New Constructions of Fuzzy Identity-Based Encryption*

Joonsang Baek† Willy Susilo‡ Jianying Zhou†

Abstract

In this paper we construct two new fuzzy identity-based encryption (IBE) schemes in the random oracle model. Not only do our schemes provide public parameters whose size is *independent* of the number of attributes in each identity (used as public key) but they also have useful structures which result in more efficient key extraction and/or encryption than the random oracle version of Sahai and Water's fuzzy IBE scheme, considered recently by Pirretti et al. We prove that the confidentiality of the proposed schemes is relative to the Bilinear Decisional Bilinear Diffie-Hellman problem.

1 Introduction

Motivation. The concept of fuzzy identity-based encryption (IBE) recently introduced by Sahai and Waters [11] is to provide an *error-tolerance property* for IBE. Namely, in fuzzy IBE, a user with the secret key for the identity ω can decrypt a ciphertext encrypted with the public key ω' if ω and ω' are within a certain distance of each other. We note that in contrast to the previous approaches [8, 4], the biometric measurement in fuzzy IBE, which is used as an identity, does not need to be kept secret [11]. However, it must be ensured that an attacker cannot convince the key issuing authority to believe that he owns a biometric identity that he does not possess. As noted in [11], fuzzy IBE can directly be applied to the situation where a user is traveling and another party wants to encrypt at an ad-hoc meeting between them. Another application of fuzzy IBE is "attribute-based encryption [11, 6, 9]" where a party can encrypt data to all users that have a certain set of attributes, e.g. {`company`, `division`, `department`}.

Related Work. Since Sahai and Water's work, fuzzy IBE has been discussed in the context of the attribute-based encryption (ABE). Very recently, Goyal et al. [6] proposed an ABE scheme that provides fine-grained sharing of encrypted data. Piretti et al. [9] used Sahai and Waters' "large universe" construction of fuzzy IBE, which we simply call "Sahai-Waters construction", to realize their secure information management architecture. They also observed that if the random oracle [1] is employed, computational overhead of the Sahai-Waters construction can greatly be reduced. We remark that the random oracle not only reduces computational overhead but also provides a very short public parameters whose size is *independent* of the number of attributes associated with an identity or the number of attributes in the defined universe, which is crucial in the storage constrained applications.

Our Contribution. In this paper, we go one step beyond Pirreti et al.'s results by presenting fuzzy IBE schemes in the random oracle model, which are structurally different from the Sahai-Waters construction. We show that the structural difference results in more efficient schemes than even the random oracle version of the Sahai-Waters construction considered by Pirretti et al. [9]. We prove that our schemes meet the security requirements as defined in [11] assuming that the Decisional Bilinear Diffie-Hellman (DBDH) problem is hard.

*A short version of this paper is accepted to present at ASIACCS 2007. This is a full version.

†Institute for Infocomm Research, Singapore

‡University of Wollongong, Australia

2 Preliminaries

Computational Primitives. We first review the definition of the admissible bilinear pairing [2, 7], denoted by e. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be groups of the same order q which is prime. (By $\mathbb{G}_1^*$ and $\mathbb{Z}_q^*$, we denote $\mathbb{G}_1 \setminus \{1\}$ where 1 is the identity element of $\mathbb{G}_1$, and $\mathbb{Z}_q \setminus \{0\}$ respectively). Suppose that $\mathbb{G}_1$ is generated by g. Then, $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ has the following properties: 1) Bilinear: $e(g^a, g^b) = e(g, g)^{ab}$, for all $a, b \in \mathbb{Z}_q$ and 2) Non-degenerate: $e(g, g) \neq 1$.

A computational problem that will be used throughout this paper is the DBDH problem, a decisional version of the Bilinear Diffie-Hellman problem on which Boneh and Franklin's IBE scheme [2] is based. Informally, the DBDH problem refers to the problem where, given (g, g^a, g^b, g^c) for random $a, b, c \in \mathbb{Z}_q^*$, a polynomial-time attacker $\mathcal{A}$ is to distinguish $e(g, g)^{abc}$ from $e(g, g)^{\gamma}$ for random $\gamma \in \mathbb{Z}_q^*$.

Fuzzy IBE and Its Security. The generic fuzzy IBE scheme [11] consists of the following algorithms.

- $\mathsf{Setup}()$: Providing some security parameter as input, the Private Key Generator (PKG) runs this algorithm to generate its master key mk and public parameters $params$ which contains an error tolerance parameter d. Note that $params$ is given to all interested parties while mk is kept secret.

- $\mathsf{Extract}(mk, \mathtt{ID})$: Providing the master key mk and an identity $\mathtt{ID}$ as input, the PKG runs this algorithm to generate a private key associated with $\mathtt{ID}$, denoted by $D_{\mathtt{ID}}$.

- $\mathsf{Encrypt}(params, \mathtt{ID}', M)$: Providing the public parameters $params$, an identity $\mathtt{ID}'$, and a plaintext M as input, a sender runs this algorithm to generate a ciphertext C'.

- $\mathsf{Decrypt}(params, D_{\mathtt{ID}}, C')$: Providing the public parameters $params$, a private key $D_{\mathtt{ID}}$ associated with the identity $\mathtt{ID}$ and a ciphertext C' encrypted with an identity $\mathtt{ID}'$ such that $|\mathtt{ID}' \cap \mathtt{ID}| \geq d$ as input, a receiver runs this algorithm to get a decryption, which is either a plaintext or a "*Reject*" message.

A first security requirement of fuzzy IBE is "indistinguishability of encryptions under fuzzy selective-ID, chosen plaintext attack (IND-FSID-CPA)" [11]. (Note that the "selective-ID attack" [3] refers to the attack in which an attacker commits ahead of time an identity that it intends to attack.) The formal definition based on the game between an attacker $\mathcal{A}$ and the "Challenger" is as follows.

In Phase 1, $\mathcal{A}$ outputs a challenge identity $\mathtt{ID}^*$. In Phase 2, the Challenger then runs the Setup algorithm to generate a master key mk and public parameters $params$. The Challenger gives $params$ to $\mathcal{A}$ while keeps mk secret from $\mathcal{A}$. In Phase 3, $\mathcal{A}$ issues private key extraction queries, each of which is denoted by $\mathtt{ID}$. A restriction here is that for all $\mathtt{ID}$, $|\mathtt{ID} \cap \mathtt{ID}^*| < d$. In Phase 4, $\mathcal{A}$ outputs equal-length messages M_0 and M_1. Upon receiving (M_0, M_1), the Challenger picks $\beta \in \{0, 1\}$ at random and creates a challenge ciphertext $C^* = \mathsf{Encrypt}(params, \mathtt{ID}^*, M_\beta)$. The Challenger returns C^* to $\mathcal{A}$. In Phase 5, $\mathcal{A}$ issues a number of private key extraction queries as in Phase 3. In Phase 6, $\mathcal{A}$ outputs its guess $\beta' \in \{0, 1\}$.

We define $\mathcal{A}$'s guessing advantage by $|\Pr[\beta' = \beta] - \frac{1}{2}|$.

Notice that a stronger notion "indistinguishability of encryptions under fuzzy selective-ID, chosen ciphertext attack (IND-FSID-CCA)" can also be defined by giving $\mathcal{A}$ an access to a decryption oracle.

Another important security requirement for a fuzzy IBE scheme is the security against colluding attack, which implies that no group of users should be able to combine their keys in such a way that they can decrypt a ciphertext that none of them alone could [11].

3 Proposed Fuzzy IBE Schemes

In the rest of the paper, $\Delta_{a,S}$ denotes the Lagrange coefficient for $a \in \mathbb{Z}_q^*$ (q, a prime) and a set S of elements in $\mathbb{Z}_q^*$. Notice that

$$\Delta_{a,S}(x) = \prod_{a \in S, b \neq a} \frac{x-b}{a-b}.$$

Without loss of generality, we assume that an identity is a *set* of n *different* elements in $\mathbb{Z}_q^*$. For example, each of n strings of arbitrary length with an index $i \in \mathbb{Z}$ can be hashed using some collision-resistant hash function whose range is $\mathbb{Z}_q^*$.

Efficient Fuzzy IBE-I (EFIBE-I) Scheme. As mentioned earlier the hash function H in our first fuzzy IBE scheme is assumed to be a random oracle, which gives rise to very short public parameters. However, we note that our scheme has a different structure compared to the random oracle version of the Sahai-Waters construction considered in [9]. The unique feature of EFIBE-I is that its private key extraction algorithm (Extract) is structurally simple and highly efficient.

- Setup(): Generate a group $\mathbb{G}_1$ of prime order q. Construct a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, where $\mathbb{G}_2$ is a group of the same order q. Pick a generator g of the group $\mathbb{G}_1$. Pick $g_1 \in \mathbb{G}_1$ at random. Pick $s \in \mathbb{Z}_q^*$ at random and compute $g_2 = g^s$. Choose a hash function $H : \mathbb{Z}_q^* \rightarrow \mathbb{G}_1$. Select a tolerance parameter d. Output a public parameter $params = (q, g, e, \mathbb{G}_1, \mathbb{G}_2, H, g_1, g_2, d)$ and a master key $mk = (q, g, e, \mathbb{G}_1, \mathbb{G}_2, H, g_1, g_2, s)$.

- Extract(mk, ID), where $\mathtt{ID} = (\mu_1, \ldots, \mu_n)$: Pick a random polynomial $p(\cdot)$ of degree $d-1$ over $\mathbb{Z}_q$ such that $p(0) = s$ and compute a private key $D_{\mu_i} = (\gamma_{\mu_i}, \delta_{\mu_i}) = (H(\mu_i)^{p(\mu_i)}, g^{p(\mu_i)})$ for $i = 1, \ldots, n$. Return $D_{\mathtt{ID}} = (D_{\mu_1}, \ldots, D_{\mu_n})$.

- Encrypt($params$, ID$'$, M), where $\mathtt{ID}' = (\mu'_1, \ldots, \mu'_n)$ and $M \in \mathbb{G}_2$: Pick $r \in \mathbb{Z}_q^*$ at random and compute

$$\begin{aligned} C' &= (\mathtt{ID}', U, V_{\mu'_1}, \ldots, V_{\mu'_n}, W) \\ &= (\mathtt{ID}', g^r, (g_1 H(\mu'_1))^r, \ldots, (g_1 H(\mu'_n))^r, e(g_1, g_2)^r M) \end{aligned}$$

- Decrypt($params$, $D_{\mathtt{ID}}$, C'), where C' is encrypted with ID$'$ such that $|\mathtt{ID}' \cap \mathtt{ID}| \geq d$ (Recall that $\mathtt{ID} = (\mu_1, \ldots, \mu_n)$).: Choose an arbitrary set $S \subseteq \mathtt{ID} \cap \mathtt{ID}'$ such that $|S| = d$ and compute

$$M = \frac{e(\prod_{\mu_j \in S} \gamma_{\mu_j}^{\Delta_{\mu_j,S}(0)}, U)}{\prod_{\mu_j \in S} e(V_{\mu_j}, \delta_{\mu_j}^{\Delta_{\mu_j,S}(0)})} \cdot W$$

(Here, notice that $\mu'_j = \mu_j$ if $\mu_j \in S$). Return M.

The above decryption algorithm is correct as

$$\begin{aligned}
\frac{e(\prod_{\mu_j\in S}\gamma_{\mu_j}^{\Delta_{\mu_j,S(0)}},U)}{\prod_{\mu_j\in S}e(V_{\mu_j},\delta_{\mu_j}^{\Delta_{\mu_j,S(0)}})}\cdot W &= \frac{e(\prod_{\mu_j\in S}\gamma_{\mu_j}^{\Delta_{\mu_j,S(0)}},g^r)}{\prod_{\mu_j\in S}e((g_1H(\mu_j))^r,\delta_{\mu_j}^{\Delta_{\mu_j,S(0)}})}\cdot W \\
&= \frac{e(\prod_{\mu_j\in S}\gamma_{\mu_j}^{\Delta_{\mu_j,S(0)}},g^r)}{\prod_{\mu_j\in S}e((g_1^{\Delta_{\mu_j,S(0)}}H(\mu_j)^{\Delta_{\mu_j,S(0)}})^r,g^{p(\mu_j)})}\cdot W \\
&= \frac{e(\prod_{\mu_j\in S}\gamma_{\mu_j}^{\Delta_{\mu_j,S(0)}},g^r)}{\prod_{\mu_j\in S}e(g_1^{\Delta_{\mu_j,S(0)p(\mu_j)}}H(\mu_j)^{\Delta_{\mu_j,S(0)p(\mu_j)}},g^r)}\cdot W \\
&= \frac{e(\prod_{\mu_j\in S}\gamma_{\mu_j}^{\Delta_{\mu_j,S(0)}},g^r)}{e(\prod_{\mu_j\in S}g_1^{\Delta_{\mu_j,S(0)p(\mu_j)}},g^r)e(\prod_{\mu_j\in S}\gamma_{\mu_j}^{\Delta_{\mu_j,S(0)}},g^r)}\cdot W \\
&= \frac{1}{e(g_1^s,g^r)}\cdot e(g_1,g_2)^rM = \frac{1}{e(g_1,g_2)^r}\cdot e(g_1,g_2)^rM = M.
\end{aligned}$$

We now prove the following theorem regarding the security of EFIBE-I in the IND-FSID-CPA sense.

Theorem 1 *The EFIBE-I scheme is IND-FSID-CPA secure in the random oracle model assuming that the DBDH problem is hard.*

Proof. Assume that an attacker $\mathcal{A}$ breaks IND-FSID-CPA of EFIBE-I with probability greater than ϵ within time t making q_H random oracle queries and q_{ex} private key extraction queries. We show that using $\mathcal{A}$, one can construct a DBDH attacker $\mathcal{B}$.

Suppose that $\mathcal{B}$ is given $(q, g, e, \mathbb{G}_1, \mathbb{G}_2, g^a, g^b, g^c, \tau)$, where τ is either $e(g,g)^{abc}$ or $e(g,g)^{\gamma}$ for random $\gamma \in \mathbb{Z}_q^*$, as an instance of the DBDH problem. By ϵ' and t', we denote $\mathcal{B}$'s winning probability and running time respectively. $\mathcal{B}$ can simulate the Challenger's execution of each phase of IND-FSID-CPA game for $\mathcal{A}$ as follows.

Simulation of Phase 1. Suppose that $\mathcal{A}$ outputs a challenge identity $\mathtt{ID}^* = (\mu_1^*, \ldots, \mu_n^*)$.

Simulation of Phase 2. $\mathcal{B}$ sets $g_1 = g^b$ and $g_2 = g^c$, and gives $\mathcal{A}$ $(q, g, e, \mathbb{G}_1, \mathbb{G}_2, H, g_1, g_2, d)$ as *params*, where $d \in \mathbb{Z}^+$ and H is a random oracle controlled by $\mathcal{B}$ as follows.

Upon receiving a query μ to H:

If there exists $\langle \mu, (l, h)\rangle$ in HList, return h. Otherwise, do the following:

If $\mu = \mu_i^*$ for some $i \in [1, n]$, choose $l \in \mathbb{Z}_q^*$ at random and compute $h = g^l/g_1$.

Else choose $l \in \mathbb{Z}_q^*$ at random and compute $h = g^l$.

Add $\langle \mu, l, h\rangle$ to HList and return $H(\mu) = h$ as answer.

Simulation of Phase 3. $\mathcal{B}$ answers $\mathcal{A}$'s private key extraction queries as follows.

Upon receiving a private key extraction query $\mathtt{ID} = (\mu_1, \ldots, \mu_n)$ such that $|\mathtt{ID} \cap \mathtt{ID}^*| < d$:

Let $\Gamma = \mathtt{ID} \cap \mathtt{ID}^*$; Let Γ' be any set such that $\Gamma \subseteq \Gamma' \subseteq \mathtt{ID}$ and $|\Gamma'| = d - 1$; Let $S = \Gamma' \cup \{0\}$.

For every $\mu_i \in \Gamma'$, run the above H-oracle simulator to get $\langle \mu_i, l_i, h_i\rangle$ in HList, pick $\lambda_i \in \mathbb{Z}_q^*$ at random and compute $D_i = (h_i^{\lambda_i}, g^{\lambda_i})$.

For every $\mu_i \in \mathtt{ID} \setminus \Gamma'$, run the above H-oracle simulator to get $\langle \mu_i, l_i, h_i \rangle$ in HList and compute

$$D_i = \Big(\big(\prod_{\mu_j \in \Gamma'} h_i^{\Delta_{\mu_j, S(\mu_i)\lambda_j}} \big) g_2^{\Delta_{0,S(\mu_i) l_i}}, \big(\prod_{\mu_j \in \Gamma'} g^{\Delta_{\mu_j, S(\mu_i)\lambda_j}} \big) g_2^{\Delta_{0,S(\mu_i)}} \Big).$$

Return $(D_{\mu_1}, \ldots, D_{\mu_n})$.

Now define $\lambda_i = p(\mu_i)$ for a random polynomial $p(\cdot)$ of degree $d-1$ over $\mathbb{Z}_q^*$ such that $p(0) = c$. Notice that when $\mu_i \in \Gamma'$, the simulated D_i's and those of D_i's in the real attack are identically distributed. Notice also that even when $\mu_i \notin \Gamma'$, the above simulation is still correct. – Since $\mu_i \notin \Gamma'$ means $\mu_i \notin \Gamma$, $h_i = H(\mu_i) = g^{l_i}$ by the simulation of H. Thus, noting that $g_2 = g^c$, we have

$$\begin{aligned} D_i &= \Big(\big(g^{l_i (\sum_{\mu_j \in \Gamma'} \Delta_{\mu_j, S(\mu_i) p(\mu_j))} \big) g^{l_i \Delta_{0,S(\mu_i) c}}, g^{\sum_{\mu_j \in \Gamma'} \Delta_{\mu_j, S(\mu_i) p(\mu_j)}} g^{\Delta_{0,S(\mu_i) c}} \Big) \\ &= \Big(g^{l_i (\sum_{\mu_j \in \Gamma'} \Delta_{\mu_j, S(\mu_i) p(\mu_j) + \Delta_{0,S(\mu_i) p(0))}}, g^{\sum_{\mu_j \in \Gamma'} \Delta_{\mu_j, S(\mu_i) p(\mu_j) + \Delta_{0,S(\mu_i) p(0)}} \Big) \\ &= (g^{l_i p(\mu_i)}, g^{p(\mu_i)}) = (H(\mu_i)^{p(\mu_i)}, g^{p(\mu_i)}). \end{aligned}$$

Consequently, the simulated key $(D_{\mu_1}, \ldots, D_{\mu_n})$ is distributed the same as the one in the real attack.

Simulation of Phase 4. $\mathcal{B}$ creates a challenge ciphertext C^* as follows.

Upon receiving (M_0, M_1):

Choose $\beta \in \{0, 1\}$ at random.
Search HList to get $l_1^*, \ldots, l_n^*$ that correspond to each of $\mathtt{ID}^* = (\mu_1^*, \ldots, \mu_n^*)$.
Compute $g^{a l_i^*}$ for $i = 1, \ldots, n$.
Return $C^* = (g^a, g^{a l_1^*}, \ldots, g^{a l_n^*}, \tau M_\beta)$ as a challenge ciphertext.

Simulation of Phase 5. $\mathcal{B}$ answers $\mathcal{A}$'s random oracle/private key extraction queries as in Phase 3.
Simulation of Phase 6. $\mathcal{A}$ outputs its guess β'. If $\beta' = \beta$, $\mathcal{B}$ outputs 1. Otherwise, it outputs 0.

Analysis. Notice in the above simulation thta if $\tau = e(g,g)^{abc}$ then $\tau M_\beta = e(g^b, g^c)^a M_\beta = e(g_1, g_2)^a M_\beta$. Notice also that $g^{a l_i^*} = (g^{l_i^*})^a = (g_1 H(\mu_i^*))^a$ for $i = 1, \ldots, n$ from the construction of the random oracle H. Hence the challenge ciphertext C^* created above is distributed the same as the one in the real attack. On the other hand, if $\tau = e(g,g)^\gamma$ for $\gamma \in \mathbb{Z}_q^*$ chosen uniformly at random, τM_β is uniform in $\mathbb{G}_2$. As justified in the simulation of Phase 3, $\mathcal{B}$ perfectly simulates the random oracle H and the key private key extraction. Hence, we get $\Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^{abc}) = 1] = \Pr[\beta' = \beta]$, where $|\Pr[\beta' = \beta] - \frac{1}{2}| > \epsilon$, and $\Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^\gamma) = 1] = \Pr[\beta' = \beta] = \frac{1}{2}$, where γ is uniform in $\mathbb{G}_2$. Consequently, we get

$$|\Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^{abc}) = 1] - \Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^\gamma) = 1]| > \Big| \big(\frac{1}{2} \pm \epsilon \big) - \frac{1}{2} \Big| = \epsilon.$$

$\mathcal{B}$'s running time is computed as $t' < t + (q_H + q_{ex}) O(T_e)$, where T_e denotes the computing time for an exponentiation in $\mathbb{G}_1$. □

By the same argument as [11], the EFIBE-I scheme prevents collusion attacks since each users' private key components are generated with different random polynomials.– Even if multiple users collude, they will not be able to combine their private key components to form a key which is useful to compromise the confidentiality of the scheme.

Efficient Fuzzy IBE-II (EFIBE-II) Scheme. Our second fuzzy IBE scheme bears some similarities to the second scheme based on the DBDH problem [11]. However, its private key extraction has been simplified by using the outputs of the chosen random polynomial as random exponents for g_1, in contrast to the scheme in [11] which introduces extra random exponents and hence incurs extra exponentiations. More precisely, our scheme computes $((g_1 H(\mu_i))^{p(\mu_i)}, g^{p(\mu_i)})$ instead of $(g_1^{p(\mu_i)} H(\mu_i)^{r_i}, g^{r_i})$ [11] to generate a private key associated with an identity $\mathtt{ID}' = (\mu'_1, \ldots, \mu'_n)$.

A description of the scheme is as follows.

- $\mathsf{Setup}()$: Generate a group $\mathbb{G}_1$ of prime order q. Construct a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, where $\mathbb{G}_2$ is a group of the same order q. Pick a generator g of the group $\mathbb{G}_1$. Pick $g_1 \in \mathbb{G}_1$ at random. Pick $s \in \mathbb{Z}_q^*$ at random and compute $g_2 = g^s$. Choose a hash function $H : \mathbb{Z}_q^* \to \mathbb{G}_1$. Select a tolerance parameter d. Output a public parameter $params = (q, g, e, \mathbb{G}_1, \mathbb{G}_2, H, g_1, g_2, d)$ and a master key $mk = (q, g, e, \mathbb{G}_1, \mathbb{G}_2, H, g_1, g_2, s)$.
- $\mathsf{Extract}(mk, \mathtt{ID})$, where $\mathtt{ID} = (\mu_1, \ldots, \mu_n)$: Pick a random polynomial $p(\cdot)$ of degree $d-1$ over $\mathbb{Z}_q$ such that $p(0) = s$ and compute a private key $D_{\mu_i} = (\gamma_{\mu_i}, \delta_{\mu_i}) = ((g_1 H(\mu_i))^{p(\mu_i)}, g^{p(\mu_i)})$ for $i = 1, \ldots, n$. Return $D_{\mathtt{ID}} = (D_{\mu_1}, \ldots, D_{\mu_n})$.
- $\mathsf{Encrypt}(params, \mathtt{ID}', M)$, where $\mathtt{ID}' = (\mu'_1, \ldots, \mu'_n)$ and $M \in \mathbb{G}_2$: Pick $r \in \mathbb{Z}_q^*$ at random and compute

$$\begin{aligned} C' &= (\mathtt{ID}', U, V_{\mu'_1}, \ldots, V_{\mu'_n}, W) \\ &= (\mathtt{ID}', g^r, H(\mu'_1)^r, \ldots, H(\mu'_n)^r, e(g_1, g_2)^r M) \end{aligned}$$

- $\mathsf{Decrypt}(params, D_{\mathtt{ID}}, C')$, where C' is encrypted with $\mathtt{ID}'$ such that $|\mathtt{ID}' \cap \mathtt{ID}| \geq d$ (Recall that $\mathtt{ID} = (\mu_1, \ldots, \mu_n)$).: Choose an arbitrary set $S \subseteq \mathtt{ID} \cap \mathtt{ID}'$ such that $|S| = d$ and compute

$$M = \frac{\prod_{\mu_j \in S} e(V_{\mu_j}, \delta_{\mu_j}^{\Delta_{\mu_j, S}(0)})}{e(\prod_{\mu_j \in S} \gamma_{\mu_j}^{\Delta_{\mu_j, S}(0)}, U)} \cdot W$$

  (Here, notice that $\mu'_j = \mu_j$ if $\mu_j \in S$). Return M.

The above decryption algorithm is correct as

$$\begin{aligned} \frac{\prod_{\mu_j \in S} e(V_{\mu_j}, \delta_{\mu_j}^{\Delta_{\mu_j, S}(0)})}{e(\prod_{\mu_j \in S} \gamma_{\mu_j}^{\Delta_{\mu_j, S}(0)}, U)} \cdot W &= \frac{\prod_{\mu_j \in S} e(H(\mu_j)^r, g^{p(\mu_j)\Delta_{\mu_j, S}(0)})}{e(\prod_{\mu_j \in S} (g_1 H(\mu_j))^{p(\mu_j)\Delta_{\mu_j, S}(0)}, g^r)} \cdot W \\ &= \frac{\prod_{\mu_j \in S} e(H(\mu_j)^{p(\mu_j)\Delta_{\mu_j, S}(0)}, g^r)}{e(\prod_{\mu_j \in S} (g_1 H(\mu_j))^{p(\mu_j)\Delta_{\mu_j, S}(0)}, g^r)} \cdot W \\ &= \frac{\prod_{\mu_j \in S} e(H(\mu_j)^{p(\mu_j)\Delta_{\mu_j, S}(0)}, g^r)}{e(\prod_{\mu_j \in S} g_1^{p(\mu_j)\Delta_{\mu_j, S}(0)}, g^r)} \cdot \frac{W}{e(\prod_{\mu_j \in S} H(\mu_j)^{p(\mu_j)\Delta_{\mu_j, S}(0)}, g^r)} \\ &= \frac{1}{e(\prod_{\mu_j \in S} g_1^{p(\mu_j)\Delta_{\mu_j, S}(0)}, g^r)} \cdot e(g_1, g_2)^r M \\ &= \frac{1}{e(g_1^s, g^r)} \cdot e(g_1, g_2)^r M = \frac{1}{e(g_1, g_2)^r} \cdot e(g_1, g_2)^r M = M. \end{aligned}$$

We then prove the following theorem regarding the security of EFIBE-II in the IND-FSID-CPA sense.

Theorem 2 *The EFIBE-II scheme is IND-FSID-CPA secure in the random oracle model assuming that the DBDH problem is hard.*

Proof. Assume that an attacker $\mathcal{A}$ breaks IND-FSID-CPA of EFIBE-II with probability greater than ϵ within time t making q_{ex} private key extraction queries. We show that using $\mathcal{A}$, one can construct a DBDH attacker $\mathcal{B}$.

Suppose that $\mathcal{B}$ is given $(q, e, \mathbb{G}_1, \mathbb{G}_2, g, g^a, g^b, g^c, \tau)$, where τ is either $e(g,g)^{abc}$ or $e(g,g)^{\gamma}$ for random $\gamma \in \mathbb{Z}_q^*$, as an instance of the DBDH problem. By ϵ' and t', we denote $\mathcal{B}$'s winning probability and running time respectively. $\mathcal{B}$ can simulate the Challenger's execution of each phase of IND-FSID-CPA game for $\mathcal{A}$ as follows.

Simulation of Phase 1. Suppose that $\mathcal{A}$ outputs a challenge identity $\mathtt{ID}^* = (\mu_1^*, \ldots, \mu_n^*)$.

Simulation of Phase 2. $\mathcal{B}$ sets $g_1 = g^b$ and $g_2 = g^c$, and gives $\mathcal{A}$ $(q, g, e, \mathbb{G}_1, \mathbb{G}_2, H, g_1, g_2, d)$ as *params*, where $d \in \mathbb{Z}^+$ and H is a random oracle controlled by $\mathcal{B}$ as follows.

Upon receiving a query μ to H:

If there exists $\langle(\mu, l), h\rangle$ in HList, return h. Otherwise, do the following:

If $\mu = \mu_i^*$ for some $i \in [1, n]$, choose $l \in \mathbb{Z}_q^*$ at random and compute $h = g^l$.
Else choose $l \in \mathbb{Z}_q^*$ at random and compute $h = g^l / g_1$.
Add $\langle \mu, l, h\rangle$ to HList and return $h = H(\mu)$ as answer.

Simulation of Phase 3. $\mathcal{B}$ answers $\mathcal{A}$'s private key extraction queries as follows.

Upon receiving a private key extraction query $\mathtt{ID} = (\mu_1, \ldots, \mu_n)$ such that $|\mathtt{ID} \cap \mathtt{ID}^*| < d$:

Let $\Gamma = \mathtt{ID} \cap \mathtt{ID}^*$; Let Γ' be any set such that $\Gamma \subseteq \Gamma' \subseteq \mathtt{ID}$ and $|\Gamma'| = d - 1$; Let $S = \Gamma' \cup \{0\}$.

For every $\mu_i \in \Gamma'$, run the above H-oracle simulator to get $\langle \mu_i, l_i, h_i\rangle$ in HList, pick $\lambda_i \in \mathbb{Z}_q^*$ at random and compute $D_i = ((g_1 h_i)^{\lambda_i}, g^{\lambda_i})$. Let $\lambda_i = p(\mu_i)$.

For every $\mu_i \in \mathtt{ID} \setminus \Gamma'$, run the above H-oracle simulator to get $\langle \mu_i, l_i, h_i\rangle$ in HList and compute

$$D_i = \Big(\big(\prod_{\mu_j \in \Gamma'} (g_1 h_i)^{\Delta_{\mu_j, S}(\mu_i)\lambda_j} \big) g_2^{\Delta_{0,S}(\mu_i) l_i}, \big(\prod_{\mu_j \in \Gamma'} g^{\Delta_{\mu_j, S}(\mu_i)\lambda_j} \big) g_2^{\Delta_{0,S}(\mu_i)} \Big).$$

Return $(D_{\mu_1}, \ldots, D_{\mu_n})$.

Now define $\lambda_i = p(\mu_i)$ for a random polynomial $p(\cdot)$ of degree $d-1$ over $\mathbb{Z}_q^*$ such that $p(0) = c$. Notice that when $\mu_i \in \Gamma'$, the simulated D_i's and those of D_i's in the real attack are identically distributed. Notice also that even when $\mu_i \notin \Gamma'$, the above simulation is still correct. – Since $\mu_i \notin \Gamma'$ means $\mu_i \notin \Gamma$, $g_1 h_i = g^{l_i}$. Noting that $g_2 = g^c$, we have

$$\begin{aligned}
D_i &= \Big(\big(g^{l_i (\sum_{\mu_j \in \Gamma'} \Delta_{\mu_j, S}(\mu_i) p(\mu_j))} \big) g^{l_i \Delta_{0,S}(\mu_i) c}, g^{\sum_{\mu_j \in \Gamma'} \Delta_{\mu_j, S}(\mu_i) p(\mu_j)} g^{\Delta_{0,S}(\mu_i) c} \Big) \\
&= (g^{l_i (\sum_{\mu_j \in \Gamma'} \Delta_{\mu_j, S}(\mu_i) p(\mu_j) + \Delta_{0,S}(\mu_i) p(0))}, g^{\sum_{\mu_j \in \Gamma'} \Delta_{\mu_j, S}(\mu_i) p(\mu_j) + \Delta_{0,S}(\mu_i) p(0)}) \\
&= (g^{l_i p(\mu_i)}, g^{p(\mu_i)}) = ((g_1 h_i)^{p(\mu_i)}, g^{p(\mu_i)}) \\
&= ((g_1 H(\mu_i))^{p(\mu_i)}, g^{p(\mu_i)}).
\end{aligned}$$

Consequently the simulated key $(D_{\mu_1}, \ldots, D_{\mu_n})$ is distributed the same as the one in the real attack.

Simulation of Phase 4. $\mathcal{B}$ creates a challenge ciphertext C^* as follows.

Upon receiving (M_0, M_1):

Choose $\beta \in \{0,1\}$ at random.

Search HList to get $l_1^*, \ldots, l_n^*$ that correspond to each of $\mathtt{ID}^* = (\mu_1^*, \ldots, \mu_n^*)$.

Compute $g^{al_i^*}$ for $i = 1, \ldots, n$.

Return $C^* = (g^a, g^{al_1^*}, \ldots, g^{al_n^*}, \tau M_\beta)$ as a challenge ciphertext.

Simulation of Phase 5. $\mathcal{B}$ answers $\mathcal{A}$'s random oracle/private key extraction queries as in Phase 3.

Simulation of Phase 6. $\mathcal{A}$ outputs its guess β'. If $\beta' = \beta$, $\mathcal{B}$ outputs 1. Otherwise, it outputs 0.

Analysis. Note that if $\tau = e(g,g)^{abc}$, $\tau M_\beta = e(g^b, g^c)^a M_\beta = e(g_1, g_2)^a M_\beta$. Note also that $g^{al_i^*} = (g^{l_i^*})^a = H(\mu_i^*)^a$ for $i = 1, \ldots, n$ from the construction of the random oracle H. Hence the challenge ciphertext C^* created above is distributed the same as the one in the real attack. On the other hand, if τ is uniform and independent in $\mathbb{G}_2$, i.e. $\tau = e(g,g)^\gamma$ for some $\gamma \in \mathbb{Z}_q^*$ uniformly chosen at random, so is τM_β. As justified in the simulation of Phase 3, $\mathcal{B}$ perfectly simulates the random oracle H and the key private key extraction. Hence, we get $\Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^{abc}) = 1] = \Pr[\beta' = \beta]$, where $|\Pr[\beta' = \beta] - \frac{1}{2}| > \epsilon$, and $\Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^\gamma) = 1] = \Pr[\beta' = \beta] = \frac{1}{2}$, where γ is uniform in $\mathbb{G}_2$. Consequently, we get

$$|\Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^{abc}) = 1] - \Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^\gamma) = 1]| > \left|\left(\frac{1}{2} \pm \epsilon\right) - \frac{1}{2}\right| = \epsilon.$$

$\mathcal{B}$'s running time is calculated as $t' < t + q_H O(T_e)$, where T_e denotes the computing time for an exponentiation in $\mathbb{G}_1$. □

Finally we note that from the same reason as EFIBE-I, EFIBE-II is also secure against collusion attacks.

Finally we remark that EFIBE-I and EFIBE-II can be extended to achieve chosen ciphertext security, *i.e.* IND-FSID-CCA, using the Fujisaki-Okamoto transform [5] in the random oracle model or the simulation-sound NIZK proofs [10] without depending on the random oracle model, as discussed in [11].

4 Comparisons

Table 1 summarizes the size of various parameters and the cost of computing sub-algorithms of the proposed fuzzy IBE schemes and the random oracle version of the Sahai-Waters construction [9], which we denote by SW-RO.

Notice that both $\mathsf{Extract}$ and $\mathsf{Encrypt}$ algorithms of EFIBE-II are more efficient than those of SW-RO. The $\mathsf{Extract}$ algorithm of EFIBE-I is the most efficient among the three schemes but its $\mathsf{Encrypt}$ is slightly less efficient than those of EFIBE-II and SW-RO.

5 Concluding Remarks

We expect that our new fuzzy IBE schemes will serve as efficient building blocks for biometric authentication systems or attribute-based encryption systems.

Construction of fuzzy IBE schemes that have the exactly the same structures as ours (that is, non-random oracle version of our schemes using the technique of [11]) is an interesting open problem.

	EFIBE-I	EFIBE-II	SW-RO
Size of $params \backslash \{g, g, e, \mathbb{G}_1, \mathbb{G}_2, d\}$	$2\|\mathbb{G}_1\|$	$2\|\mathbb{G}_1\|$	$2\|\mathbb{G}_1\|$
Size of $D_{\mathtt{ID}}$	$2n\|\mathbb{G}_1\|$	$2n\|\mathbb{G}_1\|$	$2n\|\mathbb{G}_1\|$
Size of $C \setminus \mathtt{ID}$	$(n+1)\|\mathbb{G}_1\| + \|\mathbb{G}_2\|$	$(n+1)\|\mathbb{G}_1\| + \|\mathbb{G}_2\|$	$(n+1)\|\mathbb{G}_1\| + \|\mathbb{G}_2\|$
Cost of Extract	$n(T_H + 2T_e)$	$n(T_H + T_m + 2T_e)$	$n(T_H + T_m + 3T_e)$
Cost of Encrypt	$n(T_m + T_e + T_H) + 2T_e + T_p + T'_m$	$n(T_e + T_H) + 2T_e + T_p + T'_m$	$n(T_e + T_H) + 2T_e + T_p + T'_m$
Cost of Decrypt	$d(T_e + T_m) + d(T_e + T_p) + T_p + T'_i + T'_m$	$d(T_e + T_m) + d(T_e + T_p) + T_p + T'_i + T'_m$	$d(T_e + T_m) + d(T_e + T_p) + T_p + T'_i + T'_m$
Security Rel. to	DBDH	DBDH	DBDH

Table 1: Comparisons of Various Fuzzy IBE Schemes. Abbreviations: $|S|$ – the bit-length of an element in set (or group) S; n – the number of elements in an identity; T_e – the computation time for a single exponentiation in $\mathbb{G}_1$; T_H – the computation time for a function H modeled as a random oracle; T_m – the computation time for a single multiplication in $\mathbb{G}_1$; T_i – the computation time for a single inverse operation in $\mathbb{G}_1$; T_p – the computation a single fora single pairing operation; T'_m – the computation time for a single multiplication in $\mathbb{G}_2$; T'_i – the computation time for a single inverse operation in $\mathbb{G}_2$; d – an error tolerance parameter

References

[1] M. Bellare and P. Rogaway, *Random Oracles are Practical: A Paradigm for Designing Efficient Protocols*, In ACM CCS '93, pp. 62–73, ACM Press, 1993.

[2] D. Boneh and M. Franklin, *Identity-Based Encryption from the Weil Pairing*, In Crypto '01, LNCS 2139, pp. 213–229, Springer-Verlag, 2001.

[3] R. Canetti, S. Halevi, and J. Katz, *A Forward-Secure Public-Key Encryption Scheme*, Advances in Cryptology - In Eurocrypt 2003, LNCS 2656, pp. 255–271, Springer-Verlag, 2003.

[4] Y. Dodis, L. Reyzin and A. Smith, *Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data*, In Eurocrypt '04, LNCS 3027, pp. 523 – 540, Springer-Verlag, 2004.

[5] E. Fujisaki and T. Okamoto, *Secure Integration of Asymmetric and Symmetric Encryption Schemes*, In Crypto '99, LNCS 1666, pp. 537 – 554, Springer-Verlag, 1999.

[6] V. Goyal, O. Pandey, A. Sahai and B. Waters, *Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data*, In ACM CCS '06, 2006, to appear.

[7] A. Joux: *The Weil and Tate Pairings as Building Blocks for Public Key Cryptosystems*, Algorithmic Number Theory Symposium (ANTS-V) '02, LNCS 2369, pp. 20–32, Springer-Verlag, 2002.

[8] A. Juels and M. Wattenberg, *A Fuzzy Commitment Scheme*, In ACM CCS '99, pp. 28–36, ACM Press, 1999.

[9] M. Pirretti, P. Traynor, P. McDaniel and B. Waters, *Secure Attribute-Based Systems*, In ACM CCS '06, 2006, to appear.

[10] A. Sahai, *Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security*, In FOCS '99, pp. 543–553, IEEE Computer Society.

[11] A. Sahai and B. Waters, *Fuzzy Identity-Based Encryption*, Advances in Cryptology - In Eurocrypt 2005, LNCS 3494, pp. 457–473, Springer-Verlag, 2005.