Identity-Based Encryption Resilient to Continual Auxiliary Leakage

Tsz Hon Yuen[1], Sherman S.M. Chow[2], Ye Zhang[3,⋆], and Siu Ming Yiu[1]

[1] University of Hong Kong, Hong Kong
{thyuen,smyiu}@cs.hku.hk
[2] University of Waterloo, Canada
smchow@math.uwaterloo.ca
[3] Pennsylvania State University, USA
yxz169@cse.psu.edu

Abstract. We devise the first identity-based encryption (IBE) that remains secure even when the adversary is equipped with *auxiliary input* (STOC '09) – any computationally uninvertible function of the master secret key and the identity-based secret key. In particular, this is more general than the tolerance of Chow *et al.*'s IBE schemes (CCS '10) and Lewko *et al.*'s IBE schemes (TCC '11), in which the leakage is bounded by a pre-defined number of bits; yet our construction is also fully secure in the standard model based on only static assumptions, and can be easily extended to give the first hierarchical IBE with auxiliary input.

Furthermore, we propose the model of *continual auxiliary leakage* (CAL) that can capture both memory leakage and continual leakage. The CAL model is particularly appealing since it not only gives a clean definition when there are multiple secret keys (the master secret key, the identity-based secret keys, and their refreshed versions), but also gives a generalized definition that does not assume secure erasure of secret keys after each key update. This is different from previous definitions of continual leakage (FOCS '10, TCC '11) in which the length-bounded leakage is only the secret key in the current time period. Finally, we devise an IBE scheme which is secure in this model. A major tool we use is the modified Goldreich-Levin theorem (TCC '10), which until now has only been applied in traditional public-key encryption with a single private key.

1 Introduction

In cryptography, security guarantees are usually proven under the assumption that the secret key must be kept safely and other internal (random) state is not leaked to the adversary. Even if a single bit of these secrets is leaked, the protection guaranteed by the proof is lost. In practice, however, it is difficult to avoid all possible kinds of leakage, such as side-channel attacks that exploit the physical nature of cryptographic operations (e.g., timing, power, radiation, cold

⋆ Part of this work was done while the third author was at University of Hong Kong.

D. Pointcheval and T. Johansson (Eds.): EUROCRYPT 2012, LNCS 7237, pp. 117–134, 2012.

boot attacks, etc.) or the reuse of the secret key and/or the randomness in a number of applications.

Leakage-resilient cryptography was introduced to provide formal security guarantees even when the leakage of the secret keying material is allowed. In this paper, we focus on public key encryption that is secure against memory leakage. More precisely, the adversary is allowed to specify an efficiently computable leakage function f and obtain the output of f applied to the secret key and other internal state. This function f aims to model the possible leakage that the adversary can learn in practice.

In recent years, we have seen a number of leakage models that impose different restriction on f. Recall the major (open) problem in leakage-resilient cryptography [10]:

> *"allowing for continual (overall unbounded) leakage, without additionally restricting its type."*

An ongoing line of research is on loosening the restriction of leakage types. The *relative leakage* model [1] restricts the function f to output at most l bits, where l is smaller than the secret key size. This restriction is later relaxed by allowing f to lower the entropy of the secret key by at most l bits [15]. To sum up these two models, l is defined as a fraction of the key (either in terms of the bit size or the entropy). The *bounded retrieval model* (e.g., see [2,8]), on the other hand, treats the leakage l as a system parameter. The size of the secret key can be increased to allow l bits of leakage, without affecting the public key size, communication and computation efficiency. To further relax the restriction, Dodis *et al.* [9] considered *auxiliary inputs*, which allow any f that no polynomial time adversary can invert (i.e., to output the secret key being leaked) with non-negligible probability. For example, any (computationally) one-way permutation can be used by the adversary as its auxiliary input, but is not allowed in the relative (leakage/entropy) model (as a permutation information-theoretically reveals the entire key). Therefore, auxiliary inputs allow us to consider a larger class of leakage functions.

The above line of research bounds the leakage *throughout the entire lifetime* of the secret key. Another paradigm considers a key update algorithm that continually refreshes the secret key, while bounding the leakage between updates. It addresses the first part of the aforementioned major problem. This model is known as the *continual leakage* model. There are signature, identification [10] and public key encryption schemes [6] secure in this model. Lewko *et al.* [12] recently proposed signature and encryption schemes that allow a constant fraction leakage of the secret key and the randomness during updates. In these papers, the leakage bound between updates is either based on the relative leakage or the bounded retrieval model. Therefore, the number of bits leaked between updates is still restricted.

IBE with Auxiliary Inputs. Dodis *et al.* [9] only considered public key encryption and there is no known identity-based encryption (IBE) scheme that is

secure with auxiliary input, even though there are a number of (bounded) leakage-resilient IBE schemes [1,2,8,6,13]. A distinctive feature of IBE is that any string can be used as a public key and potentially an exponential number of identities can be supported. This feature has many applications (e.g., see [2,3,5,8]), and furthermore makes the auxiliary input model appealing to IBE for various reasons. First, the auxiliary input model is useful in the context of composition. One may use the encryption public key (and corresponding secret key) in other applications (e.g., digital signatures and identification), and their composition remains secure as long as these other schemes were proved secure in the standard (no leakage) sense [9]. With the versatility of identity-based (ID-based) keys this guarantee appears to be more desirable. Second, the auxiliary input model not only tolerates a wider class of leakage, but also gives a "clean" definition of the necessary restriction on leakage. The model is free from numeric bounds (e.g., number of bits leaked from the master secret key, or number of bits leaked from the ID-based secret key of the target user) which are necessary in bounded retrieval model.

Continual Auxiliary Leakages. Recall that the key idea to achieve continual memory leakage (CML) resilience is to refresh the secret key in each time period. Previous CML models for IBE [6,13] only consider leakage of the current secret key for a given time. In other words, after a user has computed a new secret key for the next period, the old secret key should be *securely erased* from memory (so the leakage via the key-update query is the "last chance" for the adversary)[1]. This greatly diminishes the benefits offered by the formal leakage-resilience guarantee since with frequent secure erasures it is less disastrous to have memory leakage.

Combining the concept of auxiliary inputs with CML brings the possibility of new leakage-resilience guarantees. Our continual auxiliary leakage (CAL) model allows continual leakage, and the leakage between updates has minimal restriction: no polynomial time algorithm can use the leaked information to output a valid secret key. The CAL model still inherits the simplicity of the standard (non-continual) auxiliary input/leakage model. In particular, we do not need to keep track of the "version number" of keys.

Our Contributions. We tackle the problem of allowing continual (overall unbounded) leakage, without additionally restricting its type, for IBE. Brakerski *et al.*'s CML-resilient IBE [6] does not tolerate leakage from the master secret key and is only selective-secure. Lewko *et al.* [13] proposed a fully-secure CML-resilient IBE, but the leakage size during updates is limited to logarithmic. It is fair to say there is no complete solution for this major problem in leakage-resilient cryptography. To achieve our goal, we propose the continual auxiliary leakage (CAL) model and construct an IBE scheme secure in this strong model.

[1] We do not claim to have discovered any attack against the schemes in the respective papers exploiting this more general form of leakage. We are merely pointing out that the stronger attack is not covered by the current proofs.

We begin by constructing the *first* IBE scheme that is secure in the presence of *auxiliary inputs*. Our construction in §3.3 preserves the nice features of recent leakage-resilient IBE schemes [8,13]: adaptive security in the standard model and based on static assumptions, and moderate increases the size of the ciphertext and computational complexity.

Our work combines a number of techniques in the literature. We use the dual system encryption [14] for both adaptive security regarding the ID-based keys, and for the leakage-resilience via the proof technique [13] which allows the leakage of the master secret key and the ID-based secret keys. For leakage in the form of auxiliary input (or "auxiliary leakage"), we use the modified Goldreich-Levin theorem [9]. We overcome a number of technical difficulties when combining these techniques. Firstly, we cannot directly use the modified Goldreich-Levin theorem as it restricts the blinding factor of the semi-functional key (an artifact in the security reduction for allowing bounded leakage, which is created using a blinding factor from $\mathbb{Z}_{p_2}^m$ where $m \in \mathbb{N}$ and p_2 is a large prime of size 2^λ; see [13, Lemmata 6.2, 6.7]) to be a λ-bit number. Therefore, we need to construct the semi-functional key subject to this design constraint. It is also interesting to note that the λ-bit number is used as a real secret key of the public key encryption with auxiliary input [9], but in our case it just appears in the "imaginary" semi-functional secret key in the simulation. Secondly, Lewko *et al.*'s IBE [13] does not allow any leakage during setup. We twist their idea of using multiple tags (instead of a single tag in [8]) and do the "replication" in another level. (Thus we retain the same order of complexity for performance.) Although this technique by itself (see §3.3) does not allow leakage during setup, this structure leads us to construct an IBE scheme (in §4.4) which can be proven secure in the CAL model (i.e., leakage is allowed during setup). Furthermore, our scheme in §3.3 can be extended to give the *first* hierarchical IBE with auxiliary inputs.

In §4, we present the CAL model and propose the *first* IBE scheme secure in this strengthened model. There are a few problems that arise when we tried to borrow the construction technique from the (Diffie-Hellman-based) BHHO encryption in [9] to extend our IBE scheme in §3.3. Firstly, the modified Goldreich-Levin theorem [9] states that if the master secret keys α_i belongs to a subgroup H of $\mathbb{Z}_{p_1}$, then there exists an inverter with running time poly($|H|$). If $H = \mathbb{Z}_{p_1}$ and p_1 is a λ-bit prime, the running time of the inverter is poly(2^λ), which is undesirable. Secondly, if we simply change the scheme such that α_i belongs to a subgroup $H = \{0, 1\}$, then the master public key becomes $y_i = \hat{e}(g_1, v_i)^0$ or $\hat{e}(g_1, v_i)^1$. Then any adversary can determine α_i by brute-force. The same attack applies if $|H| \ll p_1$. We then try to construct the public key as in the BHHO encryption in [9], and the master public key becomes $y = \prod_{i=1}^n \hat{e}(g_1, v_i)^{\alpha_i}$. The master secret key msk becomes $v_i^{\alpha_i}$ for $i \in [1, n]$. (It seems one may set msk be $\prod_{i=1}^n v_i^{\alpha_i}$, but we want to split it into n pieces in order to apply the modified Goldreich-Levin theorem.) That means leakage will be in the form of $f(v_1^{\alpha_1}, \ldots, v_n^{\alpha_n})$. Intuitively, to simulate all possible uninvertible function f, the knowledge $v_i^{\alpha_i}$ is needed, which implies the knowledge of α_i since the brute-force attack is easy on α_i. This leads to a contradiction since the α_i's is the solution of the underlying intractability

problem. To resolve these issues, we change the structure of msk to $\prod_{j=1}^{m} v_{i,j}^{\alpha_j}$. We use a random $(n \times m)$ matrix $\boldsymbol{V} = \{v_{i,j}\}_{i\in[1,n],j\in[1,m]}$ and m random numbers $(\alpha_1, \ldots, \alpha_m)$ to obtain n master secret keys and public keys. Similar to (the semi-functional key in) our IBE in §3.3, this $\boldsymbol{V}$ is a conceptual building block for leakage-resilience and the knowledge of these group elements is not required anywhere else including key-update, encryption and decryption. This new msk does not reveal α_i under some intractability assumptions. Interestingly, these assumptions are required for a variant of Gentry-Peikert-Vaikuntanathan's (GPV) encryption scheme [11] based on learning with error (LWE) [9], which gives some evidence that our construction is not a trivial extension from the IBE in §3.3. Using lattice-based assumptions to help constructing pairing-based cryptosystems seems to be interesting on its own right. We leave it as an open problem to build a CAL-resilient IBE scheme without these assumptions.

2 Background

Composite Order Bilinear Groups [4]. Let $\mathcal{G}$ be a group generator, that takes a security parameter 1^λ as input where $\lambda \in \mathbb{N}$, outputs a description of bilinear group $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e})$, where p_1, p_2, p_3 are distinct λ-bit primes, $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order N, and $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear map such that $\forall g, h \in \mathbb{G}$ and $a, b \in \mathbb{Z}_N$, $\hat{e}(g^a, h^b) = \hat{e}(g, h)^{ab}$; $\hat{e}(g, g)$ generates $\mathbb{G}_T$ if g is a generator of $\mathbb{G}$. We denote $\mathbb{G}_{p_i}$ as the subgroup of order p_i in $\mathbb{G}$ $(i = 1, 2, 3)$. Let g_i be the generator of the subgroup $\mathbb{G}_{p_i}$. For all $h_i \in \mathbb{G}_{p_i}$ and $h_j \in \mathbb{G}_{p_j}$, if $i \neq j$, $\hat{e}(h_i, h_j) = 1$. We denote $\mathbb{G}_{p_1p_2}$ as the subgroup of order p_1p_2 in $\mathbb{G}$. For all $T \in \mathbb{G}_{p_1p_2}$, T can be written uniquely as the product of an element of $\mathbb{G}_{p_1}$ and an element of $\mathbb{G}_{p_2}$. We refer to these elements as the "$\mathbb{G}_{p_1}$ part of T" and the "$\mathbb{G}_{p_2}$ part of T" respectively. We also define $\mathbb{G}_{p_1p_3}$ and $\mathbb{G} = \mathbb{G}_{p_1p_2p_3}$ similarly.

Decisional Problems [14]. For a group generator $\mathcal{G}$, the following experiments define subgroup decision problem for $\mathbb{G}_{p_1}$ and $\mathbb{G}_{p_1p_2}$, subgroup decision problem for $\mathbb{G}_{p_1p_3}$ and $\mathbb{G}$, and subgroup decisional bilinear Diffie-Hellman problem.

Experiment $\mathbf{Exp}^{(1)}_{\mathcal{G},\mathcal{A}_1,\beta}(1^\lambda)$

$(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda), \quad g, X_1 \xleftarrow{R} \mathbb{G}_{p_1}, \quad X_2 \xleftarrow{R} \mathbb{G}_{p_2}, \quad X_3 \xleftarrow{R} \mathbb{G}_{p_3}$

$T_0 \xleftarrow{R} \mathbb{G}_{p_1p_2}, \quad T_1 \xleftarrow{R} \mathbb{G}_{p_1}.$

Return $\beta' \leftarrow \mathcal{A}_1(N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, X_1X_2, X_3, T_\beta)$.

Experiment $\mathbf{Exp}^{(2)}_{\mathcal{G},\mathcal{A}_2,\beta}(1^\lambda)$

$(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda), g, X_1, Z_1 \xleftarrow{R} \mathbb{G}_{p_1}, X_i, Y_i, Z_i \xleftarrow{R} \mathbb{G}_{p_i} (i = 2, 3),$

$T_0 = Z_1Z_3, \quad T_1 = Z_1Z_2Z_3.$

Return $\beta' \leftarrow \mathcal{A}_2(N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, X_1X_2, X_3, Y_2Y_3, T_\beta)$.

Experiment $\mathbf{Exp}^{(3)}_{\mathcal{G},\mathcal{A}_3,\beta}(1^\lambda)$

$(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda), \quad g \xleftarrow{R} \mathbb{G}_{p_1}, X_2, Y_2, Z_2 \xleftarrow{R} \mathbb{G}_{p_2}, X_3 \xleftarrow{R} \mathbb{G}_{p_3},$

$\alpha, s \xleftarrow{R} \mathbb{Z}_N, \quad T_0 = \hat{e}(g,g)^{\alpha s}, \quad T_1 \xleftarrow{R} \mathbb{G}_T.$

Return $\beta' \leftarrow \mathcal{A}_3(N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g^\alpha X_2, g^s Y_2, Z_2, X_3, T_\beta).$

For $i = 1, 2, 3$, we define the advantage of an algorithm $\mathcal{A}_i$ in breaking Assumption i to be $Adv^{(i)}_{\mathcal{G},\mathcal{A}_i}(\lambda) :=$

$$\left|\Pr[\mathbf{Exp}^{(i)}_{\mathcal{G},\mathcal{A}_i,1}(1^\lambda) = 1] - \Pr[\mathbf{Exp}^{(i)}_{\mathcal{G},\mathcal{A}_i,0}(1^\lambda) = 1]\right|.$$

Definition 1. *For $i = 1, 2, 3$, we say that $\mathcal{G}$ satisfies Assumption i if $Adv^{(i)}_{\mathcal{G},\mathcal{A}_i}(\lambda)$ is a negligible function of λ for any polynomial time algorithm $\mathcal{A}_i$.*

Goldreich-Levin Theorem for Large Fields. Dodis *et al.* [9] proved the following theorem – Goldreich-Levin theorem over any field $GF(q)$ for prime q.

Theorem 2 ([9]). *Let q be a prime, and let H be an arbitrary subset of $GF(q)$. Let $f : H^m \to \{0,1\}^*$ be any function. $\boldsymbol{s} \leftarrow H^m, y \leftarrow f(\boldsymbol{s}), \boldsymbol{r} \leftarrow GF(q)^m$. If there is a distinguisher $\mathcal{D}$ that runs in time t such that*

$$\left|\Pr[\mathcal{D}(y, \boldsymbol{r}, \langle \boldsymbol{r}, \boldsymbol{s}\rangle) = 1] - \Pr[u \leftarrow GF(q) : \mathcal{D}(y, \boldsymbol{r}, u) = 1]\right| = \epsilon,$$

then there is an inverter $\mathcal{A}$ that runs in time $t' = t \cdot \mathsf{poly}(m, |H|, 1/\epsilon)$ such that

$$\Pr[\boldsymbol{s} \leftarrow H^m, y \leftarrow f(\boldsymbol{s}) : \mathcal{A}(y) = \boldsymbol{s}] \geq \frac{\epsilon^3}{512 \cdot m \cdot q^2}.$$

Modular Lattices. Here we review some theorems for modular lattices. The first is a lemma on additive groups simplified from the lemma in [16].

Lemma 3 ([11]). *Let q be prime and let $m \geq 2n \log q$. For all but an at most q^n fraction of $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$, the subset-sums of the columns of $\boldsymbol{A}$ generate $\mathbb{Z}_q^n$; i.e., for every $\boldsymbol{u} \in \mathbb{Z}_q^n$ there exists $\boldsymbol{e} \in \{0,1\}^m$ such that $\boldsymbol{A}\boldsymbol{e} = \boldsymbol{u}$.*

We give the definition of the average-case problem of inhomogeneous small integer solution problem (ISIS), which is related to the shortest independent vectors problem and decision shortest vector problem [11].

Definition 4 ($\mathbf{ISIS}_{q,m,\beta}$). *Given an integer q, a uniformly random matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$, a random $\boldsymbol{u} \in \mathbb{Z}_n^q$, and a real β, find an integer vector $\boldsymbol{e} \in \mathbb{Z}^m$ such that $\boldsymbol{A}\boldsymbol{e} = \boldsymbol{u} \mod q$ and $||e||_2 \leq \beta$, where $||e||_2$ is the Euclidean ℓ_2 norm. We say that the $ISIS_{q,m,\beta}$ assumption holds if no polynomial time algorithm can output $\boldsymbol{e}$ with a non-negligible probability.*

3 Identity-Based Encryption with Auxiliary Inputs

An IBE scheme consists of four probabilistic polynomial-time (PPT) algorithms:

1. $\mathtt{Setup}$: On input a security parameter 1^λ, it generates a master public key mpk and a master secret key msk.
2. $\mathtt{Ext}$: On input msk and an identity ID from an identity space $\mathcal{I}$, it outputs an identity-based secret key sk_{ID}.
3. $\mathtt{Enc}$: On input $\mathsf{mpk}, \mathsf{ID}$ and a message M from a message space $\mathcal{M}$, it outputs a ciphertext $\mathfrak{C}$.
4. $\mathtt{Dec}$: On input sk_{ID}, and $\mathfrak{C}$, it outputs a message M or $\perp$ symbolizing the failure of decryption.

$\forall M \in \mathcal{M}$ and $\forall \mathsf{ID} \in \mathcal{I}$, $M \leftarrow \mathtt{Dec}(sk_{\mathsf{ID}}, \mathtt{Enc}(\mathsf{mpk}, \mathsf{ID}, M))$, where $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathtt{Setup}(1^\lambda)$, $sk_{\mathsf{ID}} \leftarrow \mathtt{Ext}(\mathsf{msk}, \mathsf{ID})$.

We denote the space of the master secret key and that of ID-based secret keys by $\mathcal{MK}$ and $\mathcal{SK}$ respectively.

3.1 Auxiliary Input Model for Confidentiality

We consider the following indistinguishability based game against adaptive chosen identity and chosen plaintext attacks (IND-ID-CPA) for semantic security with leakage in the form of auxiliary inputs. Denote a polynomial-time (in λ) computable function family $\mathcal{F}$. We define the attack game as follows.

1. Setup. The challenger runs $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathtt{Setup}(1^\lambda)$ and gives mpk to the adversary $\mathcal{A}$. The challenger also constructs an initially empty list $\mathcal{L}_{\mathsf{ID}}$.
2. $\mathsf{Query\ 1}$. The following oracles can be queried by $\mathcal{A}$:
   - Extraction Oracle $\mathcal{KEO}(\mathsf{ID}, i)$: On input $\mathsf{ID} \in \mathcal{I}, i \in \mathbb{N}^+$, it first checks the list $\mathcal{L}_{\mathsf{ID}}$ for the tuple in the form of $(sk_{\mathsf{ID}}, \mathsf{ID}, j)$. If there is no such tuple, $\bar{j}$ is set to 1, then it runs $sk_{\mathsf{ID}} \leftarrow \mathtt{Ext}(\mathsf{msk}, \mathsf{ID})$ and puts $(sk_{\mathsf{ID}}, \mathsf{ID}, \bar{j})$ in the list $\mathcal{L}_{\mathsf{ID}}$. Otherwise, the maximum j is retrieved. If $i \leq j$, then sk_{ID} from the tuple $(sk_{\mathsf{ID}}, \mathsf{ID}, i)$ in the list $\mathcal{L}_{\mathsf{ID}}$ is returned.
   - Leakage Oracle $\mathcal{LO}(f, \mathsf{ID})$: On input $f \in \mathcal{F}$, it returns $f(\mathsf{msk}, \mathcal{L}_{\mathsf{ID}}, \mathsf{mpk}, \mathsf{ID})$.
   - UpdateUSK Oracle $\mathcal{USO}(\mathsf{ID})$: This oracle is useful for schemes with probabilistic ID-based secret key generation, where a user of identity ID may request for another ID-based secret key after obtained the first copy. It first checks the list $\mathcal{L}_{\mathsf{ID}}$ for the tuple in the form of $(sk_{\mathsf{ID}}, \mathsf{ID}, j)$ where j is a positive integer. If there is no such tuple, $\bar{j}$ is set to 1. Otherwise, the maximum j is retrieved and $\bar{j}$ is set to $(j+1)$. Then, it runs $\bar{sk}_{\mathsf{ID}} \leftarrow \mathtt{Ext}(\mathsf{msk}, \mathsf{ID})$. It puts $(\bar{sk}_{\mathsf{ID}}, \mathsf{ID}, \bar{j})$ in the list $\mathcal{L}_{\mathsf{ID}}$ and returns $\bar{j}$.

   $\mathcal{KEO}, \mathcal{USO}$ and $\mathcal{LO}$ can be queried for at most q_e, q_u and q_ℓ times throughout this game respectively.
3. $\mathsf{Challenge}$. $\mathcal{A}$ sends two messages $M_0, M_1 \in \mathcal{M}$ and an identity $\mathsf{ID}^* \in \mathcal{I}$ to the challenger. The challenger picks a random bit b' and computes $\mathfrak{C}^* \leftarrow \mathtt{Enc}(\mathsf{mpk}, \mathsf{ID}^*, M_{b'})$. The challenger sends $\mathfrak{C}^*$ to $\mathcal{A}$.
4. $\mathsf{Query\ 2}$. $\mathcal{A}$ is allowed to query the Extraction Oracle adaptively.

5. Output. $\mathcal{A}$ returns a guess b^* of b'.

$\mathcal{A}$ wins the game if $b' = b^*$ and there was no query in the form of $\mathcal{KEO}(\mathsf{ID}^*, \cdot)$. The advantage of $\mathcal{A}$ is $\left|\Pr[\mathcal{A} \text{ wins}] - \frac{1}{2}\right|$. An IBE scheme is IND-ID-CPA secure w.r.t. auxiliary inputs from $\mathcal{F}$ if there is no PPT $\mathcal{A}$ with non-negligible advantage in the game above.

It remains to define a class of function families $\mathcal{F}$. For convenience, we will parametrize these families by the min-entropy k_u of the ID-based secret key respectively, as opposed to the security parameter 1^λ. (In our schemes the secret keys will be random, so k_u is simply the length of the ID-based secret key.)

Let $\mathcal{S}^*$ denote a set of all possible valid identity-based secret keys with respect to ID^*[2]. Let $\mathcal{S}$ denote a set of q_e identity-based secret keys such that $\mathcal{S}^* \cap \mathcal{S} = \emptyset$. Finally, let $\mathcal{F}_{id-ow}(g^{\mathtt{u}}(k_u))$ be the class of all polynomial-time computable functions f; such that for all $i \in [1, q_\ell]$, given

$$\mathsf{mpk}, \quad \mathsf{ID}^*, \quad \mathcal{S}, \text{ and } \quad \{f_i(\mathsf{msk}, \mathcal{L}_{\mathsf{ID}}, \mathsf{mpk}, \mathsf{ID}_i)\}_{i\in[1,q_\ell]},$$

(for $(\mathsf{msk}, \mathsf{mpk}, \{sk_{\mathsf{ID}_i}\}_{i\in[1,q_\ell]}, \mathcal{S}, \mathcal{L}_{\mathsf{ID}})$ that is randomly generated, and $\{\mathsf{ID}^*\} \cup \{\mathsf{ID}_i\}_{i\in[1,q_\ell]} \subseteq \mathcal{I}$), no PPT algorithm can find a valid secret key sk_{ID^*} of ID^* with probability greater than $g^{\mathtt{u}}(k_u)$, where $g^{\mathtt{u}}(k_u) \geq 2^{-k_u}$ is the hardness parameter. Our goal is to make $g^{\mathtt{u}}(k_u)$ as large (i.e., as close to $\mathrm{negl}(k_u)$) as possible.

Definition 5. *An identity-based encryption is said to be $(g^{\mathtt{u}}(k_u))$-AI-CPA (auxiliary input CPA) secure if it is IND-ID-CPA secure w.r.t. family $\mathcal{F}_{id-ow}(g^{\mathtt{u}}(k_u))$.*

Discussions. Our model for IBE with auxiliary inputs bears some differences from the existing model of *public key encryption (PKE) with auxiliary input* [9] and *IBE with length-bounded leakage* [13].

- For PKE, the public key itself leaks some information about the secret key. Therefore, in [9], they define the family $\mathcal{F}_{pk-ow}$ such that, given $f(\mathsf{msk}, \mathsf{mpk})$ and mpk (where $f \in \mathcal{F}_{pk-ow}$), it is difficult to output msk. For IBE, the master public key leaks some information about the master secret key, which may be exploited to compromise the security since ID-based secret key can be computed from the master secret. Therefore, we define the family $\mathcal{F}_{id-ow}$ such that given the above information, it is difficult to output sk_{ID^*}.
- The CPA security for PKE only has one single oracle which is for leakage, so adaptive leakage can be modeled by a single oracle query [9]. On the other hand, for IBE (e.g., the bounded retrieval model in [13]), the adversary can either query the leakage of msk or sk_{ID}, or unlock all bits of sk_{ID}. Thus we need to model the leakage via multiple queries for adaptive adversary.
- We combine the two separate leakage oracles in [13], for modeling the case that *the adversary may obtain leakage from msk and sk_{ID} at the same time*, and they may share the same internal randomness.

[2] A valid ID-based secret key for ID^* can decrypt all ciphertext encrypted to ID^*, hence every ID-based secret key for ID^* from `Ext` are in the set $\mathcal{S}^*$.

– Same as [13], multiple keys are allowed for each identity. For leakage oracle, we allow leakage of all these keys. Moreover, we do not need to store the amount of leakage for the master secret key and the identity-based secret keys, therefore we do not need to create a set which stores the handles of secret keys. For extraction, we could just return a new key upon each query (and store them for later leakage), but we chose to allow the adversary to supply a handle to specify which particular key to be extracted. The generality here may be useful for other higher applications of IBE, for example, those assigning keys for the same identity to different users.

3.2 Intuition

The Scheme. This construction is a parallel repetition of the Lewko-Waters IBE [14]. The key generation centre (KGC) splits the master secret key msk into m pieces $\{\alpha_i\}$. This idea can be found in many leakage-resilient schemes, e.g. [9]. The ID-based secret keys contain n components, each of which is created from a share of msk. The ciphertext also contains n components. Pairing the secret key and the ciphertext component-wise recovers a encapsulated key corresponding to α_i. Their product is the padding for hiding the actual message.

Like Lewko-Waters IBE [14] (and the underlying IBE [3]), an identity $\mathsf{ID} \in \mathbb{Z}_N$ is mapped to a group element $u^{\mathsf{ID}}h$, where $u, h \in \mathbb{G}_{p_1}$. To allow leakage of the master secret key, instead of keeping $\alpha_i \in \mathbb{Z}_N$, we store msk in a form similar to the structure of an ID-based secret key [14]. Recall that a "basic" [3] ID-based secret key contains $(g_1^{\alpha_i} \cdot (u^{\mathsf{ID}}h)^{r_i}, v_i^{r_i})$ (the second term embeds r_i for cancelling the $(u^{\mathsf{ID}}h)^{r_i}$ part in decryption), one can store $(g_1^{\alpha_i} \cdot h^{r_i}, u^{r_i}, v_i^{r_i})$ as the master secret for "undetermined" ID.

The Proof. Our proof uses the dual-system encryption technique [17,14,13]. The keys and the ciphertexts are masked by random group elements in $\mathbb{G}_{p_3}$ for adaptive security, and in the proof all these will be turned into their semi-functional (SF) version by introducing random factors from $\mathbb{G}_{p_2}$ [17]. The basic technique [14] ensures that the real key is indistinguishable to an SF key. For leakage, an SF key is further classified into two types: truly SF and nominally SF. The latter can still decrypt an SF ciphertext, but the former will make the decryption fails. Thus, a truly SF key is used to simulate the leakage oracle, which does not help the adversary.

If the adversary can distinguish between these two types of SF keys, we hope to leverage this to break the underlying assumption. In our case, we want to invert the leakage function which is supposed to be uninvertible. For Lewko-Waters IBE, it was done by applying their Lemma 6.2 for bounded leakage [14]. However, we cannot directly replace it with Theorem 2 for auxiliary input as it restricts the blinding factor of an SF key to be a λ-bit number. Therefore, these SF structures have to be changed accordingly. Since they only appear in the proof, the actual scheme is not affected.

3.3 Concrete Construction

$\mathtt{Setup}(1^\lambda)$: The KGC runs the bilinear group generator $\mathcal{G}(1^\lambda)$ to get $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e})$ as defined in §2. Suppose $\mathcal{G}$ also gives g_1 and X_3 which are generators of the subgroups $\mathbb{G}_{p_1}$ and $\mathbb{G}_{p_3}$ respectively. Let $0 < \epsilon < 1$ and $m = (3\lambda)^{1/\epsilon}$. The KGC randomly picks $\alpha_1, \ldots, \alpha_m \in \mathbb{Z}_N$, $u, h, v_1, \ldots, v_m \in \mathbb{G}_{p_1}$. The master public key is

$$\left\{N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_1, u, h, X_3, \{v_i, y_i = \hat{e}(g_1, v_i)^{\alpha_i}\}_{i\in[1,m]}\right\}.$$

The KGC also randomly picks $t_i \in \mathbb{Z}_N$, and $T_{1,i}, T_{2,i}, T_{3,i} \in \mathbb{G}_{p_3}$ for $i \in [1, m]$. The master secret key is $\{K_{1,i}, K_{2,i}, K_{3,i}\}_{i\in[1,m]}$ where

$$K_{1,i} = g_1^{\alpha_i} \cdot h^{t_i} \cdot T_{1,i}, \qquad K_{2,i} = u^{t_i} \cdot T_{2,i}, \qquad K_{3,i} = v_i^{t_i} \cdot T_{3,i}.$$

The message space $\mathcal{M}$ is $\mathbb{G}_T$ and the identity space $\mathcal{I}$ is $\mathbb{Z}_N$.

$\mathtt{Ext}(\mathsf{msk}, \mathsf{ID})$: For $i \in [1, m]$, randomly picks $r_i \in \mathbb{Z}_N$ and $R_{1,i}, R_{2,i} \in \mathbb{G}_{p_3}$, it outputs the identity-based secret key $sk_{\mathsf{ID}} = \{D_1, E_1, \ldots, D_m, E_m\}$ where

$$D_i = K_{1,i} \cdot K_{2,i}^{\mathsf{ID}} \cdot (u^{\mathsf{ID}}h)^{r_i} \cdot R_{1,i}, \qquad E_i = K_{3,i} \cdot v_i^{r_i} \cdot R_{2,i}.$$

$\mathtt{Enc}(\mathsf{ID}, M)$: For $i \in [1, m]$, it randomly picks $s_i \in \mathbb{Z}_N$ and outputs the ciphertext $\mathfrak{C} = \{A, \{B_i\}_{i\in[1,m]}, \{C_i\}_{i\in[1,m]}\}$ where

$$A = M \cdot \prod_{i=1}^{m} y_i^{s_i}, \qquad B_i = v_i^{s_i}, \qquad C_i = (u^{\mathsf{ID}}h)^{s_i}.$$

$\mathtt{Dec}(sk_{\mathsf{ID}}, \mathfrak{C})$: Given a ciphertext $\mathfrak{C} = \{A, \{B_i\}_{i\in[1,m]}, \{C_i\}_{i\in[1,m]}\}$, and a secret key $sk_{\mathsf{ID}} = \{D_1, E_1, \ldots, D_m, E_m\}$ for an identity ID, it outputs

$$M = A \cdot \frac{\prod_{i=1}^{m} \hat{e}(C_i, E_i)}{\prod_{i=1}^{m} \hat{e}(B_i, D_i)}.$$

Hierarchical Extension. Similar to existing HIBE schemes [3,14], one can extend the above IBE to n-level HIBE by extending the master public key. Due to page limitation, we just outline the modifications and omit its security proof. Specifically, mpk contains $u_1, \ldots, u_n \in \mathbb{G}_{p_1}$ instead of a single u. Accordingly, to extract a key or encrypt a message for a vector of identity $(\mathsf{ID}_1, \ldots, \mathsf{ID}_n)$ instead of just ID, $(u^{\mathsf{ID}}h)$ in the computation of D_i in $\mathtt{Ext}$ and in the computation of C_i in $\mathtt{Enc}$ is replaced by $(\prod_{j=1}^{n} u_j^{\mathsf{ID}_j} h)$. The $\mathtt{Dec}$ algorithm remains the same.

3.4 Security

Under the dual system encryption paradigm [17], we define the following three semi-functional (SF) structures which are used in the security proofs only. These

SF structures just like their normal version in the actual scheme, but "perturbed" by a $\mathbb{G}_{p_2}$ generator, denoted by either $\bar{g}_2$ or $\hat{g}_2$ below.

An *SF master-key* $\{K'_{1,i}, K'_{2,i}, K'_{3,i}\}_{i\in[1,m]}$ is given by:

$$K'_{1,i} = K_{1,i} \cdot \bar{g}_2^{\theta_i}, \qquad K'_{2,i} = K_{2,i} \cdot \bar{g}_2^{\tau\theta_i}, \qquad K'_{3,i} = K_{3,i} \cdot \bar{g}_2^{w_i},$$

where $\theta_1, \ldots, \theta_m \in [0, \lambda]$, $\tau, w_1, \ldots, w_m \in \mathbb{Z}_N$ and $\{K_{1,i}, K_{2,i}, K_{3,i}\}$ is a normal master key.

An *SF ID-based key* (or just *SF key*) is in the form of

$$\{D'_i = D_i \cdot \bar{g}_2^{\gamma_i}, \qquad E'_i = E_i \cdot \bar{g}_2^{z_i}\}_{i\in[1,m]},$$

where $z_1, \ldots, z_m, \gamma_1, \ldots, \gamma_m \in \mathbb{Z}_N$ and $\{D_i, E_i\}_{i\in[1,m]}$ is a normal ID-based key.

An *SF ciphertext* is in the form of

$$\left\{A' = A, \qquad \{B'_i = B_i \cdot \hat{g}_2^{\delta_i}, \qquad C'_i = C_i \cdot \hat{g}_2^{x_i}\}_{i\in[1,m]}\right\},$$

where $\delta_1, x_1, \ldots, \delta_m, x_m \in \mathbb{Z}_N$ and $\{A, \{B_i, C_i\}_{i\in[1,m]}\}$ is a normal ciphertext.

Decryption will succeed if an SF key is used to decrypt a normal ciphertext, or a normal key is used to decrypt an SF ciphertext. However, decrypting an SF ciphertext using an SF key will result in a message "blinded" by

$$\hat{e}(\bar{g}_2, \hat{g}_2)^{\sum_{i=1}^m z_i x_i - \sum_{i=1}^m \gamma_i \delta_i}.$$

Furthermore, the ID-based secret key generated by applying `Ext` with an SF master key is also semi-functional. If we use it to decrypt an SF ciphertext, result will be shifted by a factor

$$\hat{e}(\bar{g}_2, \hat{g}_2)^{\sum_{i=1}^m w_i x_i - \sum_{i=1}^m (1+\tau \mathsf{ID})\theta_i \delta_i}.$$

In case that the exponents in these extra blinding factors are zeros, decryption still works and this leads us to the notion of *nominally semi-functional (NSF) keys*. An NSF ID-based key is a special kind of SF key which can be used to decrypt SF ciphertext, that means $\sum_{i=1}^m \gamma_i \delta_i = \sum_{i=1}^m z_i x_i$. Similarly, an NSF master-key is a special kind of SF master-key which can be used to decrypt SF ciphertext, that means $\sum_{i=1}^m (1 + \tau \mathsf{ID})\theta_i \delta_i = \sum_{i=1}^m w_i x_i$. If an SF identity-based / master key is not nominally semi-functional, then it is *truly semi-functional*.

Theorem 6. *Our IBE scheme is* (2^{-m^ϵ})*-AI-ID-CPA secure under Assumptions 1, 2 and 3.*

Proof. We prove by a hybrid argument using a sequence of games. The first game Game_{real} is the real AI-ID-CPA game and we denote the challenge identity as ID^*. The second game $\text{Game}_{restricted}$ is the same as Game_{real} except that the adversary cannot ask for the secret key of identity ID where $\mathsf{ID} \equiv \mathsf{ID}^* \mod p_2$. This restriction will be retained throughout the subsequent games. After that,

we denote $q := q_e + q_u + q_\ell$ as the number of extraction oracle, UpdateUSK oracle and leakage oracle queries. For $k = 0$ to q, we define Game_k as follows.

Game$_k$: It is the same as Game_{real}, except that both the challenge ciphertext and the keys used to answer first k-th *distinct* oracle queries[3] are semi-functional. The keys for the rest of the queries are normal. So, for the first k-th queries:

1. If it is for extraction oracle, it returns the semi-functional key sk'_{ID}.
2. If it is for leakage oracle, it returns $f(\mathsf{msk}', \mathcal{L}_{\mathsf{ID}}, \mathsf{mpk}, \mathsf{ID})$ where msk' is semi-functional and for the last entry $(sk'_{\mathsf{ID}}, \mathsf{ID}, \cdot) \in \mathcal{L}_{ID}$, sk'_{ID} is semi-functional.
3. If it is for UpdateUSK oracle, it puts a semi-functional key sk'_{ID} into $\mathcal{L}_{ID}$.

As a result, all keys are normal and the challenge ciphertext is semi-functional in Game_0. In Game_q, all keys and the challenge ciphertext are semi-functional.

The last game is Game_{final}, which is the same as Game_q except that the challenge ciphertext is a semi-functional encryption of a random message, instead of one of the two challenge messages.

We will prove the indistinguishability between these games.

Lemma 7. *If there exists an adversary $\mathcal{A}$ such that $Adv_{\mathcal{A}}(Game_{real})$ - $Adv_{\mathcal{A}}(Game_{restricted}) = \epsilon$, then we can construct an algorithm $\mathcal{B}$ with non-negligible advantage in breaking Assumption 2.*

Lemma 8. *If there exists an adversary $\mathcal{A}$ such that $Adv_{\mathcal{A}}(Game_{restricted})$ - $Adv_{\mathcal{A}}(Game_0) = \epsilon$, then we can construct an algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 1.*

Lemma 9. *If there exists an adversary $\mathcal{A}$ such that $Adv_{\mathcal{A}}(Game_{\ell-1})$ - $Adv_{\mathcal{A}}(Game_\ell) = \epsilon$, then we can construct an algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 2.*

Lemma 10. *If there exists an adversary $\mathcal{A}$ such that $Adv_{\mathcal{A}}(Game_q)$ - $Adv_{\mathcal{A}}(Game_{final}) = \epsilon$, then we can construct an algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 3.*

The proofs of lemma 7, 8, 9 and 10 are given in the the full version.

Finally in Game_{final}, the value of b is information theoretically hidden from $\mathcal{A}$. Hence $\mathcal{A}$ has no advantage in winning Game_{final}. If Assumptions 1, 2 and 3 hold, Game_{real} is indistinguishable from Game_{final}. Hence the attacker has negligible advantage in winning Game_{real}. Therefore, our scheme is (2^{-m^ϵ})-AI-ID-CPA secure. □

[3] We consider the following parameters to determine if two queries are the same, i.e., *not distinct*. For leakage queries, we consider the function f and its argument. In particular, when they are the same, the same version of the secret key for the same ID is leaked in the same way. For extraction, we consider ID and the counter i.

4 IBE with Continual Auxiliary Inputs

4.1 Continual Auxiliary Leakage Model

We propose the continual auxiliary leakage model for IBE. First, we separate the setup algorithm into two, one for common reference string (CRS) generation and another for master key pair generation. This separation has been done previously [7] for specific security goals. It is necessary in our case since leakage is only allowed from the later part. We also introduce two additional update algorithms.

- $\mathtt{CRSGen}$: On input a security parameter 1^λ, it generates a CRS param.
- $\mathtt{Setup}$: On input a CRS param, it generates a master public key mpk and a master secret key msk. Denote the randomness used $(\mathsf{msk}, \mathsf{mpk})$ as r_s.
- $\mathtt{UpdateMSK}$: On input a master key pair $(\mathsf{msk}, \mathsf{mpk})$, it outputs a re-randomized master secret key $\bar{\mathsf{msk}}$. Denote the randomness used as r_m.
- $\mathtt{UpdateUSK}$: On input an identity-based secret key sk_{ID} for the identity ID and mpk, it outputs a re-randomized identity-based secret key $\bar{sk}_{\mathsf{ID}}$. Denote the randomness used as r_u.

After running both $\mathtt{UpdateMSK}$ and $\mathtt{UpdateUSK}$ algorithms, the corresponding public keys remain unchanged after the re-randomization; and the size of the secret keys also remain unchanged.

Denote the master secret key's, identity-based secret keys', messages' and identities' spaces as $\mathcal{MK}$, $\mathcal{SK}$, $\mathcal{M}$ and $\mathcal{I}$ respectively. Denote a polynomial-time computable function family $\mathcal{F}$. The security of IBE in the continual auxiliary leakage model is defined via the following game.

1. Setup. The challenger firstly runs $\mathsf{param} \leftarrow \mathtt{CRSGen}(1^\lambda)$ and $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathtt{Setup}(\mathsf{param})$. Denote the randomness used in $\mathtt{Setup}$ as r_s. The adversary specifies a function $f_0 \in \mathcal{F}$. Denote ϵ as an empty string. The challenger gives param, mpk and $f_0(r_s, \epsilon, \epsilon, \epsilon, \epsilon, \epsilon)$ to the adversary $\mathcal{A}$. The challenger constructs the list $\mathcal{L}_{\mathsf{msk}}$, which stores the tuples $(\mathsf{msk}, \cdot)$[4], and the lists $\mathcal{L}_{\mathsf{e}}$ and $\mathcal{L}_{\mathsf{ID}}$, which are initially empty.
2. $\mathsf{Phase\ 1}$. The following oracles can be queried by $\mathcal{A}$ adaptively:
   - Extraction Oracle $\mathcal{KEO}(\mathsf{ID})$: On input an identity $\mathsf{ID} \in \mathcal{I}$, it looks for the last $(\mathsf{ID}, sk_{\mathsf{ID}})$ entry from the list $\mathcal{L}_{\mathsf{e}}$. If such entry does not exist, it runs $sk_{\mathsf{ID}} \leftarrow \mathtt{Ext}(\mathsf{msk}, \mathsf{ID})$ and stores $(\mathsf{ID}, sk_{\mathsf{ID}})$ in the list $\mathcal{L}_e$. Finally, it returns the identity-based secret key sk_{ID}.
   - Leakage Oracle $\mathcal{LO}(f)$: On input a polynomial-time computable function $f \in \mathcal{F}$, it returns $f(\epsilon, \mathcal{L}_{\mathsf{msk}}, \epsilon, \mathsf{msk}, \epsilon, \mathsf{mpk}, \epsilon)$.
   - UpdateMSK Oracle $\mathcal{UMO}$: It runs $\bar{\mathsf{msk}} \leftarrow \mathtt{UpdateMSK}(\mathsf{msk})$. Denote the randomness used as r_m. It puts (msk, r_m) in the list $\mathcal{L}_{\mathsf{msk}}$. After that, it sets $\mathsf{msk} \leftarrow \bar{\mathsf{msk}}$ and outputs msk.

   Denote q_e, q_ℓ, q_m as the number of oracle queries to the $\mathcal{KEO}$, $\mathcal{LO}$ and $\mathcal{UMO}$ respectively in this game.

[4] An alternative definition is to include r_s in the list $\mathcal{L}_{\mathsf{msk}}$, which is part of the input for the leakage queries in the later phase.

3. Challenge Identity. $\mathcal{A}$ sends a challenge identity $\mathsf{ID}^* \in \mathcal{I}$ to the challenger. The challenger runs $sk_{\mathsf{ID}^*} \leftarrow \mathtt{Ext}(\mathsf{msk}, \mathsf{ID}^*)$.
4. Phase 2. The following oracles can be queried by $\mathcal{A}$ adaptively:
   - Extraction Oracle $\mathcal{KEO}(\mathsf{ID})$: Same as that in Phase 1.
   - Leakage Oracle $\mathcal{LO}(f)$: On input a polynomial-time computable function $f \in \mathcal{F}$, it returns $f(\epsilon, \mathcal{L}_{\mathsf{msk}}, \mathcal{L}_{\mathsf{ID}}, \mathsf{msk}, sk_{\mathsf{ID}^*}, \mathsf{mpk}, \mathsf{ID}^*)$.
   - UpdateMSK Oracle $\mathcal{UMO}$: Same as that in Phase 1.
   - UpdateUSK Oracle $\mathcal{USO}$: It runs $\bar{sk}_{\mathsf{ID}^*} \leftarrow \mathtt{UpdateUSK}(sk_{\mathsf{ID}^*})$. Denote the randomness used as r_u. It puts $(sk_{\mathsf{ID}^*}, r_u)$ in $\mathcal{L}_{\mathsf{ID}}$ and sets $sk_{\mathsf{ID}^*} = \bar{sk}_{\mathsf{ID}^*}$.

   Denote q_u as the number of oracle queries to the $\mathcal{USO}$ in this game.
5. Challenge. $\mathcal{A}$ sends two messages $M_0, M_1 \in \mathcal{M}$ to the challenger. The challenger picks a random bit b' and computes $\mathfrak{C}^* \leftarrow \mathtt{Enc}(\mathsf{mpk}, \mathsf{ID}^*, M_{b'})$.
6. Phase 3. The challenger sends $\mathfrak{C}^*$ to $\mathcal{A}$. $\mathcal{A}$ is allowed to query $\mathcal{KEO}$ adaptively.
7. Output. $\mathcal{A}$ returns a guess b^* of b'. $\mathcal{A}$ wins the game if $b' = b^*$ and there was no $\mathcal{KEO}(\mathsf{ID}^*)$ query.

The advantage of $\mathcal{A}$ is $\left|\Pr[\mathcal{A} \text{ wins}] - \frac{1}{2}\right|$. An IBE scheme is IND-ID-CPA secure in the continual auxiliary leakage model for $\mathcal{F}$ (or CAL-CPA secure) if there is no PPT $\mathcal{A}$ with non-negligible advantage in the game above.

Class of Auxiliary Functions. Let $\mathcal{S}^*$ denote a set of all possible valid identity-based secret keys with respect to ID^*. Let $\mathcal{S}$ denote a set of q_e identity-based secret keys such that $\mathcal{S}^* \cap \mathcal{S} = \emptyset$. Let $\mathcal{F}_{id-ow}(g^{\mathtt{u}}(k_u))$ be the class of all polynomial-time computable functions f; such that given

$$\mathsf{mpk}, \quad \mathsf{ID}^*, \quad \mathcal{S}, \quad f_0(r_s, \cdots) \text{ and } \{f_i(\epsilon, \mathcal{L}_{\mathsf{msk}}, \mathcal{L}_{\mathsf{ID}}, \mathsf{msk}, sk_{\mathsf{ID}^*}, \mathsf{mpk}, \mathsf{ID}^*)\}_{i \in [1, q_\ell]},$$

(for a randomly generated $(\mathsf{msk}, \mathsf{mpk}, r_s, sk_{\mathsf{ID}^*}, \mathcal{S}, \mathcal{L}_{\mathsf{msk}}, \mathcal{L}_{\mathsf{ID}})$, and $\mathsf{ID}^* \subseteq \mathcal{I}$)[5], no PPT algorithm can find a $sk_{\mathsf{ID}^*} \in \mathcal{S}^*$ with probability greater than $g^{\mathtt{u}}(k_u)$, where $g^{\mathtt{u}}(k_u) \geq 2^{-k_u}$ is the hardness parameter.

Definition 11. *An IBE scheme is said to be $(g^{\mathtt{u}}(k_u))$-CAL-CPA secure if it is IND-ID-CPA secure w.r.t. family $\mathcal{F}^{\mathtt{u}}_{id-ow}(g^{\mathtt{u}}(k_u))$.*

4.2 Construction in the Continual Auxiliary Leakage Model

We can extend our basic IBE in §3.3 to give an IBE scheme secure in the continual leakage model. The advantage is that it does not have much difference from our basic IBE. However, the extended scheme does not allow leakage during the setup phase. It implies that in the security model, the leakage f_0 is not allowed. Firstly, the common reference string is $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e})$ as generated by the bilinear group generator, i.e. $\mathtt{CRSGen}(\cdot) = \mathcal{G}()$. Then, the rest of $\mathtt{Setup}$ in §3.3 constitutes our new $\mathtt{Setup}$. Finally, we introduce the two algorithms below.

[5] The msk here is the current value of msk when the leakage oracle query for f is made. It may be changed by the UpdateMSK oracle, hence it is not a fixed value. The same applies for other variables such as $\mathcal{L}_{\mathsf{msk}}$ and $\mathcal{L}_{\mathsf{ID}}$.

`UpdateMSK`: Given $\{K_{1,i}, K_{2,i}, K_{3,i}\}$, the KGC randomly picks $t'_i \in \mathbb{Z}_N$ and $T'_{1,i}, T'_{2,i}, T'_{3,i} \in \mathbb{G}_{p_3}$, for $i \in [1, m]$. The new master secret key is defined by:

$$K'_{1,i} = K_{1,i} \cdot h^{t'_i} \cdot T'_{1,i}, \qquad K'_{2,i} = K_{2,i} \cdot u^{t'_i} \cdot T'_{2,i}, \qquad K'_{3,i} = K_{3,i} \cdot v_i^{t'_i} \cdot T'_{3,i}.$$

`UpdateUSK`: Given $sk_{\mathsf{ID}} = \{D_1, E_1, \ldots, D_m, E_m\}$ for the identity ID, it randomly picks $r'_i \in \mathbb{Z}_N$ and $R'_{1,i}, R'_{2,i} \in \mathbb{G}_{p_3}$ for $i \in [1, m]$, then the new key is given by:

$$D'_i = D_i \cdot (u^{\mathsf{ID}} h)^{r'_i} \cdot R'_{1,i}, \qquad E'_i = E_i \cdot v^{r'_i} \cdot R'_{2,i}.$$

Theorem 12. *Our IBE scheme is* (2^{-m^ϵ})*-CAL-CPA secure if Assumption 1, Assumption 2 and Assumption 3 hold.*

Compared with our basic IBE, we have to additionally simulate the oracles for updating and leak the randomness used. These updates all used random elements in $\mathbb{G}_{p_3}$, which which has no impact to the previous proof. So the proof is similar to that of our basic IBE and hence is omitted.

4.3 Further Discussions on the Continual Auxiliary Input Model

Our continual auxiliary input model extends the traditional continual memory leakage model in two dimensions. Previous definitions consider only length-bounded leakage with the requirement of secure erasure. For length-bounded leakage, continual leakage is obviously stronger than non-continual leakage since it allows more bits to be leaked, or, there cannot be arbitrarily large leakage on the same copy of the (static) secret key in the non-continual model.

Here, we consider "continual leakage without erasure" for an even stronger attack model – for example, an adversary may decide to leak more bits of the old secret key after seeing some bits of its refreshed version. But that seems to bring us back to the original non-continual scenario. If the old keys are not erased, the adversary can always choose to keep leaking the old keys even in the later "epochs" when the secret keys are refreshed. One may consider a model which allows "fine-grained" leakage, say, only allowing a particular query to leak a certain number of bits of an old key and keeping track of the number of bits leaked from the refreshed key via the same leakage function. But it might be difficult to have a clean definition for that. On the other hand, in the auxiliary input model, we can capture this stronger attack model by a simple uninvertibility condition.

Indeed, the basic auxiliary input model seems to be so "powerful" that any scheme that is secure in the basic model for a certain function family $\mathcal{F}$ is also secure in the continual auxiliary leakage model for another family $\mathcal{F}'$. However, it does not mean that the additional key-refreshing algorithms we just introduced have no significance. Instead, a careful design of these algorithms can enlarge the size of the allowed function family, which means a more general form of leakage is allowed. To see, consider an artificial scheme which "keeps state" across epochs and puts one bit of the same identity-based secret key in each version of a certain secret key. Eventually, this secret key can be recovered by leaking a single bit of

each of these keys, so any set of queries containing these leakages is ruled out by the uninvertibility condition, i.e., they are excluded from the function family. On the other hand, our scheme could allow this set of leakage queries.

4.4 Construction Supporting Leakage-Resilient Setup

Intuition. In the IBE scheme in [13], the KGC first picks an $\alpha \in \mathbb{Z}_N$ and uses n random tags (and other elements) to blind it. The master secret key contains multiple elements but α is enough for decryption of every ciphertexts. It seems leaking a part of the randomness is sufficient to break the scheme. To allow auxiliary leakage in the setup, we resort to Theorem 2 again. Our scheme picks a random $n \times m$ matrix $\boldsymbol{V}$ and multiplies it with α_i's to obtain m master public keys $\boldsymbol{Y}$, where $\alpha_i \in \{0,1\}$ for $i \in [1,n]$. Similar method is used in (LWE-based) GPV encryption in [9]. Denote the randomness $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ and the generator $\boldsymbol{g} = (g_1, \ldots, g_n)$. Roughly speaking, we set $\boldsymbol{Y} = \hat{e}(\boldsymbol{g}, \boldsymbol{V}^{\boldsymbol{\alpha}})$, where the pairing operation is taken entry-wise. The master secret key is $\boldsymbol{V}^{\boldsymbol{\alpha}} = \{\prod_{j=1}^{m} v_{i,j}^{\alpha_j}\}_{i \in [1,n]}$. In the security proof, the simulator can use the uninvertible function $F(\alpha_1, \ldots, \alpha_n)$ to output $v_{i,j}^{\alpha_j}$. It is difficult to invert if the ISIS assumption holds.

Construction.

$\mathtt{CRSGen}$: On input the security parameter 1^λ, the setup algorithm runs $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}(1^\lambda)$. We suppose the group generator $\mathcal{G}$ also gives the generators (u, h) and X_3 of the subgroups $\mathbb{G}_{p_1}$ and $\mathbb{G}_{p_3}$ respectively.

$\mathtt{Setup}$: Let $0 < \epsilon < 1$, $n = O(\lambda)$ and $m = ((n+4)\lambda)^{1/\epsilon}$. The KGC randomly picks $\alpha_1, \ldots, \alpha_m \in \{0,1\}$, a random matrix $\mathbf{V} \in \mathbb{G}_{p_1}^{n \times m}$ and a random $\mathbf{G} \in \mathbb{G}_{p_1}^{n}$:

$$\mathbf{V} = \begin{bmatrix} v_{1,1} & v_{1,2} & \ldots & v_{1,m} \\ v_{2,1} & v_{2,2} & \ldots & v_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ v_{n,1} & v_{n,2} & \ldots & v_{n,m} \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_n \end{bmatrix}.$$

Then we define $q_i = \prod_{j=1}^{m} v_{i,j}^{\alpha_j}$ for $i \in [1,n]$, and

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix}, \quad \mathbf{Y} = \begin{bmatrix} y_1 = \hat{e}(g_1, q_1) \\ y_2 = \hat{e}(g_2, q_2) \\ \vdots \\ y_n = \hat{e}(g_n, q_n) \end{bmatrix}.$$

The system parameter param is $\{N, \mathbb{G}, \mathbb{G}_T, \hat{e}, u, h, X_3, \mathbf{G}\}$. The master public key mpk is $\mathbf{Y}$. The KGC randomly picks $t_i \in \mathbb{Z}_N$ and $T_{1,i}, T_{2,i}, T_{3,i} \in \mathbb{G}_{p_3}$ for $i \in [1,n]$. The master secret key msk is $\left\{\{K_{1,i}, K_{2,i}, K_{3,i}\}_{i \in [1,n]}\right\}$ where

$$K_{1,i} = q_i \cdot h^{t_i} \cdot T_{1,i}, \qquad K_{2,i} = u^{t_i} \cdot T_{2,i}, \qquad K_{3,i} = g_i^{t_i} \cdot T_{3,i}.$$

The randomness used to generate msk are $\{\alpha_i, t_i, T_{1,i}, T_{2,i}, T_{3,i}\}$ for $i \in [1,n]$.

Define the message space $\mathcal{M}$ as $\mathbb{G}_T$ and the identity space $\mathcal{I}$ as a λ-bit integer.

`Ext`: Given the master secret key $\left\{\{K_{1,i}, K_{2,i}, K_{3,i}\}_{i\in[1,n]}\right\}$, and an identity ID, it picks a random $r_i \in \mathbb{Z}_N$ and some random $R_{1,i}, R_{2,i} \in \mathbb{G}_{p_3}$, then it calculates:

$$D_i = K_{1,i} \cdot K_{2,i}^{\mathsf{ID}} \cdot (u^{\mathsf{ID}}h)^{r_i} R_{1,i}, \qquad E_i = K_{3,i} \cdot g_i^{r_i} R_{2,i},$$

for $i \in [1, n]$. It is equivalent to $D_i = \prod_{j=1}^{m} v_{i,j}^{\alpha_j} \cdot (u^{\mathsf{ID}}h)^{\bar{r}_i} \cdot \bar{R}_{1,i}$ and $E_i = g_i^{\bar{r}_i} \cdot \bar{R}_{2,i}$, for random $\bar{r}_i \in \mathbb{Z}_N$, $\bar{R}_{1,i}, \bar{R}_{2,i} \in \mathbb{G}_{p_3}$. The output is $sk_{\mathsf{ID}} = \{D_1, E_1, \ldots, D_n, E_n\}$.

`Enc`: To encrypt a message $M \in \mathbb{G}_T$ for a user ID, for $i \in [1, n]$, it randomly picks $s_i \in \mathbb{Z}_N$ and calculates the ciphertext is $\{A, B_1, C_1, \ldots, B_n, C_n\}$ as:

$$A = M \cdot \prod_{i=1}^{n} y_i^{s_i}, \qquad B_i = g_i^{s_i}, \qquad C_i = (u^{\mathsf{ID}}h)^{s_i}.$$

`Dec`: Given $\mathfrak{C} = \{A, B_1, C_1, \ldots, B_n, C_n\}$, and $sk_{\mathsf{ID}} = \{D_1, E_1, \ldots, D_n, E_n\}$, the message is recovered from the ciphertext $\mathfrak{C}$ by:

$$M = A \cdot \frac{\prod_{i=1}^{n} \hat{e}(C_i, E_i)}{\prod_{i=1}^{n} \hat{e}(B_i, D_i)}.$$

`UpdateMSK`: Given $\left\{\{K_{1,i}, K_{2,i}, K_{3,i}\}_{i\in[1,n]}\right\}$, the KGC randomly picks $t'_i \in \mathbb{Z}_N$ and $T'_{1,i}, T'_{2,i}, T'_{3,i} \in \mathbb{G}_{p_3}$, for $i \in [1, n]$, it sets the new master secret key as:

$$K'_{1,i} = K_{1,i} \cdot h^{t'_i} \cdot T'_{1,i}, \qquad K'_{2,i} = K_{2,i} \cdot u^{t'_i} \cdot T'_{2,j}, \qquad K'_{3,i} = K_{3,i} \cdot g_i^{t'_i} \cdot T'_{3,i}.$$

`UpdateUSK`: Given $sk_{\mathsf{ID}} = \{D_1, E_1, \ldots, D_n, E_n\}$ for the identity ID, it picks some random $r'_i \in \mathbb{Z}_N$ and some random $R'_{1,i}, R'_{2,i} \in \mathbb{G}_{p_3}$ for $i \in [1, n]$, then it calculates the new identity-based secret key by:

$$D'_i = D_i \cdot (u^{\mathsf{ID}}h)^{r'_i} \cdot R'_{1,i}, \qquad E'_i = E_i \cdot v^{r'_i} \cdot R'_{2,i}.$$

Theorem 13. *Our IBE scheme is* $(2^{-m^{\epsilon}})$*-CAL-CPA secure if Assumptions 1, 2, 3 and the* $ISIS_{p_1,m,\sqrt{m}}$ *assumption hold.*

The structure of the proof is similar to the proof of Theorem 6. We prove by a hybrid argument using a sequence of games. The $\text{Game}_{restricted}$ and Game_k are almost the same as the proof of Theorem 6. After that, a new Game_{leak_i} is defined as the same as Game_q, except that $q_1, \ldots, q_i$ in the mpk are replaced by random elements in $\mathbb{G}_{p_1}$. The Game_{final} is also defined as the previous proof: the challenge ciphertext is changed to a semi-functional ciphertext encrypting a random message. The details of the proof are given in the full version.

Acknowledgement. Sherman Chow would like to thank Alfred Menezes and Dale Brydon for discussions, especially to Alfred for commenting on an earlier draft, and Jonathan Katz for pointing out the relation between the basic and continual auxiliary leakages.

References

1. Akavia, A., Goldwasser, S., Vaikuntanathan, V.: Simultaneous Hardcore Bits and Cryptography against Memory Attacks. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 474–495. Springer, Heidelberg (2009)
2. Alwen, J., Dodis, Y., Naor, M., Segev, G., Walfish, S., Wichs, D.: Public-Key Encryption in the Bounded-Retrieval Model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 113–134. Springer, Heidelberg (2010)
3. Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical Identity Based Encryption with Constant Size Ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)
4. Boneh, D., Goh, E.-J., Nissim, K.: Evaluating 2-DNF Formulas on Ciphertexts. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 325–341. Springer, Heidelberg (2005)
5. Boneh, D., Katz, J.: Improved Efficiency for CCA-Secure Cryptosystems Built Using Identity-Based Encryption. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 87–103. Springer, Heidelberg (2005)
6. Brakerski, Z., Kalai, Y.T., Katz, J., Vaikuntanathan, V.: Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In: FOCS 2010. IEEE Computer Society (2010)
7. Chow, S.S.M.: Removing Escrow from Identity-Based Encryption. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 256–276. Springer, Heidelberg (2009)
8. Chow, S.S.M., Dodis, Y., Rouselakis, Y., Waters, B.: Practical leakage-resilient identity-based encryption from simple assumptions. In: Al-Shaer, E., Keromytis, A.D., Shmatikov, V. (eds.) CCS 2010, pp. 152–161. ACM (2010)
9. Dodis, Y., Goldwasser, S., Kalai, Y.T., Peikert, C., Vaikuntanathan, V.: Public-Key Encryption Schemes with Auxiliary Inputs. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 361–381. Springer, Heidelberg (2010)
10. Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D.: Cryptography against continuous memory attacks. In: FOCS 2010, pp. 511–520. IEEE Computer Society (2010)
11. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Dwork, C. (ed.) STOC 2008, pp. 197–206. ACM (2008)
12. Lewko, A.B., Lewko, M., Waters, B.: How to leak on key updates. In: Fortnow, L., Vadhan, S.P. (eds.) STOC 2011, pp. 725–734. ACM (2011)
13. Lewko, A.B., Rouselakis, Y., Waters, B.: Achieving Leakage Resilience through Dual System Encryption. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 70–88. Springer, Heidelberg (2011)
14. Lewko, A., Waters, B.: New Techniques for Dual System Encryption and Fully Secure HIBE with Short Ciphertexts. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 455–479. Springer, Heidelberg (2010)
15. Naor, M., Segev, G.: Public-Key Cryptosystems Resilient to Key Leakage. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 18–35. Springer, Heidelberg (2009)
16. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) STOC 2005, pp. 84–93. ACM (2005)
17. Waters, B.: Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)