strong robustness of the Gentry IBE scheme without detracting the CCA security

[1]As a remark, they explicitly gave a strongly robust CCA secure Cramer-Shoup PKE scheme in their paper.

[2]Aoki [9] mentioned that majority of the cryptographic algorithms that guarantee security require assumptions, but it is sometimes not clearly written.

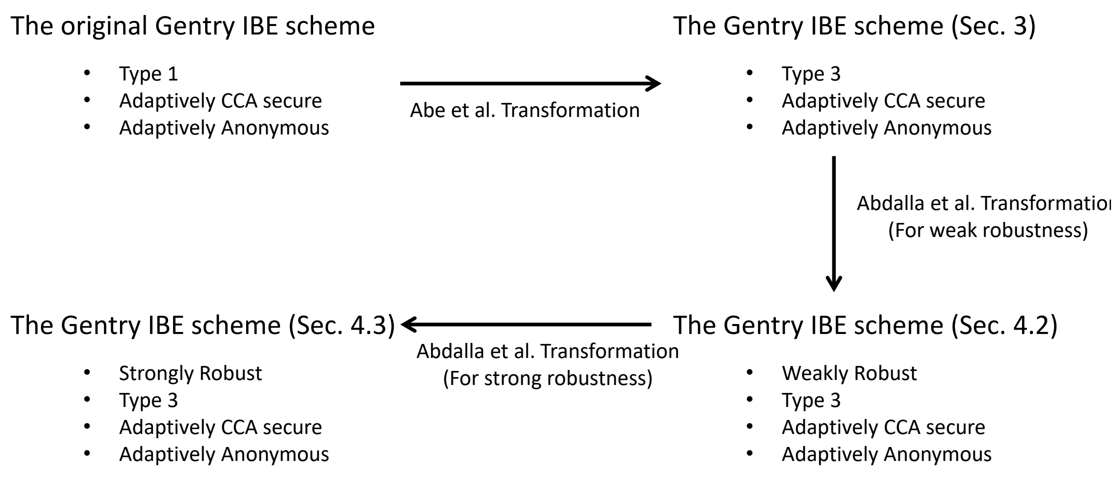

Figure 1: Overview of the Transformations

and anonymity. Unfortunately, the original Gentry IBE scheme is constructed over Type-1 pairings although the construction of schemes over Type-3 pairings is the most efficient (refer to Section 2 for details). Thus, we employ the transformation proposed by Abe et al. [6,7] to the original Gentry IBE scheme, constructed over Type-3 pairings. Then, we employ the transformation proposed by Abdalla et al. Figure 1 reveals an overview of the transformations.

For the implementation of the strongly robust Gentry IBE scheme over Type-3 pairings, we employ the mcl library [42] that supports Type-3 pairings with the parameters that have been selected by considering the state-of-the-art for solving discrete logarithm problem [12, 39]. The encryption requires about 5 ms, whereas the decryption requires about 9 ms.

2 Bilinear Groups

In this section, we define bilinear groups and types as follows:

Definition 1 (Bilinear Groups) *Let p be a λ-bit prime, $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ be the groups of order p; $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear map, and g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. We require bilinearlity: for all $h_1 \in \mathbb{G}_1$, $h_2 \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_p$, $e(h_1^a, h_2^b) = e(h_1, h_2)^{ab}$, and non-degeneracy: $e(g_1, g_2) \neq 1$ hold.*

If $\mathbb{G}_1 = \mathbb{G}_2$, the map e is called *symmetric*, and we call the setting Type 1. If $\mathbb{G}_1 \neq \mathbb{G}_2$, the map e is called *asymmetric*. If there is an efficient computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$, we call the setting Type 2 (e.g., Miyaji-Nakabayashi-Takano (MNT) curves [43]). If no efficient isomorphism is known between $\mathbb{G}_1$ and $\mathbb{G}_2$, we call the setting Type 3 (e.g., Barreto-Naehrig (BN) curves [14] or Barreto-Lynn-Scott (BLS) curves [13]). Alternatively, we use the terms Type-i pairings/curves or Type-i bilinear groups for $i = 1, 2, 3$.

Currently, constructing schemes in the Type-3 setting is the most efficient. Refer to [31] for details. From the implementation point of view, Type-3 curves are the most reasonable choice unless there is a specific reason, e.g., employing symmetric curves as Gap-DH groups [36]. The mcl library implements a BN curve [14] defined over the 462-bit prime $r(z) := 36z^4+36z^3+24z^2+6z+1$ and has the order $p(z) = 36z^4 + 36z^3 + 18z^2 + 6z + 1$ where $z = 2^{114} + 2^{101} - 2^{14} - 1$. Currently, this curve is believed to provide 128-bit security, denoted as `BN462` in this paper.

Remark. The symmetric pairing is known as a decisional Diffie-Hellman (DDH) solver. That is, for a tuple $(g, g^a, g^b, g^c) \in \mathbb{G}_1^4$, one can check whether $c = ab$ or not by checking whether $e(g^a, g^b) = e(g, g^c)$. However, in the computational Diffie-Hellman (CDH) problem, computing g^{ab} from $(g, g^a, g^b) \in \mathbb{G}_1^3$ is still believed to be difficult. Further, the DDH assumption holds in case of both $\mathbb{G}_1$ and $\mathbb{G}_2$ in Type-3 bilinear groups, which we call the symmetric external Diffie-Hellman (SXDH) assumption. We also state that the DDH assumption holds in $\mathbb{G}_1$ and does not hold in $\mathbb{G}_2$ in the Type-2 setting, which we call the external Diffie-Hellman (XDH) assumption.

3 The Gentry IBE scheme over Type-3 Pairings

As mentioned in the previous section, Type-3 pairings are better from an efficiency point of view. Unfortunately, the original Gentry IBE scheme is constructed in Type-1 pairings. Thus, in this section, we introduce the Gentry IBE scheme constructed over Type-3 pairings that is obtained by employing the Abe et al. transformation [6, 7] to the original Gentry IBE scheme. Before giving that, we define the syntax of IBE as follows.

Definition 2 (Syntax of IBE) *An IBE scheme* IBE *consists of the following four algorithms,* $\mathsf{IBE.Setup}$*,* $\mathsf{IBE.KeyGen}$*,* $\mathsf{IBE.Enc}$ *and* $\mathsf{IBE.Dec}$*:*

- $\mathsf{IBE.Setup}(1^\lambda)$*: The setup algorithm takes as an input a security parameter* $\lambda \in \mathbb{N}$*, and returns a public parameter* params *and a master key* msk*.*
- $\mathsf{IBE.KeyGen}(\mathsf{params}, \mathsf{msk}, \mathsf{ID})$*: The key extract algorithm takes as input an identity* ID *and* msk*, and returns a secret key* $\mathsf{sk}_{\mathsf{ID}}$ *corresponding to* ID*.*
- $\mathsf{IBE.Enc}(\mathsf{params}, \mathsf{ID}, M)$*: The encryption algorithm takes as input* params*,* ID*, a plaintext* M*, and returns a ciphertext* C*.*
- $\mathsf{IBE.Dec}(\mathsf{params}, \mathsf{sk}_{\mathsf{ID}}, C)$*: The decryption algorithm takes as input* params*,* $\mathsf{sk}_{\mathsf{ID}}$*, and* C*, and returns a plaintext* M *or a reject symbol* $\perp$*.*

Next, we introduce the Gentry IBE scheme over Type-3 pairings as follows. In the construction, public parameters h_1, h_2, and h_3, and secret keys $(h_{\mathsf{ID},1}, h_{\mathsf{ID},2}, h_{\mathsf{ID},3})$ are elements on $\mathbb{G}_2$ whereas these are elements on $\mathbb{G}_1$ in the original construction.

The Gentry IBE Scheme over Type-3 Pairings:

- $\mathsf{IBE.Setup}(1^\lambda)$: Choose a Type-3 bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ with λ-bit prime p where g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. Choose $\alpha \xleftarrow{\$} \mathbb{Z}_p$ and $h_1, h_2, h_3 \xleftarrow{\$} \mathbb{G}_2$, compute $g_1' = g_1^\alpha$, and output $\mathsf{params} = (g_1, g_1', g_2, h_1, h_2, h_3, H)$ where $H : \{0,1\}^* \to \mathbb{Z}_p$ is a universal one-way hash function, and $\mathsf{msk} = \alpha$.
- $\mathsf{IBE.KeyGen}(\mathsf{params}, \mathsf{msk}, \mathsf{ID})$: For identity $\mathsf{ID} \in \mathbb{Z}_p$, for $i = 1, 2, 3$ choose $r_{\mathsf{ID},i} \xleftarrow{\$} \mathbb{Z}_p$, compute $h_{\mathsf{ID},i} = (h_i g_2^{r_{\mathsf{ID},i}})^{1/(\alpha-\mathsf{ID})} \in \mathbb{G}_2$, and output $\mathsf{sk}_{\mathsf{ID}} = (r_{\mathsf{ID},i}, h_{\mathsf{ID},i})_{i=1}^3$.

- $\mathsf{IBE.Enc}(\mathsf{params}, \mathsf{ID}, M)$: Let $M \in \mathbb{G}_T$ be a plaintext to be encrypted. Choose $s \xleftarrow{\$} \mathbb{Z}_p$, compute $C_1 = {g'_1}^s g_1^{-s\mathsf{ID}}$, $C_2 = e(g_1, g_2)^s$, $C_3 = Me(g_1, h_1)^{-s}$, $\beta = H(C_1, C_2, C_3)$, and $C_4 = e(g_1, h_2)^s e(g_1, h_3)^{s\beta}$, and output $C = (C_1, C_2, C_3, C_4)$.

- $\mathsf{IBE.Dec}(\mathsf{params}, \mathsf{sk}_{\mathsf{ID}}, C)$: Compute $\beta = H(C_1, C_2, C_3)$. If $C_4 \neq e(C_1, h_{\mathsf{ID},2} h_{\mathsf{ID},3}^{\beta}) C_2^{r_{\mathsf{ID},2} + r_{\mathsf{ID},3}\beta}$, then output $\perp$. Otherwise, compute $M = C_3 e(C_1, h_{\mathsf{ID},1}) C_2^{r_{\mathsf{ID},1}}$ and output M.

The security of original Gentry IBE scheme relies on the truncated q-ABDHE assumption where ABDHE stands for augmented bilinear Diffie-Hellman exponent. Since the assumption is defined over Type-1 settings, we also need to duplicate the assumption via the Abe et al. transformation as follows. Interestingly, we do not have to duplicate not all elements. We duplicate generators and $(g_1^{\alpha}, g_2^{\alpha})$ only.

Definition 3 (The truncated q-ABDHE Assumption over Type-3 Pairings) *Let $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ be a Type-3 bilinear groups with λ-bit prime order p. Let $g_1, g'_1 \in \mathbb{G}_1$ and $g_2, g'_2 \in \mathbb{G}_2$ be generators, $\alpha \xleftarrow{\$} \mathbb{Z}_p$, and $Z \xleftarrow{\$} \mathbb{G}_T$. For all probabilistic polynomial-time adversaries $\mathcal{A}$, we define the advantage $Adv_{\mathcal{A}}^{\mathsf{ABDHE}}(\lambda) := \Pr[\mathcal{A}(g_1, g'_1, g_2, g'_2, g_1^{\alpha}, g_2^{\alpha}, g_2^{\alpha^2}, \dots, g_2^{\alpha^q}, {g'_1}^{\alpha^{q+2}}, e(g_1^{\alpha^{q+1}}, g'_2))] - \Pr[\mathcal{A}(g_1, g'_1, g_2, g'_2, g_1^{\alpha}, g_2^{\alpha}, g_2^{\alpha^2}, \dots, g_2^{\alpha^q}, {g'_1}^{\alpha^{q+2}}, Z)]$. We say that the truncated q-ABDHE assumption holds if $Adv_{\mathcal{A}}^{\mathsf{ABDHE}}(\lambda)$ is negligible.*

4 Robust Gentry IBE Schemes

In this section, we employ the Abdalla et al. transformation [4, 5] to the Gentry IBE scheme given in Section 3.

4.1 Two Robustness Notions

Abdalla et al. defined two notions of robustness, weak robustness and strong robustness. Weak robustness ensures that $M' \neq \perp$ holds with negligible probability where M' is defined as $M' := \mathsf{IBE.Dec}(\mathsf{params}, \mathsf{sk}_{\mathsf{ID}'}, C_{\mathsf{ID}})$ where $\mathsf{sk}_{\mathsf{ID}'} := \mathsf{IBE.KeyGen}(\mathsf{params}, \mathsf{msk}, \mathsf{ID}')$ and $C_{\mathsf{ID}} := \mathsf{IBE.Enc}(\mathsf{params}, \mathsf{ID}, M)$. Here, $(M, \mathsf{ID}, \mathsf{ID}')$ is selected by an adversary with the condition $M \neq \perp$ and $\mathsf{ID} \neq \mathsf{ID}'$. Further, we remark that the ciphertext of ID on M is honestly computed. Strong robustness ensures that $M \neq \perp$ and $M' \neq \perp$ hold with negligible probability where M and M' are defined as $M := \mathsf{IBE.Dec}(\mathsf{params}, \mathsf{sk}_{\mathsf{ID}}, C)$ and $M' := \mathsf{IBE.Dec}(\mathsf{params}, \mathsf{sk}_{\mathsf{ID}'}, C)$ where $\mathsf{sk}_{\mathsf{ID}} := \mathsf{IBE.KeyGen}(\mathsf{params}, \mathsf{msk}, \mathsf{ID})$ and $\mathsf{sk}_{\mathsf{ID}'} := \mathsf{IBE.KeyGen}(\mathsf{params}, \mathsf{msk}, \mathsf{ID}')$. Here, $(C, \mathsf{ID}, \mathsf{ID}')$ is selected by an adversary with the condition $\mathsf{ID} \neq \mathsf{ID}'$. We remark that the ciphertext C is adversarially computed.

4.2 Weakly Robust Gentry IBE Scheme

First, we denote a weakly robust Gentry IBE scheme. We employ the Abdalla et al. transformation for weak robustness. Intuitively, the auxiliary information $K \xleftarrow{\$} \{0,1\}^{\lambda}$ is chosen and is contained in public parameters. The encryption algorithm encrypts the plaintext $M||K$, and the decryption algorithm checks whether K is appropriately recovered. We remark that $M||K$ is not an element of $\mathbb{G}_T$ since $K \in \{0,1\}^{\lambda}$. Thus, we need to consider an encoding for K and how to recover M and K from the decryption result. Thus, we employ the KEM/DEM framework and $M||K$ is encrypted by AES-GCM. Let $\mathsf{AES.Enc}(\cdot, \cdot)$ and $\mathsf{AES.Dec}(\cdot, \cdot)$ be encryption and decryption algorithms whose the first input is a key and the second input is either a plaintext or a ciphertext.

Weakly Robust Gentry IBE Scheme:

- $\mathsf{IBE.Setup}(1^\lambda)$: Choose a Type-3 bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ with λ-bit prime order p where g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. Choose $K \xleftarrow{\$} \{0,1\}^\lambda$, $\alpha \xleftarrow{\$} \mathbb{Z}_p$, and $h_1, h_2, h_3 \xleftarrow{\$} \mathbb{G}_2$, compute $g_1' = g_1^\alpha$, and output $\mathsf{params} = (K, g_1, g_1', g_2, h_1, h_2, h_3, H, H_{\mathsf{AES}})$ where $H : \{0,1\}^* \to \mathbb{Z}_p$ is a universal one-way hash function, $H_{\mathsf{AES}} : \mathbb{G}_T \to \{0,1\}^{256}$ be a collision resistant hash function, and $\mathsf{msk} = \alpha$.

- $\mathsf{IBE.KeyGen}(\mathsf{params}, \mathsf{msk}, \mathsf{ID})$: For identity $\mathsf{ID} \in \mathbb{Z}_p$, for $i = 1, 2, 3$ choose $r_{\mathsf{ID},i} \xleftarrow{\$} \mathbb{Z}_p$, compute $h_{\mathsf{ID},i} = (h_i g_2^{r_{\mathsf{ID},i}})^{1/(\alpha - \mathsf{ID})} \in \mathbb{G}_2$, and output $\mathsf{sk}_{\mathsf{ID}} = (r_{\mathsf{ID},i}, h_{\mathsf{ID},i})_{i=1}^3$.

- $\mathsf{IBE.Enc}(\mathsf{params}, \mathsf{ID}, M)$: Choose $s \xleftarrow{\$} \mathbb{Z}_p$, compute $C_1 = {g_1'}^s g_1^{-s\mathsf{ID}}$, $C_2 = e(g_1, g_2)^s$, $K_{\mathsf{AES}} = H_{\mathsf{AES}}(e(g_1, h_1)^{-s})$, $C_3 = \mathsf{AES.Enc}(K_{\mathsf{AES}}, M||K)$, $\beta = H(C_1, C_2, C_3)$, and $C_4 = e(g_1, h_2)^s e(g_1, h_3)^{s\beta}$, and output $C = (C_1, C_2, C_3, C_4)$.

- $\mathsf{IBE.Dec}(\mathsf{params}, \mathsf{sk}_{\mathsf{ID}}, C)$: Compute $\beta = H(C_1, C_2, C_3)$. If $C_4 \neq e(C_1, h_{\mathsf{ID},2} h_{\mathsf{ID},3}^\beta) C_2^{r_{\mathsf{ID},2} + r_{\mathsf{ID},3}\beta}$, then output $\perp$. Otherwise, compute $M||K' = \mathsf{AES.Dec}(H_{\mathsf{AES}}((e(C_1, h_{\mathsf{ID},1}) C_2^{r_{\mathsf{ID},1}})^{-1}), C_3)$. Output $\perp$ if $K' \neq K$, and output M, otherwise.

4.3 Strongly Robust Gentry IBE Scheme

Next, we denote a strongly robust Gentry IBE scheme. We employ the Abdalla et al. transformation to the weakly robust Gentry IBE scheme as the building block. The transformation additionally introduces a commitment scheme $\mathcal{CMT} = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Ver})$. The parameter generation algorithm CPG takes as input a security parameter, and outputs the public parameter *cpar*. The committal algorithm Com takes as input *cpar* and data x, and outputs a commitment *com* and a decommittal key *dec*. The verification algorithm Ver takes as input $(cpar, x, com, dec)$, and outputs 1 or 0. We employ the Pedersen commitment scheme [46]. Let $\mathbb{G}_1$ be a group with a prime order p. The CPG algorithm chooses $g', h' \xleftarrow{\$} \mathbb{G}_1$. The Com algorithm chooses $dec \xleftarrow{\$} \mathbb{Z}_p$ and computes $com = {g'}^x {h'}^{dec}$. The Ver algorithm outputs 1 if $com = {g'}^x {h'}^{dec}$ and 0 otherwise. The Pedersen commitment scheme is perfectly hiding, and is computationally binding under the DL assumption. Since we do not have to assume the hardness of the DDH problem, we implement the scheme on $\mathbb{G}_1$ regardless of types of pairings.

In the scheme, a public key ID is committed. The encryption algorithm encrypts a plaintext $M||dec$ and the commitment *com* is contained in the ciphertext. Because the weakly robust IBE scheme is used as the building block, the encryption algorithm encrypts a plaintext $M||K||dec$ in the actual scheme. The decryption algorithm checks whether *com* is a commitment of ID or not. As in the weak one, we remark that $M||K||dec$ is not an element of $\mathbb{G}_T$ and thus we employed the KEM/DEM framework.

Strongly Robust Gentry IBE Scheme:

- $\mathsf{IBE.Setup}(1^\lambda)$: Choose a Type-3 bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ with λ-bit prime order p where g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. Choose $K \xleftarrow{\$} \{0,1\}^\lambda$, $g', h' \xleftarrow{\$} \mathbb{G}_1$, $\alpha \xleftarrow{\$} \mathbb{Z}_p$, and $h_1, h_2, h_3 \xleftarrow{\$} \mathbb{G}_2$, compute $g_1' = g_1^\alpha$, and output $\mathsf{params} = (K, g', h', g_1, g_1', g_2, h_1, h_2, h_3, H, H_{\mathsf{AES}}))$ where $H : \{0,1\}^* \to \mathbb{Z}_p$ is a universal one-way hash function, $H_{\mathsf{AES}} : \mathbb{G}_T \to \{0,1\}^{256}$ be a collision resistant hash function, and $\mathsf{msk} = \alpha$.

Table 1: Comparison among Gentry IBE Schemes over Type-3 Pairings (millisec)

	Normal (Sec. III)	Weakly Robust (Sec. IV)	Strongly Robust (Sec. IV)
IBE.Setup	14.66	14.63	15.39
IBE.KeyGen	4.54	4.54	4.49
IBE.Enc	4.66	4.63	5.19
IBE.Dec	8.49	8.46	9.00

- $\mathsf{IBE.KeyGen}(\mathsf{params}, \mathsf{msk}, \mathsf{ID})$: For identity $\mathsf{ID} \in \mathbb{Z}_p$, for $i = 1, 2, 3$ choose $r_{\mathsf{ID},i} \xleftarrow{\$} \mathbb{Z}_p$, compute $h_{\mathsf{ID},i} = (h_i g_2^{r_{\mathsf{ID},i}})^{1/(\alpha-\mathsf{ID})} \in \mathbb{G}_2$, and output $\mathsf{sk}_{\mathsf{ID}} = (\mathsf{ID}, (r_{\mathsf{ID},i}, h_{\mathsf{ID},i})_{i=1}^{3})$.

- $\mathsf{IBE.Enc}(\mathsf{params}, \mathsf{ID}, M)$: Choose $dec \xleftarrow{\$} \mathbb{Z}_p$ and compute $com = g'^{\mathsf{ID}} h'^{dec}$. Choose $s \xleftarrow{\$} \mathbb{Z}_p$, compute $C_1 = g_1'^{s} g_1^{-s\mathsf{ID}}$, $C_2 = e(g_1, g_2)^s$, $K_{\mathsf{AES}} = H_{\mathsf{AES}}(e(g_1, h_1)^{-s})$, $C_3 = \mathsf{AES.Enc}(K_{\mathsf{AES}}, M||K||dec)$, $\beta = H(C_1, C_2, C_3)$, and $C_4 = e(g_1, h_2)^s e(g_1, h_3)^{s\beta}$, and output $C = (com, C_1, C_2, C_3, C_4)$.

- $\mathsf{IBE.Dec}(\mathsf{params}, \mathsf{sk}_{\mathsf{ID}}, C)$: Compute $\beta = H(C_1, C_2, C_3)$. If $C_4 \neq e(C_1, h_{\mathsf{ID},2} h_{\mathsf{ID},3}^{\beta}) C_2^{r_{\mathsf{ID},2}+r_{\mathsf{ID},3}\beta}$, then output $\perp$. Otherwise, compute $M||K'||dec = \mathsf{AES.Dec}(H_{\mathsf{AES}}((e(C_1, h_{\mathsf{ID},1}) C_2^{r_{\mathsf{ID},1}})^{-1}), C_3)$. Output $\perp$ if $K' \neq K$. Otherwise, output $\perp$ if $com \neq g'^{\mathsf{ID}} h'^{dec}$, and output M, otherwise.

Remark. To the best of our knowledge, the strongest notion among several robustnesses is complete robustness defined by Farshim et al. [29].[3] They also showed that the transformation from weakly robust IBE to strongly robust IBE, employed in this paper, is already powerful enough to construct completely robust IBE. Thus, the strongly robust Gentry IBE scheme given in this paper provides complete robustness in the strict sense.

5 Implementation

In this section, we give our implementation results.

5.1 Implementation Results

Our implementation environment includes CPU: Intel(R) Core(TM) i7-6950X (3.00GHz), gcc 4.9.2, openssl 1.0.1t, and mcl v1.00. First, we denote a comparison among the Gentry IBE schemes over Type-3 pairings in Table 1. All the schemes are implemented using the mcl library and `BN462`. We add $e(g_1, g_2)$, $e(g_1, h_1)$, $e(g_1, h_2)$, and $e(g_1, h_3)$ to params in case of all the implementations. Then, no pairing computation is required in the encryption algorithm at the expense of the pre-computation costs on the setup algorithm. As presented in Table 1, the efficiencies of all the schemes are almost the same, revealing a strong robustness with little additional costs.

In Table 2, we compare the running times of the strongly robust Gentry IBE schemes to denote the effectiveness of the Type-3 curves. Here, PBC stands for the PBC library [1] (we employ pbc-0.5.14). We use the effective security system because the PBC library offers no support to elliptic curves that provide 128-bit security by default. For PBC (Type-1), we generate parameters for a symmetric bilinear group (Type A curve in PBC), defined over 1536-bit prime r using the

[3]As another robustnesses notions, Mohassel [44] defined robustness for key-encapsulation mechanisms, Farshim et al. [30] defined robustness for symmetric primitives (authenticated-encryption, message-authentication codes and pseudo-random functions), and Géraud et al. [33] defined robustness for functional encryption and digital signatures.

Table 2: Comparison of Strongly Robust Gentry IBE Schemes among Pairing Libraries (millisec)

	PBC (Type-1)	PBC (Type-3)	mcl (Type-3)
IBE.Setup	83.85	317.69	15.39
IBE.KeyGen	42.08	33.32	4.49
IBE.Enc	21.99	62.13	5.19
IBE.Dec	34.83	181.69	9.00

Table 3: Benchmarks (millisec)

	PBC (Type-1)	PBC (Type-3)	mcl (Type-3)
Exp. on $\mathbb{G}_1$	6.84	3.22	0.35
MulExp. on $\mathbb{G}_1$	9.38	4.50	0.50
Exp. on $\mathbb{G}_2$	–	5.37	0.75
MulExp. on $\mathbb{G}_2$	–	7.40	1.19
Exp. on $\mathbb{G}_T$	0.77	15.65	1.14
MulExp. on $\mathbb{G}_T$	1.43	21.71	1.75
Pairing	8.50	70.70	2.70

`PairingParametersGenerator` API supported by jPBC [22]. Here, $\mathbb{G}_T$ is a subgroup of $\mathbb{F}^*_{r^2}$. For additional information, we generate parameters for a BN curve (Type F curve in PBC) defined over 462-bit prime using the same API. The implementation results are presented in the PBC (Type-3) column. Moreover, we employ the `element_pow2_mpz` function supported by PBC and the `mclBnG1_mulVec`, the `mclBnG2_mulVec`, and the `mclBnGT_powVec` functions supported by the mcl library that compute $g^x h^y$ directly from g, h, x, and y, which is generally faster than performing two separate exponentiations (i.e., compute g^x and h^y separately, and compute $g^x h^y$). We employ them to compute $C_1 = {g'_1}^s g_1^{-s\mathsf{ID}}$, $com = {g'}^x {h'}^{dec}$, and $C_4 = e(g_1, h_2)^s e(g_1, h_3)^{s\beta}$ in the encryption algorithm.

In Table 3, we show the benchmarks. Here, MulExp is the running time of multi-scalar multiplications with the length 2, i.e, compute $g^x h^y$ directly as mentioned above. For the mcl library, MulExp. on $\mathbb{G}_1$ (resp. $\mathbb{G}_2$) is the running time of the `mclBnG1_mulVec` function (resp. the `mclBnG2_mulVec` function), and Exp. on $\mathbb{G}_T$ is the running time of the `mclBnGT_powVec` function. With respect to the exponentiations of $\mathbb{G}_1$, Type-3 curves is faster than that of Type-1 curves regardless of the library. This is a reasonable result because we set the order of $\mathbb{G}_1$ is 462-bit prime in Type-3 settings and that is 1536-bit prime in Type-1 settings. More precisely, that of mcl (Type-3) is almost 20 times faster than that of PBC (Type-1). However, that of PBC (Type-3) is just two times faster than that of PBC (Type-1). PBC (Type-1) is more efficient than mcl (Type-3) with respect to the exponentiations of $\mathbb{G}_T$. This is also a reasonable result. In PBC (Type-1), $\mathbb{G}_T$ is a subgroup of $\mathbb{F}^*_{r^2}$ where r is a 1536-bit prime. In mcl (Type-3), $\mathbb{G}_T$ is a subgroup of $\mathbb{F}^*_{r^{12}}$ where r is a 462-bit prime. In terms of the element size of $\mathbb{G}_T$, PBC (Type-1) is more efficient than mcl (Type-3) with respect to the exponentiations of $\mathbb{G}_T$. More precisely, that of PBC (Type-1) is almost 1.5 times faster than that of mcl (Type-3), and that of PBC (Type-3) is much slower among them. Although the same type is used in mcl (Type-3) and PBC (Type-3), there is a significant difference from the running time point of view. It seems the reason behind is that the parameter employed in mcl (Type-3) is optimized but that of PBC (Type-3) is randomly generated by the API. The dominant computations in IBE schemes are (Mul)Exp. on $\mathbb{G}_1$ and $\mathbb{G}_T$ since almost pairings are pre-computed in our implementations. As a reslut, PBC (Type-1) provides a more efficient implementation than that provided by PBC (Type-3). In total, mcl (Type-3) yields the most efficient implementation result due to the efficency of (Mul)Exp. on $\mathbb{G}_1$ and pairing computations, and Exp. on $\mathbb{G}_T$ is comparable to that of PBC (Type-1).

6 Application to Searchable Encryption

Although robustness itself is already an attractive property, in this section we introduce an application of robust IBE to searchable encryption. Before giving the application, we introduce the relation between consistency of searchable encryption and robustness as follows. As a well known result, public key encryption with keyword search (PEKS) [16] can be constructed from anonymous IBE [3]. Intuitively, a keyword ω to be searched is regarded as an identity of an IBE scheme, and a random plaintext R is encrypted by using ω. A secret key sk_ω is regarded as a trapdoor, and one (typically a server that has a role of searching) can check whether a ciphertext is associated to ω or not by checking the decryption result of the ciphertext using the trapdoor is R. Due to the anonymity, no information of keyword is revealed from the ciphertext.[4] Remark that the original paper [16] indicated the fixed $R = 0^{|\lambda|}$ (for a security parameter $\lambda \in \mathbb{N}$), but as mentioned in [3] the construction does not guarantee (wrong keyword) consistency where a ciphertext of ω may be searched by a trapdoor $\mathsf{sk}_{\omega'}$ with $\omega \neq \omega'$. As an application of robustness, Abdalla et al. [4, 5] showed that if the underlying IBE is robust, then this original construction above provides consistency. This result fits the following intuition where robustness guarantees that the decryption result of a ciphertext encrypted by ω is $\perp$ when $\mathsf{sk}_{\omega'}$ with $\omega \neq \omega'$ is used.

Suzuki et al. [53, 54] showed that secure-channel free PEKS with public key encryption (SCF-PEKS/PKE), which is explained later more clearly, can be constructed from anonymous IBE with a certain robustness, which they call unrestricted strong collision-freeness. Since unrestricted strong collision-freeness is weaker than complete robustness [29], and the transformation from weakly robust IBE to strongly robust IBE, employed in this paper, is already powerful enough to construct completely robust IBE. Thus, the strongly robust Gentry IBE scheme given in this paper provides unrestricted strong collision-freeness in the strict sense, and it can be employed as an underlying IBE scheme. So, in this section we explicitly present a SCF-PEKS/PKE scheme based on the strongly robust Gentry IBE scheme. Before giving the construction, we explain SCF-PEKS/PKE as follows. In PEKS, a trapdoor needs to be sent to the server via a secure channel since anyone can run the test algorithm if they obtain the trapdoor. We call secure-channel free if no secure channel is required for sending trapdoors to the server. To add the functionality, the server also has a public key and a secret key, and a ciphertext is generated both a keyword and the server public key. The test algorithm requires both a trapdoor and the server secret key. Such schemes, SCF-PEKS (which is also called designated tester PEKS), have been proposed in [11, 23, 24, 26, 27, 35, 37, 47, 48, 55]. In addition, PEKS does not provide a decryption functionality as PKE. Thus, schemes with both PEKS and PKE functionalities, which we call PEKS/PKE, have been proposed in [10,19,20,51,56]. SCF-PKES/PKE provides both these functionalities. Here there are three entities, a receiver, an encryptor, and the server. The receiver has a public key and a master secret key, generates a trapdoor for searching using the masker secret key, and sends the trapdoor to the server via a

[4] We remark that we just consider whether information of keyword is revealed from the "ciphertext" or not as a standard security requirement of PEKS. Thus, the server may obtain information of keyword when it runs the test algorithm. Especially, in some schemes (e.g., PEKS-STAT [3]) a keyword to be searched is directly contained in a trapdoor. Then, obviously the server can know what keyword is searched. Our SCF-PEKS/PKE instantiation also follows this structure due to strong robustness. More concretely, the IBE decryption algorithm needs to know the identity for checking the validity of the commitment contained in the ciphertext. How to prevent the information leakage, i.e., preventing keyword guessing attacks [28], is left as a future work.

public channel. An encryptor generates a ciphertext of both a keyword and a plaintext, and sends the ciphertext to the server. The server that has a trapdoor searches a ciphertext by using the trapdoor, and sends the ciphertext to the receiver. Finally, the decrptor decrypts the ciphertext, and obtain the plaintext. We introduce the syntax of SCF-PEKS/PKE as follows. Let $\mathcal{K}$ be the keyword space and $\mathcal{M}$ be the message space.

Definition 4 (Syntax of SCF-PEKS/PKE [53, 54]) *A SCF-PEKS/PKE scheme* $\mathsf{SCF\text{-}PEKS/PKE}$ *consists of the following six algorithms,* $\mathsf{KeyGen_S}$*,* $\mathsf{KeyGen_R}$*,* $\mathsf{Trapdoor}$*,* Enc*,* Dec *and* Test*:*

- $\mathsf{KeyGen_S}(1^\lambda)$*: The server key generation algorithm takes as input the security parameter* $\lambda \in \mathrm{N}$*, and returns a server public key* pk_{S} *and a server secret key* sk_{S}*.*
- $\mathsf{KeyGen_R}(1^\lambda)$*: The receiver key generation algorithm takes as input the security parameter* $\lambda \in \mathrm{N}$*, and returns a receiver public key* pk_{R} *and a receiver secret key* sk_{R}*.*
- $\mathsf{Trapdoor}(pk_{\mathrm{R}}, sk_{\mathrm{R}}, \omega)$*: The trapdoor generation algorithm takes as input* pk_{R}*,* sk_{R}*, and a keyword* $\omega \in \mathcal{K}$*, and returns a trapdoor* t_ω *corresponding to keyword* ω*.*
- $\mathsf{Enc}(pk_{\mathrm{S}}, pk_{\mathrm{R}}, \omega, M)$*: The encryption algorithm takes as input* pk_{R}*,* pk_{S}*,* ω*, and a message* $M \in \mathcal{M}$*, and returns a ciphertext* CT*.*
- $\mathsf{Dec}(pk_{\mathrm{R}}, sk_{\mathrm{R}}, \mathsf{CT})$*: The decryption algorithm takes as input* pk_{R}*,* sk_{R}*, and* CT*, and returns a message* M *or a reject symbol* $\perp$*.*
- $\mathsf{Test}(pk_{\mathrm{S}}, sk_{\mathrm{S}}, pk_{\mathrm{R}}, t_\omega, \mathsf{CT})$*: The test algorithm takes as input* pk_{S}*,* sk_{S}*,* pk_{R}*,* t_ω*, and* CT*, and returns 1 if* $\omega = \omega'$*, where* ω' *is the keyword which was used for computing* CT*, and 0 otherwise.*

Suzuki et al. gave a generic construction of SCF-PEKS/PKE by extending the generic construction of SCF-PEKS [25] from anonymous IBE, tag-based encryption (TBE) [38], and one-time signature (OTS). The Suzuki et al. construction is briefly described as follows. The server has a public key and a decryption key of a TBE scheme, and the receiver has a public key and a decryption key of a TBE scheme and a master public key and a master secret key of an IBE scheme. The Enc algorithm generates a ciphertext as follows. Let $(\mathsf{vk}, \mathsf{sigk})$ be a verification key and a signing key of the underlying OTS scheme. vk is regarded as a tag of the underlying TBE scheme, and a plaintext M is encrypted by using the TBE scheme using the receiver public key (let $C_{\mathsf{TBE,R}}$ be the ciphertext). A keyword ω is regarded as an identity of the underlying IBE scheme, and a plaintext R is encrypted. Here, R is computed by $R = H_R(\mathsf{vk})$ where H_R is a target-collision resistant hash function. The relation between R and vk is important for preventing the re-encryption attack. See [53, 54] for details. The IBE ciphertext is also encrypted by using the TBE scheme with the same tag vk using the server public key (let $C_{\mathsf{TBE,S}}$ be the ciphertext). Remark that to encode vk to the tag space of the TBE scheme, we employ a target-collision resistant hash function $H_{\mathsf{TBE}} : \{0,1\}^* \to \mathbb{Z}_p$. Finally, a signature σ is computed on two TBE ciphertexts and R by using sigk. A SCF-PEKS/PKE ciphertext is $(C_{\mathsf{TBE,S}}, C_{\mathsf{TBE,R}}, \mathsf{vk}, \sigma)$. The Dec algorithm computes $R = H_R(\mathsf{vk})$ and decrypts $C_{\mathsf{TBE,R}}$ by using the receiver secret key if σ is a valid signature on $(C_{\mathsf{TBE,S}}, C_{\mathsf{TBE,R}}, R)$. The Test algorithm decrypts $C_{\mathsf{TBE,S}}$ using the server secret key, and decrypts the result by using the trapdoor. Let R' be the decryption result. The algorithm outputs 1 if $R' = H(\mathsf{vk})$ and σ is a valid signature on $(C_{\mathsf{TBE,S}}, C_{\mathsf{TBE,R}}, R')$, and 0 otherwise.

Next, we describe a SCF-PEKS/PKE scheme. In addition to employ the strongly robust Gentry IBE scheme, we employ the Ghadafi TBE scheme [34] which is a Type-3 pairings variant of the

Kiltz TBE scheme [38]. We also employ the discrete-log-based Wee OTS scheme. We call the instantiation Gentry-Ghadafi-Wee (GGW). For encrypting an IBE ciphertext by using a TBE scheme, usually KEM/DEM framework (KEM stands for key encapsulation mechanism and DEM stands for data encapsulation mechanism) is required since the plaintext space of the TBE scheme is different from the ciphertext space of the IBE scheme. In order to avoid employing the KEM/DEM framework, Emura and Rahman [26] introduced partitioned ciphertext structures where an IBE ciphertext can be split into two parts, $C_{\mathsf{IBE},1}$ and $C_{\mathsf{IBE},2}$ such that $C_{\mathsf{IBE},1}$ only includes an identity (i.e., $C_{\mathsf{IBE},2}$ is independent of the identity) and for any common plaintext M and distinct identities ID_0 and ID_1, $C_{\mathsf{IBE},2}$ can be commonly used for $(C_{\mathsf{IBE},1}, C_{\mathsf{IBE},2}) \leftarrow \mathsf{IBE.Enc}(\mathsf{params}, \mathsf{ID}_0, M; s)$ and $(C'_{\mathsf{IBE},1}, C_{\mathsf{IBE},2}) \leftarrow \mathsf{IBE.Enc}(\mathsf{params}, \mathsf{ID}_1, M; s)$ if the same randomness s is used. This structure is employed for computing the challenge ciphertext regardless of whether ID_0 or ID_1 is encrypted. See [26] for details. If $C_{\mathsf{IBE},1}$ belongs to the plaintext space of the TBE scheme (e.g., $C_{\mathsf{IBE},1} \in \mathbb{G}_1$), it is enough to encrypt $C_{\mathsf{IBE},1}$, and $C_{\mathsf{IBE},2}$ can be directly included to the SCF-PEKS/PKE ciphertext. Remark that $C_{\mathsf{IBE},2}$ needs to be a part of signed message. Unfortunately, now $C_{\mathsf{IBE},1} = (C_1, C_4)$ due to the CCA security, i.e., $C_1 = {g'_1}^s g_1^{-s\mathsf{ID}}$ is generated by ID and C_4 depends on C_1 via the hashed value β.[5] Thus, still we need to employ the KEM/DEM framework for encrypting (C_1, C_4). In the following construction, we denote $M \odot K$ meaning that the underlying DEM scheme encrypts a plaintext M using K.

Without loss of generality, all algorithms use the same Type-3 bilinear groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, p)$. Moreover, two TBE schemes also use the same target-collision resistant hash function $H_{\mathsf{TBE}} : \{0,1\}^* \to \mathbb{Z}_p$. For the Wee OTS scheme, let $\mathbb{G}$ be a group with prime order q and $g \in \mathbb{G}$ be a generator, and $H_{\mathsf{sig}} : \{0,1\}^* \to \mathbb{Z}_q$ be a collision resistant hash function.

GGW SCF-PEKS/PKE Scheme:

- $\mathsf{KeyGen_S}(1^\lambda)$: Let $\bar{g}_1 \in \mathbb{G}_1$ and $\bar{g}_2 \in \mathbb{G}_2$ be generators. Choose $\bar{h}, \bar{w}, \bar{z}, \bar{u}, \bar{v} \xleftarrow{\$} \mathbb{Z}_p$, compute $\bar{H}_1 = \bar{g}_1^{\bar{h}}$, $\bar{H}_2 = \bar{g}_2^{\bar{h}}$, $\bar{U}_1 = \bar{H}_1^{\bar{u}}$, $\bar{U}_2 = \bar{H}_2^{\bar{u}}$, $\bar{V}_1 = \bar{U}_1^{1/\bar{v}}$, $\bar{V}_2 = \bar{U}_2^{1/\bar{v}}$, $\bar{W}_1 = \bar{H}_1^{\bar{w}}$, $\bar{W}_2 = \bar{H}_2^{\bar{w}}$, $\bar{Z}_1 = \bar{V}_1^{\bar{z}}$, and $\bar{Z}_2 = \bar{V}_2^{\bar{z}}$. Return a server public key $pk_{\mathrm{S}} = (\bar{g}_1, \bar{g}_2, \bar{H}_1, \bar{H}_2, \bar{U}_1, \bar{U}_2, \bar{V}_1, \bar{V}_2, \bar{W}_1, \bar{W}_2, \bar{Z}_1, \bar{Z}_2, H_{\mathsf{TBE}})$ and a server secret key $sk_{\mathrm{S}} = (\bar{u}, \bar{v})$.

- $\mathsf{KeyGen_R}(1^\lambda)$: Let $g_1, \hat{g}_1 \in \mathbb{G}_1$, $g_2, \hat{g}_2 \in \mathbb{G}_2$, and $g \in \mathbb{G}$ be generators. Choose $K \xleftarrow{\$} \{0,1\}^\lambda$, $g', h' \xleftarrow{\$} \mathbb{G}_1$, $\alpha \xleftarrow{\$} \mathbb{Z}_p$, and $h_1, h_2, h_3 \xleftarrow{\$} \mathbb{G}_2$, and compute $g'_1 = g_1^\alpha$. Let $H : \{0,1\}^* \to \mathbb{Z}_p$ be a universal one-way hash function and $H_{\mathsf{AES}} : \mathbb{G}_T \to \{0,1\}^{256}$ be a collision resistant hash function. Choose $\hat{h}, \hat{w}, \hat{z}, \hat{u}, \hat{v} \xleftarrow{\$} \mathbb{Z}_p$, compute $\hat{H}_1 = \hat{g}_1^{\hat{h}}$, $\hat{H}_2 = \hat{g}_2^{\hat{h}}$, $\hat{U}_1 = \hat{H}_1^{\hat{u}}$, $\hat{U}_2 = \hat{H}_2^{\hat{u}}$, $\hat{V}_1 = \hat{U}_1^{1/\hat{v}}$, $\hat{V}_2 = \hat{U}_2^{1/\hat{v}}$, $\hat{W}_1 = \hat{H}_1^{\hat{w}}$, $\hat{W}_2 = \hat{H}_2^{\hat{w}}$, $\hat{Z}_1 = \hat{V}_1^{\hat{z}}$, and $\hat{Z}_2 = \hat{V}_2^{\hat{z}}$. Let $H_{\mathsf{sig}} : \{0,1\}^* \to \mathbb{Z}_q$ be a collision resistant hash function. Return a receiver public key $pk_{\mathrm{R}} = (K, g', h', g_1, g'_1, g_2, h_1, h_2, h_3, H, H_{\mathsf{AES}}, \hat{g}_1, \hat{g}_2, \hat{H}_1, \hat{H}_2, \hat{U}_1, \hat{U}_2, \hat{V}_1, \hat{V}_2, \hat{W}_1, \hat{W}_2, \hat{Z}_1, \hat{Z}_2, H_{\mathsf{TBE}}, g, H_{\mathsf{sig}})$ and a receiver secret key $sk_{\mathrm{R}} = (\alpha, \hat{u}, \hat{v})$.

- $\mathsf{Trapdoor}(pk_{\mathrm{R}}, sk_{\mathrm{R}}, \omega)$: Parse $pk_{\mathrm{R}} = (K, g', h', g_1, g'_1, g_2, h_1, h_2, h_3, H, \hat{g}_1, \hat{g}_2, \hat{H}_1, \hat{H}_2, \hat{U}_1, \hat{U}_2, \hat{V}_1, \hat{V}_2, \hat{W}_1, \hat{W}_2, \hat{Z}_1, \hat{Z}_2, H_{\mathsf{TBE}}, g, H_{\mathsf{sig}})$ and $sk_{\mathrm{R}} = (\alpha, \hat{u}, \hat{v})$. For a keyword $\omega \in \mathbb{Z}_p$, for $i = 1, 2, 3$ choose $r_{\omega,i} \xleftarrow{\$} \mathbb{Z}_p$, compute $h_{\omega,i} = (h_i g_2^{r_{\omega,i}})^{1/(\alpha-\omega)}$, set $t_\omega := (\omega, (r_{\omega,i}, h_{\omega,i})_{i=1}^3)$, and return t_ω.

- $\mathsf{Enc}(pk_{\mathrm{S}}, pk_{\mathrm{R}}, \omega, M)$: Parse $pk_{\mathrm{S}} = (\bar{g}_1, \bar{g}_2, \bar{H}_1, \bar{H}_2, \bar{U}_1, \bar{U}_2, \bar{V}_1, \bar{V}_2, \bar{W}_1, \bar{W}_2, \bar{Z}_1, \bar{Z}_2, H_{\mathsf{TBE}})$ and $pk_{\mathrm{R}} = (K, g', h', g_1, g'_1, g_2, h_1, h_2, h_3, H, \hat{g}_1, \hat{g}_2, \hat{H}_1, \hat{H}_2, \hat{U}_1, \hat{U}_2, \hat{V}_1, \hat{V}_2, \hat{W}_1, \hat{W}_2, \hat{Z}_1, \hat{Z}_2, H_{\mathsf{TBE}}, g, H_{\mathsf{sig}})$.

[5] We remark that a commitment com is also generated by ID. However, thanks to the perfect hiding property of the Pedersen commitment scheme, for any value $com \in \mathbb{G}_1$ and distinct $\mathsf{ID}_0, \mathsf{ID}_1 \in \mathbb{Z}_p$ there exists $dec_0, dec_1 \in \mathbb{Z}_p$ such that $com = {g'}^{\mathsf{ID}_0} {h'}^{dec_0} = {g'}^{\mathsf{ID}_1} {h'}^{dec_1}$. Thus, com is not required to be a part of $C_{\mathsf{IBE},1}$.

  - Generate OTS keys: Choose $s_0, s_1, x \xleftarrow{\$} \mathbb{Z}_q$ and compute $u_0 = g^{s_0}$, $u_1 = g^{s_1}$, and $c = g^x$. Set $\mathsf{vk} := (u_0, u_1, c)$.
  - Compute IBE ciphertext for R: Choose $dec \xleftarrow{\$} \mathbb{Z}_p$ and compute $com = g'^{\omega} h'^{dec}$. Choose $s \xleftarrow{\$} \mathbb{Z}_p$. Compute $R = H_R(\mathsf{vk})$, $C_1 = {g_1'}^{s} g_1^{-s\omega}$, $C_2 = e(g_1, g_2)^s$, $K_{\mathsf{AES}} = H_{\mathsf{AES}}(e(g_1, h_1)^{-s})$, $C_3 = \mathsf{AES.Enc}(K_{\mathsf{AES}}, R||K||dec)$, $\beta = H(C_1, C_2, C_3)$, and $C_4 = e(g_1, h_2)^s e(g_1, h_3)^{s\beta}$. Set $C_{\mathsf{IBE},1} := (C_1, C_4)$ and $C_{\mathsf{IBE},2} := (com, C_2, C_3)$.
  - Compute TBE ciphertext for $C_{\mathsf{IBE},1}$: Compute $t = H_{\mathsf{TBE}}(\mathsf{vk})$. Choose $\bar{r}_1, \bar{r}_2 \xleftarrow{\$} \mathbb{Z}_p$. Compute $\bar{C}_1 = \bar{H}_1^{\bar{r}_1}$, $\bar{C}_2 = \bar{V}_1^{\bar{r}_2}$, $\bar{C}_3 = C_{\mathsf{IBE},1} \odot \bar{U}_1^{\bar{r}_1 + \bar{r}_2}$, $\bar{C}_4 = (\bar{U}_1^t \bar{W}_1)^{\bar{r}_1}$, and $\bar{C}_5 = (\bar{U}_1^t \bar{Z}_1)^{\bar{r}_2}$. Set $C_{\mathsf{TBE,S}} := (\bar{C}_1, \bar{C}_2, \bar{C}_3, \bar{C}_4, \bar{C}_5)$.
  - Compute TBE ciphertext for M: Compute $t = H_{\mathsf{TBE}}(\mathsf{vk})$. Choose $\hat{r}_1, \hat{r}_2 \xleftarrow{\$} \mathbb{Z}_p$. $\hat{C}_1 = \hat{H}_1^{\hat{r}_1}$, $\hat{C}_2 = \hat{V}_1^{\hat{r}_2}$, $\hat{C}_3 = M \hat{U}_1^{\hat{r}_1 + \hat{r}_2}$, $\hat{C}_4 = (\hat{U}_1^t \hat{W}_1)^{\hat{r}_1}$, and $\hat{C}_5 = (\hat{U}_1^t \hat{Z}_1)^{\hat{r}_2}$. Set $C_{\mathsf{TBE,R}} := (\hat{C}_1, \hat{C}_2, \hat{C}_3, \hat{C}_4, \hat{C}_5)$.
  - Compute OTS for $(C_{\mathsf{IBE},2}, C_{\mathsf{TBE,S}}, C_{\mathsf{TBE,R}}, R)$: Choose $e' \xleftarrow{\$} \mathbb{Z}_q$, compute $w' = x + e's_0 + (H_{\mathsf{sig}}(C_{\mathsf{IBE},2}, C_{\mathsf{TBE,S}}, C_{\mathsf{TBE,R}}, R) + e')s_1$, and set $\sigma := (e', \omega')$.

  Return $\mathsf{CT} = (C_{\mathsf{IBE},2}, C_{\mathsf{TBE,S}}, C_{\mathsf{TBE,R}}, \mathsf{vk}, \sigma)$.

- $\mathsf{Dec}(pk_{\mathrm{R}}, sk_{\mathrm{R}}, \mathsf{CT})$: Parse $pk_{\mathrm{R}} = (K, g', h', g_1, g_1', g_2, h_1, h_2, h_3, H, \hat{g}_1, \hat{g}_2, \hat{H}_1, \hat{H}_2, \hat{U}_1, \hat{U}_2, \hat{V}_1, \hat{V}_2, \hat{W}_1, \hat{W}_2, \hat{Z}_1, \hat{Z}_2, H_{\mathsf{TBE}}, g, H_{\mathsf{sig}})$, $sk_{\mathrm{R}} = (\alpha, \hat{u}, \hat{v})$, and $\mathsf{CT} = (C_{\mathsf{IBE},2}, C_{\mathsf{TBE,S}}, C_{\mathsf{TBE,R}}, \mathsf{vk}, \sigma)$.
  - Verify OTS σ: Parse $\mathsf{vk} = (u_0, u_1, c)$ and $\sigma = (e', \omega')$. Compute $R = H_R(\mathsf{vk})$. If $g^{\omega'} \neq c \cdot u_0^{e'} \cdot u_1^{H_{\mathsf{sig}}(C_{\mathsf{IBE},2}, C_{\mathsf{TBE,S}}, C_{\mathsf{TBE,R}}, R) + e'}$ then output $\perp$.
  - Decrypt TBE ciphertext $C_{\mathsf{TBE,R}}$: Parse $C_{\mathsf{TBE,R}} = (\hat{C}_1, \hat{C}_2, \hat{C}_3, \hat{C}_4, \hat{C}_5)$. If $e(\hat{C}_1, \hat{U}_2^t \hat{W}_2) \neq e(\hat{C}_4, \hat{H}_2)$ or $e(\hat{C}_2, \hat{U}_2^t \hat{Z}_2) \neq e(\hat{C}_5, \hat{V}_2)$, then output $\perp$. Otherwise, return $M = \hat{C}_3 / \hat{C}_1^{\hat{u}} \hat{C}_2^{\hat{v}}$.
- $\mathsf{Test}(pk_{\mathrm{S}}, sk_{\mathrm{S}}, pk_{\mathrm{R}}, t_\omega, \mathsf{CT})$: Parse $pk_{\mathrm{S}} = (\bar{g}_1, \bar{g}_2, \bar{H}_1, \bar{H}_2, \bar{U}_1, \bar{U}_2, \bar{V}_1, \bar{V}_2, \bar{W}_1, \bar{W}_2, \bar{Z}_1, \bar{Z}_2, H_{\mathsf{TBE}})$, $sk_{\mathrm{S}} = (\bar{u}, \bar{v})$, $pk_{\mathrm{R}} = (K, g', h', g_1, g_1', g_2, h_1, h_2, h_3, H, \hat{g}_1, \hat{g}_2, \hat{H}_1, \hat{H}_2, \hat{U}_1, \hat{U}_2, \hat{V}_1, \hat{V}_2, \hat{W}_1, \hat{W}_2, \hat{Z}_1, \hat{Z}_2, H_{\mathsf{TBE}}, g, H_{\mathsf{sig}})$, $t_\omega = (\omega, (r_{\omega,i}, h_{\omega,i})_{i=1}^3)$, and $\mathsf{CT} = (C_{\mathsf{IBE},2}, C_{\mathsf{TBE,S}}, C_{\mathsf{TBE,R}}, \mathsf{vk}, \sigma)$.
  - Decrypt TBE ciphertext $C_{\mathsf{TBE,S}}$: Parse $C_{\mathsf{TBE,S}} = (\bar{C}_1, \bar{C}_2, \bar{C}_3, \bar{C}_4, \bar{C}_5)$. If $e(\bar{C}_1, \bar{U}_2^t \bar{W}_2) \neq e(\bar{C}_4, \bar{H}_2)$ or $e(\bar{C}_2, \bar{U}_2^t \bar{Z}_2) \neq e(\bar{C}_5, \bar{V}_2)$, then output 0. Otherwise, set $C_{\mathsf{IBE},1} = \bar{C}_3 \odot \bar{C}_1^{\bar{u}} \bar{C}_2^{\bar{v}}$.
  - Decrypt IBE ciphertext $(C_{\mathsf{IBE},1}, C_{\mathsf{IBE},2})$: Parse $C_{\mathsf{IBE},1} = (C_1, C_4)$, $C_{\mathsf{IBE},2} = (com, C_2, C_3)$, and $t_\omega = (\omega, (r_{\omega,i}, h_{\omega,i})_{i=1}^3)$. Compute $\beta = H(C_1, C_2, C_3)$. If $C_4 \neq e(C_1, h_{\omega,2} h_{\omega,3}^{\beta}) C_2^{r_{\omega,2} + r_{\omega,3}\beta}$, then output 0. Otherwise, compute $R'||K'||dec = \mathsf{AES.Dec}(H_{\mathsf{AES}}((e(C_1, h_{\omega,1}) C_2^{r_{\omega,1}})^{-1}), C_3)$. Output 0 if $K' \neq K$. Otherwise, output $\perp$ if $com \neq g'^{\omega} h'^{dec}$. Otherwise, output 0 if $R' \neq H_R(\mathsf{vk})$.
  - Verify OTS σ: Parse $\mathsf{vk} = (u_0, u_1, c)$ and $\sigma = (e', \omega')$. If $g^{\omega'} \neq c \cdot u_0^{e'} \cdot u_1^{H_{\mathsf{sig}}(C_{\mathsf{IBE},2}, C_{\mathsf{TBE,S}}, C_{\mathsf{TBE,R}}, R') + e'}$ then output 0. Otherwise, output 1.

Acknowledgment

The authors would like to thank Dr. Miyako Ohkubo for her helpful advice to employ the Abe et al. transformation.

References

[1] The PBC (pairing-based cryptography) library. Available at `http://crypto.stanford.edu/pbc/`.

[2] Emad Abd-Elrahman, Hatem Ibn-Khedher, Hossam Afifi, and Thouraya Toukabri. Fast group discovery and non-repudiation in D2D communications using IBE. In *IWCMC*, pages 616–621, 2015.

[3] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. *J. Cryptology*, 21(3):350–391, 2008.

[4] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust encryption. In *TCC*, pages 480–497, 2010.

[5] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust encryption. *J. Cryptology*, 31(2):307–350, 2018.

[6] Masayuki Abe, Jens Groth, Miyako Ohkubo, and Takeya Tango. Converting cryptographic schemes from symmetric to asymmetric bilinear groups. In *CRYPTO*, pages 241–260, 2014.

[7] Masayuki Abe, Fumitaka Hoshino, and Miyako Ohkubo. Design in type-i, run in type-iii: Fast and scalable bilinear-type conversion using integer programming. In *CRYPTO*, pages 387–415, 2016.

[8] Bayu Anggorojati and Ramjee Prasad. Securing communication in inter domains Internet of Things using identity-based cryptography. In *IWBIS*, pages 137–142, 2017.

[9] Kazumaro Aoki. Towards reducing the gap between cryptography and its usage. *IEICE Transactions*, 102-A(1):11–16, 2019.

[10] Joonsang Baek, Reihaneh Safavi-Naini, and Willy Susilo. On the integration of public key data encryption and public key encryption with keyword search. In *ISC*, pages 217–232, 2006.

[11] Joonsang Baek, Reihaneh Safavi-Naini, and Willy Susilo. Public key encryption with keyword search revisited. In *ICCSA*, pages 1249–1259, 2008.

[12] Razvan Barbulescu and Sylvain Duquesne. Updating key size estimations for pairings. *J. Cryptology*, 2018.

[13] Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Constructing elliptic curves with prescribed embedding degrees. In *Security and Cryptography for Networks*, pages 257–267, 2002.

[14] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In *SAC*, pages 319–331, 2005.

[15] Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity-based encryption without random oracles. In *EUROCRYPT*, pages 223–238, 2004.

[16] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In *EUROCRYPT*, pages 506–522, 2004.

[17] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, pages 213–229, 2001.

[18] Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In *CRYPTO*, pages 290–307, 2006.

[19] Francesco Buccafurri, Gianluca Lax, Rajeev Anand Sahu, and Vishal Saraswat. Practical and secure integrated PKE+PEKS with keyword privacy. In *SECRYPT*, pages 448–453, 2015.

[20] Yu Chen, Jiang Zhang, Dongdai Lin, and Zhenfeng Zhang. Generic constructions of integrated PKE and PEKS. *Des. Codes Cryptography*, 78(2):493–526, 2016.

[21] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *EUROCRYPT*, pages 103–118, 1997.

[22] Angelo De Caro and Vincenzo Iovino. jPBC: Java pairing based cryptography. In *ISCC*, pages 850–855. IEEE, 2011.

[23] Keita Emura. A generic construction of secure-channel free searchable encryption with multiple keywords. In *NSS*, pages 3–18, 2017.

[24] Keita Emura, Atsuko Miyaji, and Kazumasa Omote. Adaptive secure-channel free public-key encryption with keyword search implies timed release encryption. In *ISC*, pages 102–118, 2011.

[25] Keita Emura, Atsuko Miyaji, Mohammad Shahriar Rahman, and Kazumasa Omote. Generic constructions of secure-channel free searchable encryption with adaptive security. *Security and Communication Networks*, 8(8):1547–1560, 2015.

[26] Keita Emura and Mohammad Shahriar Rahman. Constructing secure-channel free searchable encryption from anonymous IBE with partitioned ciphertext structure. In *SECRYPT*, pages 84–93, 2012.

[27] Liming Fang, Willy Susilo, Chunpeng Ge, and Jiandong Wang. A secure channel free public key encryption with keyword search scheme without random oracle. In *CANS*, pages 248–258, 2009.

[28] Liming Fang, Willy Susilo, Chunpeng Ge, and Jiandong Wang. Public key encryption with keyword search secure against keyword guessing attacks without random oracle. *Inf. Sci.*, 238:221–241, 2013.

[29] Pooya Farshim, Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. Robust encryption, revisited. In *Public-Key Cryptography*, pages 352–368, 2013.

[30] Pooya Farshim, Claudio Orlandi, and Razvan Rosie. Security of symmetric primitives under incorrect usage of keys. *IACR Trans. Symmetric Cryptol.*, 2017(1):449–473, 2017.

[31] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

[32] Craig Gentry. Practical identity-based encryption without random oracles. In *EUROCRYPT*, pages 445–464, 2006.

[33] Rémi Géraud, David Naccache, and Razvan Rosie. Robust encryption, extended. In *CT-RSA*, pages 149–168, 2019.

[34] Essam Ghadafi. Efficient distributed tag-based encryption and its application to group signatures with efficient distributed traceability. In *LATINCRYPT*, pages 327–347, 2014.

[35] Lifeng Guo and Wei-Chuen Yau. Efficient secure-channel free public key encryption with keyword search for EMRs in cloud storage. *J. Medical Systems*, 39(2):11, 2015.

[36] Antoine Joux and Kim Nguyen. Separating decision Diffie-Hellman from computational Diffie-Hellman in cryptographic groups. *J. Cryptology*, 16(4):239–247, 2003.

[37] Dalia Khader. Public key encryption with keyword search based on k-resilient IBE. In *ICCSA*, pages 1086–1095, 2007.

[38] Eike Kiltz. Chosen-ciphertext security from tag-based encryption. In *Theory of Cryptography*, pages 581–600, 2006.

[39] Taechan Kim and Razvan Barbulescu. Extended tower number field sieve: A new complexity for the medium prime case. In *CRYPTO*, pages 543–571, 2016.

[40] Tobias Markmann, Thomas C. Schmidt, and Matthias Wählisch. Federated end-to-end authentication for the constrained Internet of Things using IBC and ECC. In *ACM SIGCOMM*, pages 603–604, 2015.

[41] Micro Focus. Voltage SecureMail On-Premise: How it Works, 2019.

[42] Shigeo Mitsunari. mcl: A generic and fast pairing-based cryptography library, 2019/Sep/30 v1.00.

[43] Atsuko Miyaji, Masaki Nakabayashi, and Shunzou Takano. New explicit conditions of elliptic curve traces for FR-reduction. *IEICE Transactions*, 84-A(5):1234–1243, 2001.

[44] Payman Mohassel. A closer look at anonymity and robustness in encryption schemes. In *ASIACRYPT*, pages 501–518, 2010.

[45] Hiroshi Okano, Keita Emura, Takuya Ishibashi, Toshihiro Ohigashi, and Tatsuya Suzuki. Implementation of a strongly robust identity-based encryption scheme over type-3 pairings. In *CANDAR*, 2019, to appear.

[46] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO*, pages 129–140, 1991.

[47] Hyun Sook Rhee, Jong Hwan Park, and Dong Hoon Lee. Generic construction of designated tester public-key encryption with keyword search. *Inf. Sci.*, 205:93–109, 2012.

[48] Hyun Sook Rhee, Jong Hwan Park, Willy Susilo, and Dong Hoon Lee. Trapdoor security in a searchable public-key encryption scheme with a designated tester. *Journal of Systems and Software*, 83(5):763–771, 2010.

[49] Shahidatul Sadiah and Toru Nakanishi. Implementation of revocable group signatures with compact revocation list using vector commitments. In *CANDAR*, pages 489–495, 2017.

[50] Sriram Sankaran. Lightweight security framework for IoTs using identity based cryptography. In *ICACCI*, pages 880–886, 2016.

[51] Vishal Saraswat and Rajeev Anand Sahu. Short integrated PKE+PEKS in standard model. In *SPACE*, pages 226–246, 2017.

[52] Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.

[53] Tatsuya Suzuki, Keita Emura, and Toshihiro Ohigashi. A generic construction of integrated secure-channel free PEKS and PKE. In *ISPEC*, pages 69–86, 2018.

[54] Tatsuya Suzuki, Keita Emura, and Toshihiro Ohigashi. A generic construction of integrated secure-channel free PEKS and PKE and its application to EMRs in cloud storage. *J. Medical Systems*, 43(5):128:1–128:15, 2019.

[55] Tingting Wang, Man Ho Au, and Wei Wu. An efficient secure channel free searchable encryption scheme with multiple keywords. In *NSS*, pages 251–265, 2016.

[56] Rui Zhang and Hideki Imai. Combining public key encryption with keyword search and public key encryption. *IEICE Transactions*, 92-D(5):888–896, 2009.

Appendix

In this appendix, we explain why our encodings given in [45] do not work well.

Consider that $K \in \{0,1\}^\lambda$, $dec \in \mathbb{Z}_p$, and the plaintext space of the Gentry IBE scheme is $\mathbb{G}_T$. If we simply encode K (or dec) as G^K (or G^{dec}) for a public value $G \in \mathbb{G}_T$ (e.g., $G := e(g_1, g_2)$), then the decryption algorithm needs to compute K (or dec) from G^K (or G^{dec}). This requires to solve the discrete logarithm problem over $\mathbb{G}_T$. If K (or dec) is relatively small, then we can employ the lifted ElGamal encryption approach [21]. However, these values are randomly selected from $\{0,1\}^\lambda$ (or $\mathbb{Z}_p$) and thus we cannot employ the approach.

We pay attention to the algebraic structure of $\mathbb{G}_T$ and the decryption procedure. For a BN curve, $\mathbb{G}_T$, that has a prime order p, is a subgroup of $\mathbb{F}^*_{r^{12}}$ where r is a prime and $p|r^{12}-1$. $\mathbb{F}_{r^{12}}$ is represented as $\mathbb{F}_{r^2}[X]/(X^6-\xi)$ where $\xi \in \mathbb{F}_{r^2}$ is neither a square nor a cube, and hence $X^6-\xi$ is irreducible over $\mathbb{F}_{r^2}[X]$. Thus, an element of $\mathbb{F}_{r^{12}}$ can be written as $[[X_1, X_2], [X_3, X_4], \ldots, [X_{11}, X_{12}]]$ where each X_i $(i = 1, 2, \ldots, 12)$ is an element of $\mathbb{F}_r$. So, we set the plaintext space is $\mathbb{F}_r$. Moreover, K is λ-bit number and for 128-bit security (in our implementation), $\lambda = 128$, and $r > p$, $K \in \{0,1\}^\lambda$ and $dec \in \mathbb{Z}_p$ also in $\mathbb{F}_r$. So, we encode $(M||K||dec)$ as $\bar{M} := [[M, 0], [K, 0], [dec, 0], [0, 0], \ldots, [0, 0]] \in \mathbb{F}_{r^{12}}$. Remark that, now $C_3 := \bar{M} \cdot e(g_1, h_1)^{-s} \in \mathbb{F}_{r^{12}}$ and more precisely $C_3 \in \mathbb{F}_{r^{12}} \setminus \mathbb{G}_T$ with high probability $1 - p/(r^{12}-1)$. However, the decryption procedure is unaffected. The decryption algorithm computes $E := e(C_1, h_{\mathsf{ID},1}) C_2^{r_{\mathsf{ID},1}}$ over $\mathbb{G}_T$, and computes $C_3 \cdot E$ over $\mathbb{F}_{r^{12}}$. This offers no group operation for $\hat{M}$. Thus, the decryption algorithm can uniquely recover M, K, and dec. For a symmetric bilinear group, $\mathbb{G}_T$ is a subgroup of $\mathbb{F}^*_{r^2}$ where r is a prime $r = 3 \pmod 4$, and the order p is a prime factor of $r+1$. An element of $\mathbb{G}_T$ can be written as $[X_1, X_2]$ where each X_i $(i = 1, 2)$ is an element of $\mathbb{F}_r$. We set the plaintext space is $\mathbb{Z}_p$, and $(M||K||dec)$ is encoded as $\bar{M} := [M', 0] \in \mathbb{F}_{r^2}$ where $M' := M \cdot p^2 + K \cdot p + dec$. Then, since we set r as a 1536-bit prime and p is a 256-bit prime for ensuring 128-bit security (i.e., $r > p^4$), $M' \in \mathbb{F}_r$. Thus, the decryption algorithm can uniquely recover M, K, and dec.

At first glance, our encodings work well. However, there is an attack since $C_3^p \neq 1$ holds. This leaks information of plaintext and thus our encodings do not work well. Thus, in the current version we employed the KEM/DEM framework.