— An extended abstract of this work appears in the proceedings of PKC 2012 —

From Selective to Full Security: Semi-Generic Transformations in the Standard Model

Michel Abdalla¹

Dario $Fiore^2$

Vadim Lyubashevsky¹

¹ Département d'Informatique, École normale supérieure {Michel.Abdalla,Vadim.Lyubashevsky}@ens.fr ² Department of Computer Science New York University fiore@cs.nyu.edu

Abstract

In this paper, we propose an efficient, standard model, semi-generic transformation of selectivesecure (Hierarchical) Identity-Based Encryption schemes into fully secure ones. The main step is a procedure that uses admissible hash functions (whose existence is implied by collision-resistant hash functions) to convert any selective-secure *wildcarded* identity-based encryption (WIBE) scheme into a fully secure (H)IBE scheme. Since building a selective-secure WIBE, especially with a selective-secure HIBE already in hand, is usually much less involved than directly building a fully secure HIBE, this transform already significantly simplifies the latter task. This black-box transformation easily extends to schemes secure in the Continual Memory Leakage (CML) model of Brakerski et al. (FOCS 2010), which allows us obtain a new fully secure IBE in that model. We furthermore show that if a selective-secure HIBE scheme satisfies a particular security notion, then it can be generically transformed into a selective-secure WIBE. We demonstrate that several current schemes already fit this new definition, while some others that do not obviously satisfy it can still be easily modified into a selective-secure WIBE.

Keywords: Selective security, full security, identity-based encryption.

Contents

1	Introduction	1
	1.1 Our results	2
2	Basic Definitions 2.1 Code-Based Games	4 5 5 6 7
3	Fully-Secure HIBE from Selective-Secure WIBE 3.1 Our transformation 3.2 Extensions of our transformation	9 11 18
4	Selective WIBE schemes from selective HIBE 4.1 Security under Correlated Randomness for HIBE 4.2 From HIBE selective-secure under Correlated Randomness to selective-secure WIBE 4.2.1 Our WIBE scheme 4.2.2 A sufficient distribution for building a WIBE	 19 20 22 23 24
5	A leakage-resilient WIBE scheme based on Decision Linear	25
6	Lattice-Based WIBE6.1Lattices and the LWE Problem6.2Algorithms used in constructing the HIBE and WIBE6.3Our Lattice-Based WIBE scheme6.4Security Proof	31 32 32 33 34
7	Future Directions	35
Re	eferences	36
\mathbf{A}	A proof without artificial abort	38
в	HIBE Schemes Selective-Secure under Correlated Randomness B.1 The case of the Boneh-Boyen HIBE [8, 1]	46 46

1 Introduction

The concept of identity-based encryption (IBE) is a generalization of the standard notion of publickey encryption in which the sender can encrypt messages to a user based only on the identity of the latter and a set of user-independent public parameters. In these systems, there exists a trusted authority, called private key generator, that is responsible for generating decryption keys for all identities in the system. Since being introduced by Shamir in 1984 [33], IBE has received a lot of attention due to the fact that one no longer needs to maintain a separate public key for each user. Despite being an attractive concept, it was only in 2001 that the first practical IBE construction was proposed based on elliptic curve pairings [13]. Later that year, Cocks proposed an alternative IBE construction based on the quadratic residuosity problem [23].

The now-standard definition of security of IBE schemes, first suggested by Boneh and Franklin [13], is indistinguishability under adaptive chosen-identity attacks (we refer to it as *full security*). In this security model, the adversary is allowed to obtain secret keys for adaptively chosen identities before deciding the identity upon which it wishes to be challenged. By allowing these queries, this notion implicitly captures resistance against collusion attacks as different users should be unable to combine their keys in an attempt to decrypt ciphertexts intended to another user.

In 2002, Horwitz and Lynn introduced the notion of hierarchical identity-based encryption (HIBE), which allows intermediate nodes to act as private key generators. They also provided a two-level HIBE construction based on the Boneh-Franklin IBE scheme, but their scheme could provide full collusion resistance only in the upper level. The first HIBE scheme to provide full collusion resistance in all levels is due to Gentry and Silverberg [26]. Like the Horwitz-Lynn HIBE scheme, the Gentry-Silverberg HIBE scheme was also based on the Boneh-Franklin IBE scheme and proven secure in the random-oracle model [6].

The first HIBE to be proven secure in the standard model is due to Canetti, Halevi, and Katz [20], but in a weaker security model, called the *selective-identity* model. Unlike the security definitions used in previous constructions of (H)IBE schemes, the selective-identity model requires the adversary to commit to the challenge identity before obtaining the public parameters of the scheme. Despite providing weaker security guarantees, Canetti, Halevi, and Katz showed that the selective-identity model is sufficient for building forward-secure encryption schemes, which was the main motivation of their paper.

Although the selective-identity model has been considered in many works, and is interesting in its own right (e.g., it implies forward-secure public key encryption), if we focus solely on the (H)IBE application, then the selective notion is clearly unrealistic because it does not model the real capabilities of an adversary attacking a (H)IBE scheme. So while the design of selective-identity secure schemes seems to be an easier task, the quest for fully secure solutions is always considered the main goal for (H)IBE construction.

It is therefore a very interesting problem to investigate whether there are ways to efficiently convert a selective secure scheme into a fully secure one. In the random oracle model, this question has been resolved by Boneh, Boyen and Goh [10], who provided a very efficient black-box transformation. In the standard model, however, no such conversion is known¹, and all fully-secure (H)IBE schemes (e.g., [9], [35], [22]) had to be constructed and proved secure essentially from scratch.

¹It was shown by Boneh and Boyen in [8] that any selective secure IBE scheme is already fully secure, but the concrete security degrades by a factor $1/|I\mathcal{D}|$, where $I\mathcal{D}$ is the scheme's identity space. Since $I\mathcal{D}$ is usually of exponential size, this conversion is too expensive in terms of efficiency to be considered practical.

1.1 Our results

In this paper, we explore the relationship between selective-identity and fully secure (H)IBE schemes in the standard model.

From selective-secure WIBE to fully-secure HIBE. Our first main contribution is a generic construction of fully-secure HIBE schemes from selective-pattern-secure wildcarded identity-based encryption (WIBE) schemes. The notion of a WIBE, introduced by Abdalla et al. [1], is very similar to the notion of a HIBE except that the sender can encrypt messages not only to a specific identity, but to a whole range of receivers whose identities match a certain pattern defined through a sequence of fixed strings and a special wildcard symbol (*). The security notion, called selective-pattern security, requires the adversary to commit ahead of time to the pattern P^* that he intends to attack. He can then ask for the secret keys of any identity not matching P^* , and for the challenge ciphertext on any pattern P matching P^* . This notion of security is slightly more general and natural than that given in [1]. Yet, as noted in Remark 2.5 at the end of Section 2, it is satisfied by all known WIBE constructions.

Our transformation from *any* selective-pattern-secure WIBE to a fully-secure HIBE is generic and relies on the notion of admissible hash functions (whose existence is implied by collision-resistant hash functions) introduced by Boneh and Boyen in [9]. Since building selective-pattern-secure WIBE schemes seems to be much easier than directly building a fully secure HIBE scheme, this transformation already significantly simplifies the latter task. In fact, it is worth noticing that the selective-pattern security of all currently-known instantiations of WIBE schemes follows from the selective-identity security of their respective underlying HIBE schemes (see [1]).

One direct consequence of our construction is that several existing fully secure (H)IBE schemes can be seen as a particular case of our transformation. For instance, the fully secure IBE scheme of Boneh and Boyen in [9] turns out to be a particular case of our generic construction when instantiated with the selective-pattern-secure Boneh-Boyen WIBE scheme given in [1]. Likewise, the fully secure HIBE by Cash, Hofheinz, Kiltz, and Peikert [22] can be seen as the result of our generic transformation when applied to our new WIBE scheme in Section 6. Another consequence of our transformation is that one can obtain new constructions of fully secure HIBE schemes by applying our methodology to existing selective-pattern-secure WIBE schemes, such as the Boneh-Boyen-Goh WIBE in [1]. Interestingly, the result obtained from this instantiation closely resembles the Waters (H)IBE scheme [35].

THE TRANSFORMATION IN THE CONTINUAL MEMORY LEAKAGE MODEL. An important point about our transformation from WIBE to (H)IBE is that it also works in the Continual Memory Leakage (CML) model [19, 24]. In this model, security is defined with respect to an adversary that may learn a bounded number of bits related to the secret information of a user, such as his secret key, over a given time period. In particular, secret keys are updated regularly and information about new secret keys and the randomness used during their updates may also leak to the adversary. In [19], Brakerski et al. extended the IBE construction in [17] to obtain a selective-secure IBE in the CML model based on the Decision Linear assumption. While Brakerski and Kalai's IBE construction can be made fully secure using admissible hash functions as suggested in [17], a similar result is not known to hold in the CML model. In this paper, we show how to modify the scheme in [19] into a WIBE scheme and prove it selective-pattern-secure in the CML model under the same assumption. Then, by applying our transformation to this newly-constructed WIBE, we obtain a (CML) fullysecure version of the IBE in [19]. As in the original IBE, our new IBE construction assumes that there is no leakage from the master secret key. We observe, however, that this restriction is not that critical because, in the case of IBE, it may be reasonable to assume that the key generation center uses strong countermeasures to avoid leaking secret information.

THE ROLE OF WIBE IN OUR TRANSFORMATION. Somewhat surprisingly, our transformation seems to imply that the WIBE notion is of central importance when going from selective to full security in (H)IBE. To see why, one has to take a look at our proof strategy and at the notion of Admissible hash functions (AHF). AHFs are a tool which allows to partition the identity space into two subsets, B and R (both of which are of exponential size) so that in the security proof the identities of secret key queries fall in B while the challenge identity falls in R. In particular, by carefully selecting the AHFs parameters (as described in [9], for instance) one can make sure that the above (good) event occurs with non-negligible probability. In our proof from selective-secure WIBE to fully-secure HIBE, the simulator first uses AHFs to partition the identity space into B and R. Next, it declares to the WIBE challenger a challenge pattern which corresponds to R, by expressing R in the form of a pattern. By the property of AHFs, if the good event occurs (for all key derivation queries and the challenge identity chosen by the adversary), then the simulator can easily forward all queries to the WIBE challenger. In particular, it is guaranteed that the challenge identity falls in R. When that happens, the simulator can output the challenge identity chosen by the adversary as its own challenge.

We remark that the proof strategy described above does not work if one starts from a selectivesecure HIBE instead of a WIBE. Unlike the selective-WIBE simulator, the simulator against the selective security of a HIBE should commit to the challenge identity ID^* at the very beginning. And even if the simulator chooses the AHFs parameters so that all secret key queries fall in B and the challenge identity falls in R, it still needs to guess ID^* in R at the very beginning. But the probability that the challenge identity chosen by the adversary matches such ID^* is $1/|\mathbf{R}|$, which is negligible (recall that both B and R are of exponential size).

Selective WIBE from selective HIBE. The second main contribution of this paper is to identify conditions under which we can generically transform a selective-identity-secure HIBE scheme into a selective-pattern-secure WIBE scheme. Towards this goal, we introduce a new notion of security for HIBE schemes, called security under correlated randomness, which allows us to transform a given HIBE into a WIBE by simply re-encrypting the same message to a particular set of identities by reusing the same randomness. Informally speaking, in order for a HIBE scheme to be secure under correlated randomness, it must satisfy the following two properties. First, when given an encryption of the same message under the same randomness for two identity vectors $ID_0 = (ID_{0,1}, \ldots, ID_{0,j}, \ldots, ID_{0,\lambda})$ and $ID_1 = (ID_{1,1}, \ldots, ID_{1,j}, \ldots, ID_{1,\lambda})$ differing in exactly one position (say j), one can easily generate a ciphertext for any identity vector matching the pattern $ID = (ID_{1,1}, \ldots, *, \ldots, ID_{1,\lambda})$. Secondly, when given these two ciphertexts, the adversary should not be able to generate an encryption of the same message under the same randomness for any identity vector that does not match the pattern. In Section 4 we show that selective-correlatedrandomness-secure HIBE schemes can be converted to selective-pattern-secure WIBEs. Moreover, in Appendix B, we show that several existing HIBE schemes already satisfy this slightly stronger notion of security, e.g., [8, 10, 35], and in particular we show that their security under correlated randomness black-box reduces to their selective-identity security.

Hence, if we combine our first generic transformation from selective-pattern-secure WIBE to fully-secure (H)IBE, together with our second result described above, we obtain a compiler that allows us to construct a fully secure (H)IBE starting from a selective-secure (H)IBE. In particular, the resulting transformation works in the standard model and is semi-generic because the second part assumes a specific property of the underlying scheme (i.e., security under correlated randomness). Nevertheless, by reducing the task of building fully secure HIBE schemes to that of building a selective-pattern-secure WIBE scheme, we believe that our result makes the former task significantly easier to achieve.

New WIBE schemes. One final contribution of this paper are two constructions of selective-

pattern-secure WIBE schemes.

The first one, whose description is given in Section 5, is obtained by modifying the IBE in [19]. It is based on pairings and is secure under the Decision Linear assumption in the CML model. Such modification essentially follows the correlated-randomness paradigm. Since for some technical reasons (related to the specific scheme) the selective-pattern security of this WIBE cannot be blackbox reduced to the selective-identity security of the related IBE (like we do for other pairing-based WIBEs), we decided to give a direct proof under the Decision Linear assumption. However, we notice that such proof closely follows the one in [19].

The second WIBE is based on lattices and its security follows from the selective-identity secure HIBE construction from [22]. Even though the Cash-Hofheinz-Kiltz-Peikert HIBE scheme does not meet the notion of security under correlated randomness introduced in Section 4 (because the scheme is not secure when the same randomness is reused for encryption), we show in Section 6 that one can easily modify it to obtain a selective-pattern-secure WIBE scheme. Similarly to the case of pairing-based WIBE schemes, the selective-pattern security of the new WIBE can be reduced directly to the selective-identity security of the original Cash-Hofheinz-Kiltz-Peikert HIBE scheme. However, in this case, it turns out to be even simpler to prove the selective-pattern security of our scheme directly from the decisional Learning With Errors Problem (LWE) [32, 31].

Discussion. In this paper, we concentrate on building HIBE schemes that are adaptive-identitysecure against chosen-plaintext attacks. As shown by Boneh, Canetti, Halevi, and Katz [21, 15, 12], such schemes can easily be made chosen-ciphertext-secure with the help of one-time signature schemes or message authentication codes. Similarly to the (H)IBE schemes by Boneh and Boyen [9], by Waters [35], and by Cash, Hofheinz, Kiltz, and Peikert [22], the schemes obtained via our transformation are only provably secure when the maximum hierarchy's depth L is some fixed constant due to the loss of a factor which is exponential in L. While for lattice-based HIBE schemes [22, 3, 4], this seems to be the state of the art, the same is not true for pairing-based HIBE schemes. More precisely, there have been several proposals in recent years (e.g., [25, 34, 29, 28]), which are fully secure even when the HIBE scheme has polynomially many levels. Most of these schemes use a new proof methodology, known as dual system encryption [34].

Organization. The paper is organized as follows. In Section 2, we start by recalling some standard definitions and notations used throughout the paper. Next, in Section 3, we present our first main contribution, which is a generic construction which can transform any selective-pattern-secure WIBE into a fully secure HIBE scheme. Then, in Section 4, we introduce the notion of security under correlated randomness for HIBE schemes and show how such schemes can be used to build selective-pattern-secure WIBEs. Though such security notion does not necessarily hold for all HIBE schemes, we show in Appendix B that several existing selective-pattern-secure WIBE schemes do meet this notion. Next, in Sections 5 and 6, we show two selective-pattern-secure WIBE schemes that are obtained by transforming, respectively, the Brakerski-Kalai-Katz-Vaikuntanathan IBE and the Cash-Hofheinz-Kiltz-Peikert HIBE. Finally, in Section 7, we summarize some future directions left open by our work.

2 Basic Definitions

In this section we describe the notation and the basic definitions that we use in the paper.

Notation. We say that a function is negligible if it vanishes faster than the inverse of any polynomial. If S is a set, then $x \stackrel{\$}{\leftarrow} S$ indicates the process of selecting x uniformly at random over S. If $A(\cdot)$ is an algorithm then we denote with $y \stackrel{\$}{\leftarrow} A(\cdot)$ the operation of running A (on some input) and assigning the output to y. For any $\ell \in \mathbb{N}$ we denote with $[\ell]$ the set $\{1, 2, \ldots, \ell\}$. "PPT" stands for probabilistic polynomial time and "PTA" for PPT algorithm or adversary.

2.1 Code-Based Games

In this work, we state our definitions and give our proofs using code-based games [7]. A game is usually defined by two procedures **Initialize** and **Finalize**, and by other procedures that model the answers to the adversary's oracle queries. A game G is executed with an adversary \mathcal{A} as follows. First, \mathcal{A} runs **Initialize**, and gets its output. Then, \mathcal{A} can make oracle queries by executing the corresponding procedures. At the end, before halting, the adversary is required to execute the procedure **Finalize** whose output is the output of the game G. If b is G's output, then we denote all this process by writing $G^{\mathcal{A}} \Rightarrow b$. Usually, a game keeps a flag bad which is initialized to false, and that may be set true during the execution of the game. We denote with Bad_i (resp. Good_i) the event that $G_i^{\mathcal{A}}$ sets (resp. does not set) bad \leftarrow true.

Two games G_i and G_j are said "identical-until-bad" if their code differs only in statements that are executed when bad is set. Bellare and Rogaway show in [7] that if G_i and G_j are identical-until-bad, and \mathcal{A} is an adversary, then $\Pr[\mathsf{Bad}_i] = \Pr[\mathsf{Bad}_j]$. Moreover, the fundamental lemma of game-playing [7] states that if G_i and G_j are identical-until-bad, then for any b: $|\Pr[G_i^{\mathcal{A}} \Rightarrow b] - \Pr[G_j^{\mathcal{A}} \Rightarrow b]| \leq \Pr[\mathsf{Bad}_i]$.

In our work we use a variant of this lemma formulated by Bellare and Ristenpart in [5]:

Lemma 2.1 [[5]] If G_i and G_j are identical-until-bad games, and A is an adversary, then for any b:

$$\Pr[\mathbf{G}_i^{\mathcal{A}} \Rightarrow b \land \neg \mathsf{Bad}_i] = \Pr[\mathbf{G}_i^{\mathcal{A}} \Rightarrow b \land \neg \mathsf{Bad}_j].$$

2.2 (Hierarchical) Identity Based Encryption

A hierarchical identity-based encryption scheme (HIBE) is defined by a tuple of algorithms $\mathcal{H}I\mathcal{BE} = ($ Setup, KeyDer, Enc, Dec), a message space \mathcal{M} , and an identity space $I\mathcal{D}$. The algorithm Setup is run by a trusted authority to generate a pair of keys (mpk, msk) such that mpk is made public, whereas msk is kept private. The users are hierarchically organized in a tree of depth L whose root is the trusted authority. The identity of a user at level $1 \leq \ell \leq L$ is represented by a vector $\overrightarrow{ID} = (ID_1, \ldots, ID_\ell) \in I\mathcal{D}^\ell$. A user at level ℓ with identity $\overrightarrow{ID} = (ID_1, \ldots, ID_\ell)$ can use the key derivation algorithm KeyDer $(sk_{\overrightarrow{ID}}, \overrightarrow{ID'})$ to generate a secret key for any of its children $\overrightarrow{ID'} = (ID_1, \ldots, ID_\ell, ID_{\ell+1})$ at level $\ell + 1$. Since this process can be iterated, every user can generate keys for all its descendants. Then, every user holding the master public key mpk, can encrypt a message $m \in \mathcal{M}$ for the identity \overrightarrow{ID} by running $C \stackrel{\$}{\leftarrow} \operatorname{Enc}(mpk, \overrightarrow{ID}, m)$. Finally, the ciphertext C can be decrypted by running the deterministic decryption algorithm, $m \leftarrow \operatorname{Dec}(sk_{\overrightarrow{ID'}}, C)$.

For correctness, it is required that for all honestly generated master keys $(mpk, msk) \stackrel{*}{\leftarrow} \mathsf{Setup}$, for all messages $m \in \mathcal{M}$, all identities $\overrightarrow{ID} \in I\mathcal{D}^{\ell}$ and all \overrightarrow{ID}' ancestors of \overrightarrow{ID} ,

$$m \leftarrow \mathsf{Dec}(\mathsf{KeyDer}(msk, \overrightarrow{ID}'), \mathsf{Enc}(mpk, \overrightarrow{ID}, m))$$

holds with overwhelming probability. An IBE is defined as an HIBE with a hierarchy of depth 1.

The security of a HIBE scheme is captured by the standard notion of indistinguishability under chosen-plaintext attacks. In particular, this is formalized by a game, IND-HID-CPA, that we recall in Figure 1 using the notation of code-based games. The game is defined by four procedures that can be run by an adversary \mathcal{A} and works as follows. As usual, \mathcal{A} starts by executing **Initialize** and runs **Finalize** before halting. We assume that \mathcal{A} makes at most one query $(\overrightarrow{ID}^*, m_0, m_1)$ to

Game IND-HID-CPA		Game IND-sHID-CPA
$\begin{array}{ c c c } \hline \mathbf{procedure Initialize} \\ \hline (mpk, msk) \stackrel{\$}{\leftarrow} Setup \\ \beta \stackrel{\$}{\leftarrow} \{0, 1\} \\ \texttt{Return } mpk \\ \hline \mathbf{procedure Extract}(\overrightarrow{ID}) \\ \hline sk_{\overrightarrow{ID}} \stackrel{\$}{\leftarrow} KeyDer(msk, \overrightarrow{ID}) \\ \texttt{Return } sk_{\overrightarrow{ID}} \end{array}$	$\frac{\mathbf{procedure LR}(\overrightarrow{ID}, m_0, m_1)}{C \stackrel{*}{\leftarrow} Enc(mpk, \overrightarrow{ID}, m_\beta)}$ Return C $\frac{\mathbf{procedure Finalize}(\beta')}{\text{Return } (\beta' = \beta)}$	$\frac{\text{procedure Initialize}(\overrightarrow{ID}^*)}{(mpk, msk) \stackrel{\$}{\leftarrow} \text{Setup; } \beta \stackrel{\$}{\leftarrow} \{0, 1\}$ Return mpk $\frac{\text{procedure LR}(m_0, m_1)}{C \stackrel{\$}{\leftarrow} \text{Enc}(mpk, \overrightarrow{ID}^*, m_\beta)}$ Return C

Figure 1: On the left the definition of Game IND-HID-CPA. On the right the procedures **Initialize** and **LR** of the game IND-sHID-CPA. Notice that in the latter game the procedures **Extract** and **Finalize** are the same as those of game IND-sHID-CPA.

the **LR** procedure, under the requirement that $|m_0| = |m_1|$ (i.e., the two messages have the same length), and that all the identities submitted to **Extract** and **LR** are *legitimate*. For this notion, a set of queries is said *legitimate* if \mathcal{A} never queries **Extract** on an identity \overrightarrow{ID} such that $\overrightarrow{ID} = \overrightarrow{ID}^*$ or \overrightarrow{ID} is an ancestor of \overrightarrow{ID}^* . We define the IND-HID-CPA-advantage of any adversary \mathcal{A} against a HIBE scheme \mathcal{HIBE} as

$$\mathbf{Adv}_{\mathcal{HIRF}}^{\mathrm{IND-HID-CPA}}(\mathcal{A}) = 2 \cdot \Pr[\mathrm{IND-HID-CPA}^{\mathcal{A}} \Rightarrow 1] - 1$$

where IND-HID-CPA^{\mathcal{A}} \Rightarrow 1 denotes that a run of the IND-HID-CPA with adversary \mathcal{A} outputs 1.

Definition 2.2 [IND-HID-CPA-security] A HIBE scheme is IND-HID-CPA-secure if for any PPT adversary \mathcal{A} , $\mathbf{Adv}_{\mathcal{HIBE}}^{\mathrm{IND-HID-CPA}}(\mathcal{A})$ is at most negligible.

In the context of hierarchical identity-based encryption a lot of works in the literature also considered a weaker notion of security, called *selective-identity* indistinguishability under chosenplaintext attacks (IND-sHID-CPA). The main difference with the standard IND-HID-CPA notion is that here the adversary is required to commit ahead of time to the identity that he will use to query the **LR** procedure. The corresponding game is recalled in Figure 1, on the right. Precisely, we describe only the procedures **Initialize** and **LR**, as **Extract** and **Finalize** remain the same as in the game IND-HID-CPA. The IND-sHID-CPA-advantage of any adversary \mathcal{A} against a HIBE scheme \mathcal{HIBE} is defined as

$$\mathbf{Adv}_{\mathcal{HIBE}}^{\mathrm{IND-sHID-CPA}}(\mathcal{A}) = 2 \cdot \Pr[\mathrm{IND-sHID-CPA}^{\mathcal{A}} \Rightarrow 1] - 1$$

Definition 2.3 [IND-sHID-CPA-security] A HIBE scheme is IND-sHID-CPA-secure if for any PPT adversary \mathcal{A} , $\mathbf{Adv}_{\mathcal{HIBE}}^{\mathrm{IND-sHID-CPA}}(\mathcal{A})$ is at most negligible.

Sometimes, in order to have a clear distinction with the standard notion of IND-HID-CPA, the latter is called "full security".

2.3 Identity Based Encryption with Wildcards

The notion of *Identity-Based Encryption with Wildcards* was introduced by Abdalla *et al.* in [1] as a generalization of the HIBE's notion. A WIBE scheme is defined by a tuple of algorithms $\mathcal{W}I\mathcal{BE} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ that works exactly as a HIBE, except that here the encryption

algorithm takes as input a value $P \in (I\mathcal{D} \cup *)^{\ell}$ (for $1 \leq \ell \leq L$), i.e., the pattern, instead of an identity vector. Such pattern may contain a special "don't care" symbol *, the wildcard, at some levels. An identity $\overline{ID} = (ID_1, \ldots, ID_{\ell}) \in I\mathcal{D}^{\ell}$ is said to match a pattern $P \in (I\mathcal{D} \cup *)^{\ell'}$, denoted as $\overline{ID} \in *P$, if and only if $\ell \leq \ell'$ and $\forall i = 1, \ldots, \ell$: $ID_i = P_i$ or $P_i = *$. Note that under this definition, any ancestor of a matching identity is also a matching identity. This makes sense for the notion of WIBE, as any ancestor can derive the secret key of a matching descendant identity anyway. For any pattern $P \in (I\mathcal{D} \cup *)^{\ell}$, we denote with W(P) the set of indices $j \in [\ell]$ such that $P_j = *$. For correctness, it is required that for all honestly generated master keys $(mpk, msk) \stackrel{\$}{=} \text{Setup}$, for all messages $m \in \mathcal{M}$, all patterns $P \in (I\mathcal{D} \cup *)^{\ell'}$ and all identities $\overline{ID} \in I\mathcal{D}^{\ell}$ such that $\overline{ID} \in *P$, $m \leftarrow \text{Dec}(\text{KeyDer}(msk, \overline{ID}), \text{Enc}(mpk, P, m))$ holds with all but negligible probability.

	$\begin{vmatrix} \mathbf{procedure } \mathbf{LR}(P, m_0, m_1) \\ C \stackrel{\$}{\leftarrow} Enc(mpk, P, m_\beta) \\ \text{Return } C \end{vmatrix}$
$\frac{\textbf{procedure Extract}(\overrightarrow{ID})}{sk_{\overrightarrow{ID}} \stackrel{\hspace{0.1em} \ast}{\leftarrow} KeyDer(msk,\overrightarrow{ID})}$ Return $sk_{\overrightarrow{ID}}$	$\frac{\text{procedure Finalize}(\beta')}{\text{Return } (\beta' = \beta)}$

Figure 2: Game IND-sWID-CPA.

Similarly to HIBE, WIBE allows for similar notions of security under chosen-plaintext attacks. In particular, in our work we consider only the notion of selective security. Roughly speaking, it is similar to the IND-sHID-CPA notion for HIBE, except that here the adversary has to commit to a pattern P^* at the beginning of the game. Next, when he calls the **LR** procedure, he can provide a pattern P that matches P^* , i.e., such that either P is an identity matching P^* , or P is a sub-pattern of P^* . The security notion is formalized by the game IND-sWID-CPA in Figure 2. So, we define the IND-sWID-CPA-advantage of any adversary \mathcal{A} against a WIBE scheme \mathcal{WIBE} as

$$\operatorname{Adv}_{WIBE}^{\operatorname{IND-sWID-CPA}}(\mathcal{A}) = 2 \cdot \Pr[\operatorname{IND-sWID-CPA}^{\mathcal{A}} \Rightarrow 1] - 1$$

Definition 2.4 A WIBE scheme is IND-sWID-CPA-secure if $\mathbf{Adv}_{\mathcal{WIBE}}^{\mathrm{IND-sWID-CPA}}(\mathcal{A})$ is negligible for any PTA \mathcal{A} .

Remark 2.5 We notice that our notion of selective-security for WIBE schemes is slightly more general than the one that was originally proposed in [1]. The main difference is that in the original work of Abdalla *et al.* the notion is purely selective, meaning that the adversary declares the challenge pattern P^* at the beginning of the game, and later it receives an encryption of either m_0 or m_1 under P^* . Instead, our notion allows for more flexibility. Indeed, the adversary still declares P^* at the beginning of the game, but later it may ask the challenge ciphertext on a pattern P, possibly different from P^* , but such that P matches P^* . We stress that this property is not artificial for at least two reasons. First, it is more general than the previous one. Second, it is satisfied by all known WIBE schemes, and in particular we will show that it is satisfied by those schemes obtained through our transformation, from selective-secure HIBE to selective WIBE, that we describe in Section 4.

2.4 The Continual Memory Leakage Model

In this section we present an extension of the definitions of hierarchical identity-based encryption and wildcarded identity-based encryption in the Continual Memory Leakage (CML) Model proposed

Game CML-IND-HID-CPA		
	procedure Leak (f, \overrightarrow{ID})	
procedure Initialize	If $\overrightarrow{ID} \neq \overrightarrow{ID}^*$ and \overrightarrow{ID} not ancestor of \overrightarrow{ID}^* Then	
$(mpk, msk) \stackrel{\$}{\leftarrow} Setup$ $\beta \stackrel{\$}{\leftarrow} \{0, 1\}$ $C[\overrightarrow{ID}] \leftarrow \bot \forall \overrightarrow{ID}$ $L^{[\overrightarrow{ID}]} \leftarrow 0 \lor \lor \overrightarrow{ID}$	Return \perp Else continue let $i \leftarrow C[\overrightarrow{ID}]$ If $i = \perp$ Then	
$\begin{array}{c} L[ID,0] \leftarrow 0 \ \forall ID \\ \text{Return } mpk \end{array}$	$sk_{\overrightarrow{ID},0} \stackrel{\$}{\leftarrow} KeyDer(msk,\overrightarrow{ID})$ $C[\overrightarrow{ID}] \leftarrow 0$	
$\frac{\text{procedure Extract}(\overrightarrow{ID})}{\text{If } C[\overrightarrow{ID}] = \bot \text{ Then}}$ $sk_{\overrightarrow{ID},0} \stackrel{\text{\$}}{\leftarrow} \text{KeyDer}(msk,\overrightarrow{ID})$ $C[\overrightarrow{ID}] \leftarrow 0$	$\begin{split} & \text{If } L[\overrightarrow{ID},i] + f(sk_{\overrightarrow{ID},i}) < \rho_M \cdot sk_{\overrightarrow{ID},i} \text{ Then} \\ & L[\overrightarrow{ID},i] + f(sk_{\overrightarrow{ID},i}) < \rho_M \cdot sk_{\overrightarrow{ID},i} \text{ Then} \\ & L[\overrightarrow{ID},i] \leftarrow L[\overrightarrow{ID},i] + f(sk_{\overrightarrow{ID},i}) \\ & \text{Return } f(sk_{\overrightarrow{ID},i}) \end{split}$	
$\begin{array}{c} \mathbb{C}[ID] \leftarrow 0\\ \text{Return } sk_{\overline{ID},C[\overline{ID}]} \end{array}$	Else Return \perp procedure Update (f, \overrightarrow{ID})	
$\frac{\textbf{procedure Challenge}(\overrightarrow{ID}^*)}{\text{Store }\overrightarrow{ID}^*}$	If $\overrightarrow{ID} \neq \overrightarrow{ID}^*$ and \overrightarrow{ID} not ancestor of \overrightarrow{ID}^* Then Return \perp	
$\frac{\text{procedure } \mathbf{LR}(m_0, m_1)}{C \stackrel{*}{\leftarrow} Enc(mpk, \overrightarrow{ID}^*, m_\beta)}$ Return C procedure Finalize(β')	Else continue let $i \leftarrow C[\overrightarrow{ID}]$ $sk_{\overrightarrow{ID},i+1} \stackrel{\hspace{0.1em}\circ}{\leftarrow} Update_{user}(mpk, sk_{\overrightarrow{ID},i}, r)$ $C[\overrightarrow{ID}] \leftarrow i+1$	
$\frac{\text{procedure rinalize}(\beta)}{\text{Return } (\beta' = \beta)}$	$\begin{split} &\text{If } L[\overrightarrow{ID},i] + f(sk_{\overrightarrow{ID},i},r) < \rho_U \cdot sk_{\overrightarrow{ID},i} \text{ Then} \\ &L[\overrightarrow{ID},i+1] \leftarrow f(sk_{\overrightarrow{ID},i},r) \\ &\text{Return } f(sk_{\overrightarrow{ID},i},r) \\ &\text{Else Return } \bot \end{split}$	

Figure 3: Definition of Game CML-IND-HID-CPA.

by Brakerski et al. [19].

In particular, we consider the model with the restriction that there is no leakage from the master secret key. This means that both the Setup and KeyDer algorithms do not leak secret information. In this setting a (H)IBE scheme is defined by the same algorithms as a standard (H)IBE with an additional Update_{user} algorithm that takes as input the public parameters, the secret key of some identity \overrightarrow{ID} and some randomness (from an appropriate domain), and it outputs a new updated secret key for the same identity \overrightarrow{ID} .

The notion of indistinguishability under chosen-plaintext attack in the CML model (that we call CML-IND-HID-CPA) is defined as follows.

The game consists of six procedures that can be run by an adversary \mathcal{A} and it works in the following way. As usual, \mathcal{A} starts by executing **Initialize** and runs **Finalize** before halting. The adversary can run the procedure **Extract** and then it is allowed one query to the procedure **Challenge** on some identity \overrightarrow{ID}^* such that \overrightarrow{ID}^* , nor an ancestor of it, have been asked to **Extract** before. Next, the adversary can run procedures **Extract**, **Leak** and **Update** as described in Figure 3. Notice that **Leak** and **Update** can be queried on identities \overrightarrow{ID} that decrypt \overrightarrow{ID}^* . These procedures take as input also a computable function f. As specified in the figure, such functions must have a sufficiently bounded output size. We also assume that \mathcal{A} makes at most one query (m_0, m_1) to the **LR** procedure, under the requirement that $|m_0| = |m_1|$ (i.e., the two messages have the same length), and that all identities submitted to **Extract**, **Leak**, **Update** and **LR** are *legitimate*. Finally, once the adversary has queried **LR** it can no longer run **Leak** and **Update**. In the CML model, a set of queries is said *legitimate* if \mathcal{A} never queries **Extract** on an identity \overrightarrow{ID} such that $\overrightarrow{ID} = \overrightarrow{ID}^*$ or \overrightarrow{ID} is an ancestor of \overrightarrow{ID}^* . Furthermore, the total number of bits of each secret key of \overrightarrow{ID}^* (or of any ancestor of \overrightarrow{ID}^*) that are leaked through **Leak** and **Update** must be less than $\rho_M \cdot |sk_{\overrightarrow{ID}^*}|$ and $\rho_U \cdot |sk_{\overrightarrow{ID}^*}|$ respectively. So, ρ_M and ρ_U represent the fraction of bits that can be leaked from the memory (i.e., from a secret key) and from the update operation (i.e., from the secret key and the randomness used in the update). Notice that (ρ_M, ρ_U) parametrize the security game.

We define the CML-IND-HID-CPA-advantage of any adversary \mathcal{A} against a HIBE scheme $\mathcal{H}I\mathcal{B}\mathcal{E}$ with leakage rate (ρ_M, ρ_U) as

$$\mathbf{Adv}_{\mathcal{HIBE}}^{\mathrm{CML-IND-HID-CPA}}(\mathcal{A}) = 2 \cdot \Pr[\mathrm{CML-IND-HID-CPA}^{\mathcal{A}} \Rightarrow 1] - 1$$

where CML-IND-HID-CPA^{\mathcal{A}} \Rightarrow 1 denotes that a run of the experiment CML-IND-HID-CPA (parametrized by (ρ_M, ρ_U)) with adversary \mathcal{A} outputs 1.

Definition 2.6 [CML-IND-HID-CPA-security] A HIBE scheme is CML-IND-HID-CPA-secure with leakage rate (ρ_M, ρ_U) if for any PPT adversary \mathcal{A} , $\mathbf{Adv}_{\mathcal{HIBE}}^{\mathrm{CML-IND-HID-CPA}}(\mathcal{A})$ is at most negligible.

In a very similar way it is possible to define the notion of selective security, CML-IND-sHID-CPA, for (H)IBE in the CML model. The game is described by the procedures in Figure 4. The procedures are similar to the ones of the CML-IND-HID-CPA game, but they are a bit simpler. For consistency, in order for the game to make sense, we require that the total number of bits of secret keys of \overrightarrow{ID}^* (or of any ancestor of \overrightarrow{ID}^*) that are leaked through **Leak** and **Update** must be less than $\rho_M \cdot |sk_{\overrightarrow{ID}^*}|$ and $\rho_U \cdot |sk_{\overrightarrow{ID}^*}|$ respectively.

Definition 2.7 [CML-IND-sHID-CPA-security] A HIBE scheme is CML-IND-sHID-CPA-secure with leakage rate (ρ_M, ρ_U) if for any PPT adversary \mathcal{A} , $\mathbf{Adv}_{\mathcal{HIBE}}^{\mathrm{CML-IND-sHID-CPA}}(\mathcal{A})$ is at most negligible.

WIBE in the CML model. Finally, we extend the security notion of WIBE to the CML model. To do this, we define the game CML-IND-sWID-CPA which is similar to IND-sWID-CPA, except that in addition it contains the procedures **Leak** and **Update**. The game is described in details in Figure 5. The main difference is in the definition of what is the set of *legitimate queries* in this setting. First, we require that the adversary calls the **LR** procedure on a pattern P that matches the pattern P^* provided to **Initialize** at the beginning of the game. Second, we require that **Leak** and **Update** are queried on identities matching the challenge pattern, and that for each of these identities the total number of leaked bits is at most $\rho_M \cdot |sk_{\overrightarrow{U}}|$ and $\rho_U \cdot |sk_{\overrightarrow{U}}|$ respectively.

Definition 2.8 [CML-IND-sWID-CPA-security] A WIBE scheme is CML-IND-sWID-CPA-secure with leakage rate (ρ_M, ρ_U) if for any PPT adversary \mathcal{A} , $\mathbf{Adv}_{\mathcal{WIBE}}^{\mathrm{CML-IND-sWID-CPA}}(\mathcal{A})$ is at most negligible.

3 Fully-Secure HIBE from Selective-Secure WIBE

In this section we concentrate on the first part of our main result. We show how to construct a fully-secure HIBE scheme starting from any WIBE scheme that is secure only in a selective sense.

Game CML-IND-sHID-CPA	
$\mathbf{procedure} \ \mathbf{Initialize}(\overrightarrow{ID}^*)$	procedure $\mathbf{Leak}(f)$
$(mpk, msk) \stackrel{*}{\leftarrow} Setup$	$ \overline{\text{If } L[i] + f(sk_{\overrightarrow{ID},i}) < \rho_M \cdot sk_{\overrightarrow{ID},i} \text{ Then} } $
$eta \stackrel{\$}{\leftarrow} \{0,1\}$	$L[i] \leftarrow L[i] + f(sk_{\overrightarrow{ID},i}) $
$i \leftarrow \bot$	Return $f(sk_{\overrightarrow{ID},i})$
$L[i] \leftarrow 0$	Else Return \perp
$sk_{\overrightarrow{ID}^*,0} \stackrel{\hspace{0.1em}\scriptscriptstyle\$}{\leftarrow} KeyDer(msk,\overrightarrow{ID}^*)$	procedure $Update(f)$
Return mpk	$sk_{\overrightarrow{ID},i+1} \stackrel{\$}{\leftarrow} Update_{user}(mpk, sk_{\overrightarrow{ID},i}, r)$
$\mathbf{procedure} \ \mathbf{Extract}(\overrightarrow{ID})$	If $L[i] + f(sk_{\overrightarrow{ID}i}, r) < \rho_U \cdot sk_{\overrightarrow{ID}i} $ Then
$sk_{\overrightarrow{ID}} \stackrel{\hspace{0.1em}\scriptscriptstyle\$}{\leftarrow} KeyDer(msk,\overrightarrow{ID})$	$L[i+1] \leftarrow f(sk_{\overrightarrow{ID},i},r) $
Return $sk_{\overrightarrow{ID}}$	Return $f(sk_{\overrightarrow{ID},i},r)$
procedure $LR(m_0, m_1)$	$i \leftarrow i+1$
$\frac{1}{C \stackrel{\$}{\leftarrow} Enc(mnk \ \overrightarrow{ID}^* \ m_2)}$	Else Return \perp
$\begin{array}{c} \begin{array}{c} \begin{array}{c} \\ \end{array} \\ \\ \\ \end{array} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ $	procedure $\mathbf{Finalize}(\beta')$
	$\boxed{\text{Return } (\beta' = \beta)}$

Figure 4: Definition of Game CML-IND-sHID-CPA.

Game CML-IND-sWID-CPA		
	procedure $\mathbf{Leak}(f, \overrightarrow{ID})$	
procedure Initialize (P^*)	$\boxed{\frac{1}{\text{let } i \leftarrow C[\overrightarrow{ID}]}}$	
$(mpk, msk) \stackrel{s}{\leftarrow} Setup$	If $i = \perp$ Then	
$eta \stackrel{\$}{\leftarrow} \{0,1\}$	$sk_{\overrightarrow{ID},0} \stackrel{s}{\leftarrow} KeyDer(msk,\overrightarrow{ID})$	
$C[\overrightarrow{ID}] \leftarrow \perp \forall \overrightarrow{ID}$	$C[\overrightarrow{ID}] \leftarrow 0$	
$L[\overrightarrow{ID},0] \leftarrow 0 \ \forall \overrightarrow{ID}$	If $L[\overrightarrow{ID}, i] + f(sk_{\overrightarrow{ID}}, i) < \rho_M \cdot sk_{\overrightarrow{ID}} $ Then	
Return mpk	$L[\overrightarrow{ID},i] \leftarrow L[\overrightarrow{ID},i] + f(sk_{\overrightarrow{ID},i}) $	
$\mathbf{procedure} \ \mathbf{Extract}(\overrightarrow{ID})$	Return $f(sk_{\overrightarrow{ID},i})$	
If $C[\overrightarrow{ID}] = \bot$ Then	Else Return \perp	
$sk_{\overrightarrow{ID},0} \stackrel{\hspace{0.1em}\scriptscriptstyle\$}{\leftarrow} KeyDer(msk,\overrightarrow{ID})$	procedure Update (f, \overrightarrow{ID})	
$C[\overrightarrow{ID}] \leftarrow 0$	$\boxed{\operatorname{let} i \leftarrow C[\overrightarrow{ID}]}$	
Return $sk_{\overrightarrow{ID},C[\overrightarrow{ID}]}$	$sk_{\overrightarrow{ID},i+1} \stackrel{\hspace{0.1em}\overset{\hspace{\{0}\hspace{1em}\overset{\hspace{01em}\overset{\hspace{01em}\overset{\hspace{01em}\overset{\hspace{01em}\overset{\hspace{01em}\overset{\hspace{01em}\overset{\hspace{01em}\overset{\hspace{01em}\overset{\hspace{01em}\overset{\hspace{01em}\overset{\hspace{01em}\overset{\hspace{01em}\overset{\hspace{1em}\overset{\hspace{1em}\overset{\hspace{1em}\overset{\hspace{01em}\overset{\hspace{1em}\overset{\scriptstyle{\hspace{1em}\overset{\hspace{1em}\overset{\hspace{1em}\overset{\hspace{1em}\overset{\hspace{1em}\overset{\hspace{1em}\overset{\scriptstyle{\hspace{1em}\overset{\hspace{1em}\overset{\scriptstyle{\hspace{1em}\overset{\hspace{1em}\overset{\scriptstyle}\overset{\scriptstyle{\hspace{1em}\atop\scriptstyle}\scriptstyle{\scriptstyle\\$	
procedure LR (P, m_0, m_1)	$C[\overrightarrow{ID}] \leftarrow i+1$	
$C \stackrel{\hspace{0.1em}\scriptscriptstyle\$}{\leftarrow} Enc(mpk,\overrightarrow{ID},m_{eta})$	If $L[\overrightarrow{ID}, i] + f(sk_{\overrightarrow{ID},i}, r) < \rho_U \cdot sk_{\overrightarrow{ID},i} $ Then	
Return C	$L[\overrightarrow{ID}, i+1] \leftarrow f(sk_{\overrightarrow{ID}}, r) $	
$\underline{\mathbf{procedure}\ \mathbf{Finalize}}(\beta')$	Return $f(sk_{\overrightarrow{ID}_{i}}, r)$	
Return $(\beta' = \beta)$	Else Return \perp	

Figure 5: Definition of Game CML-IND-sWID-CPA.

Our transformation is black-box and makes use of admissible hash functions, a notion introduced by Boneh and Boyen in [9] that we recall below.

Admissible Hash Functions. Admissible hash functions were first introduced by Boneh and Boyen in [9] as a tool for proving the full security of their identity-based encryption scheme in

the standard model. Such functions turn out to be particularly suitable for this purpose as they provide a way to implement the so-called "partitioning technique", a proof methodology that allows to secretly partition the identity space into two sets, the *blue* set and the *red* set, both of exponential size, so that there is a non-negligible probability that the adversary's secret key queries fall in the blue set and the challenge identity falls in the red set. This property has been shown useful to prove the full security of some identity-based encryption schemes (e.g., [9, 35, 22]). In particular, it fits those cases when, in the reduction, one can program the simulator so that it can answer secret key queries for all the blue identities, whereas it is prepared to generate a challenge ciphertext only for red identities.

In our work we employ admissible hash functions for a similar purpose, i.e., constructing a fullysecure HIBE from a selective-secure WIBE, and in particular we adopt a definition of admissible hash functions which follows the one used by Cash *et al.* in [22]. The formal definition follows.

Let $k \in \mathbb{N}$ be the security parameter, w and λ be two values that are at most polynomial in k, and Σ be an alphabet of size s. Let $\mathcal{H} = \{H : \{0,1\}^w \to \Sigma^\lambda\}$ be a family of functions. For $H \in \mathcal{H}$, $K \in (\Sigma \cup \{*\})^\lambda$ and any $x \in \{0,1\}^w$ we define the following function which colors strings in $\{0,1\}^w$ as follows:

$$F_{K,H}(x) = \begin{cases} \mathbf{R} & \text{if } \forall i \in \{1, \dots, \lambda\} : H(x)_i = K_i \text{ or } K_i = * \\ \mathbf{B} & \text{if } \exists i \in \{1, \dots, \lambda\} : H(x)_i \neq K_i \end{cases}$$

For any $\mu \in \{0, ..., \lambda\}$, we denote with $\mathcal{K}^{(\lambda,\mu)}$ the uniform distribution over $(\Sigma \cup \{*\})^{\lambda}$ such that exactly μ components are not *. Moreover, for every $H \in \mathcal{H}$, $K \in \mathcal{K}^{(\lambda,\mu)}$, and every vector $\vec{x} \in (\{0,1\}^w)^{Q+1}$ we define the function

$$\gamma(\vec{x}) = \Pr[F_{K,H}(x_0) = \mathbb{R} \land F_{K,H}(x_1) = \mathbb{B} \land F_{K,H}(x_2) = \mathbb{B} \land \dots \land F_{K,H}(x_Q) = \mathbb{B}].$$

Definition 3.1 [Admissible Hash Functions] $\mathcal{H} = \{H : \{0,1\}^w \to \Sigma^\lambda\}$ is a family of (Q, δ_{min}) admissible hash functions if for every polynomial Q = Q(k), there exists an efficiently computable function $\mu = \mu(k)$, efficiently recognizable sets $bad_H \subseteq (\{0,1\}^w)^*$ and an inverse of a polynomial $\delta_{min} = 1/\delta(k, Q)$ such that the following properties holds:

1. For every PPT algorithm \mathcal{A} that, on input $H \in \mathcal{H}$, outputs $\vec{x} \in (\{0,1\}^w)^{Q+1}$, there exists a negligible function $\epsilon(k)$ such that:

$$\mathbf{Adv}_{\mathcal{H}}^{adm}(\mathcal{A}) = \Pr[\vec{x} \in bad_H : H \leftarrow \mathcal{H}, \vec{x} \leftarrow \mathcal{A}(H)] \le \epsilon(k)$$

2. For every $H \in \mathcal{H}, K \stackrel{*}{\leftarrow} \mathcal{K}^{(\lambda,\mu)}$, and every vector $\vec{x} \in (\{0,1\}^w)^{Q+1} \setminus bad_H$ such that $x_0 \notin \{x_1,\ldots,x_Q\}$ we have: $\gamma(\vec{x}) \geq \delta_{min}$.

3.1 Our transformation

Let $\mathcal{W}I\mathcal{BE}$ be a WIBE scheme with identity space $I\mathcal{D} = \Sigma$ of size s and depth $\leq \lambda \cdot L$, and $\mathcal{H} = \{H : \{0,1\}^w \to \Sigma^\lambda\}$ be a family of functions. Then we construct the following HIBE scheme that has identity space $I\mathcal{D}' = \{0,1\}^w$ and depth at most L:

- $\mathcal{H}IB\mathcal{E}.\mathsf{Setup:} \operatorname{run}(mpk', msk') \stackrel{\hspace{0.1em}\mathsf{\leftarrow}}{\leftarrow} \mathcal{W}IB\mathcal{E}.\mathsf{Setup} \text{ and select } H_1, \ldots, H_L \stackrel{\hspace{0.1em}\mathsf{\leftarrow}}{\leftarrow} \mathcal{H}. \text{ Output } mpk = (mpk', H_1, \ldots, H_L) \text{ and } msk = msk'.$
- $\mathcal{H}I\mathcal{B}\mathcal{E}.\mathsf{KeyDer}(msk, \overrightarrow{ID}): \text{ let } \overrightarrow{ID} = (ID_1, \dots, ID_\ell) \text{ and define } \vec{I} = (H_1(ID_1), \dots, H_\ell(ID_\ell)) \in \Sigma^{\lambda \cdot \ell}.$ Output $sk_{\overrightarrow{ID}} = \mathcal{W}I\mathcal{B}\mathcal{E}.\mathsf{KeyDer}(msk, \vec{I}).$

 $\mathcal{H}I\mathcal{B}\mathcal{E}.\mathsf{Enc}(mpk, \overrightarrow{ID}, \mathfrak{m})$: let $\overrightarrow{ID} = (ID_1, \ldots, ID_\ell)$ and define $\overrightarrow{I} = (H_1(ID_1), \ldots, H_\ell(ID_\ell)) \in \Sigma^{\lambda \cdot \ell}$. Output $C = \mathcal{W}I\mathcal{B}\mathcal{E}.\mathsf{Enc}(mpk, \overrightarrow{I}, m)$.

 $\mathcal{H}I\mathcal{B}\mathcal{E}.\mathsf{Dec}(sk_{\overrightarrow{ID}},C): \text{ return } m = \mathcal{W}I\mathcal{B}\mathcal{E}.\mathsf{Dec}(sk_{\overrightarrow{ID}},C).$

Our scheme is very simple. Essentially, the HIBE algorithm uses the algorithms of the WIBE scheme in a black-box way, where each identity component ID_i is first hashed using a function $H_i \in \mathcal{H}$. Boneh and Boyen show how to construct admissible hash functions based on collision-resistance and error-correction, and propose some concrete parameters for their instantiation (which satisfy our definition). In particular, for convenience of their construction, they consider functions that map to strings in an alphabet Σ of size s = 2. Here we notice that if the given WIBE has an alphabet Σ' of size s' > 2, then one can simply choose two values $x_1, x_2 \in \Sigma'$, set $\Sigma = \{x_1, x_2\}$, and then consider the same WIBE restricted to these two identities.

The security of our scheme follows from the following theorem.

Theorem 3.2 If $\mathcal{H} = \{H : \{0, 1\}^w \to \Sigma^\lambda\}$ is a family of (Q, δ_{min}) -admissible hash functions, and \mathcal{WIBE} is IND-sWID-CPA-secure, then the scheme \mathcal{HIBE} given in Section 3 is IND-HID-CPA-secure, where the maximum hierarchy's depth L is some fixed constant.

PROOF INTUITION. Although the scheme is simple, its proof of security is rather technical. Therefore, we first provide some informal intuitions about our strategy. Intuitively speaking, the proof proceeds by showing an algorithm \mathcal{B} that plays game IND-sWID-CPA against the scheme $\mathcal{W}I\mathcal{BE}$ and simulates the game IND-HID-CPA to an adversary \mathcal{A} against $\mathcal{H}I\mathcal{BE}$. \mathcal{B} first generates the parameters for the admissible hash functions, which define partitions B and R of the identity space, and then it declares the set R as the challenge pattern (notice that by definition of $K \in \mathcal{K}^{(\lambda,\mu)}$, R can be described in a compact way using a pattern). Next, all secret key queries made by \mathcal{A} for identities in B are forwarded by \mathcal{B} to its own challenger, and the same can be done if the challenge identity chosen by \mathcal{A} falls in R. In particular, by the properties of admissible hash functions, the event that the identities of secret key queries fall in B and the challenge identity falls in R occurs with non-negligible probability. However, things are not that simple, as there may be unlucky events in which \mathcal{B} is unable to simulate the right game to \mathcal{A} and thus it needs to abort. As it already occurred in other works [35, 22], these events may not be independent of the adversary's view, and one solution is to force the simulator to run an expensive artificial abort step. Our proof of Theorem 3.2 proceeds in this way, requiring \mathcal{B} to (eventually) artificially abort at the end of the simulation.

Alternatively, one can extend the techniques introduced by Bellare and Ristenpart in [5] to obtain a proof of Theorem 3.2 which avoids the need of artificial aborts. However, this requires a slightly different definition of admissible hash functions. In Appendix A we describe this alternative proof without artificial aborts. It may be of independent interest.

Proof: To prove Theorem 3.2 we describe a sequence of games that allows to show that an adversary for the game IND-HID-CPA can be efficiently turned into an adversary for the game IND-sWID-CPA.

THE SIMULATOR ALGORITHM \mathcal{B} . In Figure 6 we describe an adversary \mathcal{B} that plays game IND-sWID-CPA against the scheme $\mathcal{W}I\mathcal{BE}$, by simulating the game IND-HID-CPA to an adversary \mathcal{A} . To avoid confusion between the games IND-sWID-CPA and IND-HID-CPA, we prepend the prefix **sW** to the procedures of IND-sWID-CPA.

In order to show that such simulation can be carried on efficiently, we proceed by describing a sequence of games G_0-G_8 , where G_0 is the game simulated by our algorithm \mathcal{B} , and G_8 is essentially

Algorithm \mathcal{B} : $K_1,\ldots,K_L \stackrel{\star}{\leftarrow} \mathcal{K}^{(\lambda,\mu)}$ $P^* \leftarrow (K_1, \dots, K_L)$ Run $mpk' \leftarrow \mathbf{sW.Initialize}(P^*)$ $H_1,\ldots,H_L \stackrel{s}{\leftarrow} \mathcal{H}$ $cnt \leftarrow 1$ $mpk \leftarrow (mpk', H_1, \ldots, H_L)$ Run $\mathcal{A}'(mpk)$, answering queries as follows: $\mathbf{Extract}(\overrightarrow{ID})$: $X^{cnt} \leftarrow \overrightarrow{ID}, cnt \leftarrow cnt + 1$ let $\ell = |\vec{ID}|, \vec{I} \leftarrow (H_1(ID_1), \dots, H_\ell(ID_\ell))$ $sk_{\overrightarrow{ID}} \leftarrow \bot$ If $F_{H_i,K_i}(ID_i) = \mathbb{R} \ \forall i = 1 \ \text{to} \ \ell \ \text{Then}$ $bad \leftarrow true$ Else $sk_{\vec{I}\vec{D}} \stackrel{s}{\leftarrow} \mathbf{sW.Extract}(\vec{I})$ Return $sk_{\overrightarrow{ID}}$ $LR(ID, m_0, m_1):$ $X^0 \leftarrow \overrightarrow{ID}$ let $\ell^* = |\overrightarrow{ID}|, \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_{\ell^*}(ID_{\ell^*}))$ $C^* \leftarrow \bot$ If $\exists i \in [\ell^*] : F_{H_i, K_i}(ID_i) = B$ Then $bad \leftarrow true$ Else $C^* \xleftarrow{\hspace{0.1em} \$} \mathbf{sW.LR}(\vec{I}, m_0, m_1)$ return C^* let β' be \mathcal{A}' 's output If $\exists i \in [L] : X_i \in bad_{H_i}$ Then $\beta' \stackrel{s}{\leftarrow} \{0, 1\}$ If bad \neq true $\tilde{\eta} \leftarrow 0$ for j = 1 to $\lfloor kS/\delta_{min}^L \rfloor$ do $\tilde{K}_1,\ldots,\tilde{K}_L \stackrel{\$}{\leftarrow} \mathcal{K}^{(\lambda,\mu)}$ If $\bigwedge_{i=1}^{L} (F_{\tilde{K}_i,H_i}(X_i^0) = \mathbf{R} \land F_{\tilde{K}_i,H_i}(X_i^1) = \mathbf{B} \land$ $\wedge \cdots \wedge F_{\tilde{K}_i,H_i}(X_i^Q) = \mathsf{B})$ Then $\tilde{\eta} \leftarrow \tilde{\eta} + 1$ $\tilde{\delta} \leftarrow \tilde{\eta} / \lceil kS / \delta_{min}^L \rceil$ Set bad \leftarrow true with probability $1 - \delta_{min}^L / \tilde{\delta}$ If bad = true Then $\beta' \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} \{0,1\}$ $sW.Finalize(\beta')$

procedure Initialize: Games $G_0 - G_3$ 001 $K_1, \ldots, K_L \stackrel{\$}{\leftarrow} \mathcal{K}^{(\lambda,\mu)}$ $002 \ P^* \leftarrow (K_1, \ldots, K_L)$ 003 $(mpk', msk') \stackrel{\$}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{Setup}; \beta \stackrel{\$}{\leftarrow} \{0, 1\}$ 004 $H_1, \ldots, H_L \stackrel{s}{\leftarrow} \mathcal{H}$ $005 \ cnt \leftarrow 1$ 006 $mpk \leftarrow (mpk', H_1, \ldots, H_L)$ 007 return mpkprocedure Extract(ID): Games G₀, G₁ 010 $X^{cnt} \leftarrow \overrightarrow{ID}; cnt \leftarrow cnt + 1$ 011 let $\ell = |\overrightarrow{ID}|, \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_\ell(ID_\ell))$ 012 $sk_{\overrightarrow{ID}} \leftarrow \bot$ 013 If $F_{H_i,K_i}(ID_i) = \mathbb{R} \ \forall i = 1 \text{ to } \ell \text{ Then}$ 014 $bad \leftarrow true$ 015 $sk_{\overrightarrow{ID}} \stackrel{\$}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{KeyDer}(msk', \vec{I})$ 016 Else $sk_{\overrightarrow{ID}} \stackrel{\$}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{KeyDer}(msk', \vec{I})$ 017 Return $sk_{\overrightarrow{ID}}$ procedure LR(ID, m_0, m_1): Games G₀, G₁ 020 $X^0 \leftarrow \overrightarrow{ID}$ 021 let $\ell^* = |\vec{DD}|, \vec{I} \leftarrow (H_1(ID_1), \dots, H_{\ell^*}(ID_{\ell^*}))$ 022 $C^* \leftarrow \bot$ 023 If $\exists i \in [\ell^*] : F_{H_i,K_i}(ID_i) = B$ Then 024 $bad \leftarrow true$ $C^* \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} \mathcal{W}I\mathcal{B}\mathcal{E}.\mathsf{Enc}(mpk', \vec{I}, m_\beta)$ 025026 Else 027 $C^* \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} \mathcal{W}I\mathcal{B}\mathcal{E}.\mathsf{Enc}(mpk', \vec{I}, m_\beta)$ 028 return C^* **procedure Finalize**(β'): Games G₀, |G₁ 030 If $\exists i \in [L] : X_i \in bad_{H_i}$ Then $\beta' \stackrel{*}{\leftarrow} \{0, 1\}$ 031 $\beta'' \leftarrow \beta'$ 032 If bad \neq true $\tilde{\eta} \leftarrow 0$ 033for j = 1 to $\lceil kS/\delta_{min}^L \rceil$ do 034 $\tilde{K}_1, \ldots, \tilde{K}_L \xleftarrow{\hspace{0.1cm}\$} \mathcal{K}^{(\lambda,\mu)}$ 035If $\bigwedge_{i=1}^{L} (F_{\tilde{K}_i,H_i}(X_i^0) = \mathbf{R} \wedge F_{\tilde{K}_i,H_i}(X_i^1) = \mathbf{B}$ 036 $\wedge \cdots \wedge F_{\tilde{K}_i, H_i}(X_i^Q) = \mathsf{B}$) Then $\tilde{\eta} \leftarrow \tilde{\eta} + 1$ 037 $\tilde{\delta} \leftarrow \tilde{\eta} / [kS / \delta_{min}^L]$ 038Set bad \leftarrow true with probability $1 - \delta_{min}^L / \tilde{\delta}$ 039040 If bad = true Then $\beta'' \stackrel{\$}{\leftarrow} \{0,1\}, \beta'' \leftarrow \beta'$ 041 If $\beta'' = \beta$ Then return 1 042 Else return 0

Figure 6: Adversary \mathcal{B} and description of the games G_0 and G_1 .

IND-HID-CPA with some additional code that, however, does not condition the output. Our approach is based on code-based games where each game is defined as a set of procedures that can be run by the adversary.

Before focusing on the game sequence, we first show that the simulation provided by \mathcal{B} is correct whenever **bad** is not set, and that \mathcal{B} plays the game IND-sWID-CPA correctly. For ease of exposition we assume that the adversary always outputs identities of the same (maximum) length L. However, this can be formalized by assuming that for any set of identities $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^Q)$ output by \mathcal{A} , for i = 1 to Q all those \overrightarrow{ID}^i such that $|\overrightarrow{ID}^i| < L$ are padded to reach length L using some special symbol so that $F_{H_i,K_i}(ID_j^i)$ always returns B on positions j such that $|\overrightarrow{ID}^i| < j \leq L$. On the other hand, if the challenge identity has length $\ell^* < L$, then it is padded with some symbol so that $F_{H_i,K_i}(ID_j^0)$ always returns R on positions $j > \ell^*$.

First, observe that all the identities \vec{I} for which \mathcal{B} runs $\mathbf{sW.Extract}(\vec{I})$ are legitimate queries, namely they do not match the challenge pattern P^* declared by \mathcal{B} to $\mathbf{sW.Initialize}$. In the code of \mathcal{B} , if $\mathbf{sW.Extract}(\vec{I})$ is called, then there exists an index $i \in \{1, \ldots, \ell\}$ for which $F_{K_i, H_i}(ID_i) = B$, namely $I_i \neq P_i$ (and $P_i \neq *$), thus $\vec{I} \notin_* P^*$. Second, note that the ciphertext C^* is distributed as the challenge ciphertext in the game IND-HID-CPA for the scheme \mathcal{HIBE} . However, we have also to check that the procedure $\mathbf{sW.LR}$ be run on an identity $\vec{I} \in_* P^*$. To see this, observe that the procedure is run only if bad is not set, namely when $F_{H_i,K_i}(ID_i) = \mathbb{R}$ for all $i \in [\ell^*]$, which is equivalent to say $\vec{I} \in_* P^*$.

A critical part in \mathcal{B} 's simulation is that it may set bad \leftarrow true and, as a consequence, \mathcal{B} returns a random bit (basically, it fails its simulation). Such bad event depends on the values K_1, \ldots, K_L chosen by \mathcal{B} as well as on the set of identities asked by \mathcal{A} to **Extract** and **LR**. As shown in other works, such as [35], these cases are problematic as the event that the simulation fails is not independent of the adversary's view. This difficulty is overcome by introducing an "artificial" abort event in the simulation that allows to balance the probability of failing so that it is sufficiently independent of the adversary's view. This is why, at the end of the simulation, even if bad was not set, the algorithm \mathcal{B} may abort. Precisely, the simulator \mathcal{B} proceeds as follows. Before terminating the simulation, \mathcal{B} repeats $\lceil kS/\delta_{min}^L \rceil$ times the following step: it samples L vectors $\tilde{K}_1, \ldots, \tilde{K}_L$ as at the beginning of the simulation, and for each sample it checks whether such choice (combined with the given set X of identities returned by the adversary) would set bad \leftarrow true or not. At the end of this step, \mathcal{B} evaluates "on the fly" the average probability, over the random choices of the vectors \tilde{K}_i , that bad is set, given the set X. Let $\tilde{\delta}$ be such estimation, then \mathcal{B} sets bad \leftarrow true with probability $1 - \delta_{min}^L/\tilde{\delta}$. In particular, here S is an arbitrary polynomial such that by Hoeffding's inequality, $\lceil kS/\delta^L \rceil$ samples are sufficient to get $\tilde{\delta} \geq \delta^L$ such that

$$\Pr\left[\left|\Gamma(X) - \tilde{\delta}\right| \ge \frac{\delta^L}{S}\right] \le \frac{1}{2^k}.$$
(1)

THE SEQUENCE OF GAMES. Now, let us focus on the sequence of games $G_0 - G_8$. In particular, the Lemma 3.3 given below proves that we can move from the game IND-sWID-CPA played by \mathcal{B} to game G_4 .

Following the notation given in Section 2, we write $G_i^{\mathcal{A}} \Rightarrow b$ to denote that an execution of game G_i by \mathcal{A} returns b. Also, let Bad_i (resp. Good_i) be the event that G_i sets (resp. does not set) $\mathsf{bad} \leftarrow \mathsf{true}$.

Our adversary \mathcal{B} and the games G_0 - G_8 are described in Figures 6 and 7. When some games share a procedure with very similar code we use a compact description with boxed statements. If a

procedure Extract(ID): Game G₂ 210 $X^{cnt} \leftarrow \overrightarrow{ID}; cnt \leftarrow cnt + 1$ 211 let $\ell = |\overrightarrow{ID}|, \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_\ell(ID_\ell))$ 212 If $F_{H_i,K_i}(ID_i) = \mathbb{R} \ \forall i = 1 \ \text{to} \ \ell \ \text{Then}$ $\mathsf{bad} \gets \mathsf{true}$ 213214 $sk_{\overrightarrow{ID}} \stackrel{*}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{KeyDer}(msk', \vec{I})(\vec{I})$ 215 Return $sk_{\overrightarrow{ID}}$ procedure $LR(ID, m_0, m_1)$: Game G₂ 220 $X^0 \leftarrow \overrightarrow{ID}$ 221 let $\ell^* = |\overrightarrow{ID}|, \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_{\ell^*}(ID_{\ell^*}))$ 222 If $\exists i \in [\ell^*] : F_{H_i,K_i}(ID_i) = B$ Then 223 bad \leftarrow true 224 $C' \stackrel{\$}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{Enc}(mpk', \vec{I}, m_{\beta})$ 225 return C^* procedure Finalize(β'): Game G₂, G₃ 230 If $\exists i \in [L] : X_i \in bad_{H_i}$ Then $\beta' \stackrel{\$}{\leftarrow} \{0, 1\}$ 231 for j = 1 to cnt do let $\ell_j \leftarrow |ID^j|$ 232If $F_{H_i,K_i}(X_i^j) = \mathbb{R} \ \forall i = 1 \text{ to } \ell_j \text{ Then}$ 233234 $\mathsf{bad} \gets \mathsf{true}$ 235 If $\exists i \in [\ell^*] : F_{H_i, K_i}(X_i^0) = B$ Then $\mathsf{bad} \leftarrow \mathsf{true}$ 236237 If bad \neq true 238 $\tilde{\eta} \leftarrow 0$ for j = 1 to $\lfloor kS/\delta_{min}^L \rfloor$ do 239 $\tilde{K}_1,\ldots,\tilde{K}_L \stackrel{\$}{\leftarrow} \mathcal{K}^{(\lambda,\mu)}$ 240If $\bigwedge_{i=1}^{L} (F_{\tilde{K}_i,H_i}(X_i^0) = \mathbf{R} \wedge F_{\tilde{K}_i,H_i}(X_i^1) = \mathbf{B}$ $\wedge \cdots \wedge F_{\tilde{K}_i,H_i}(X_i^Q) = \mathbf{B}$) Then 241 $\tilde{\eta} \leftarrow \tilde{\eta} + 1$ 242 $\tilde{\delta} \leftarrow \tilde{\eta} / [kS / \delta_{min}^L]$ 243Set bad \leftarrow true with probability $1 - \delta_{min}^L / \tilde{\delta}$ 244245 If $\beta' = \beta$ Then return 1 246 Else return 0 procedure Initialize(ℓ^*): Games $G_4 - G_8$ $\overline{400} \ (mpk', msk') \stackrel{\hspace{0.1em}\hspace{0.1em}\hspace{0.1em}}\leftarrow \quad \mathcal{W}\!I\!\mathcal{B}\!\mathcal{E}.\mathsf{Setup}; \beta \stackrel{\hspace{0.1em}\hspace{0.1em}\hspace{0.1em}\hspace{0.1em}}\leftarrow \quad \{0,1\}$ 401 $H_1, \ldots, H_L \stackrel{\$}{\leftarrow} \mathcal{H}$ 402 $mpk \leftarrow (mpk', H_1, \ldots, H_L)$ 403 $cnt \leftarrow 1$ 404 return mpk procedure Finalize(β'): Game G₆ 640 If $\exists i \in [L] : X_i \in bad_{H_i}$ Then $\beta' \stackrel{*}{\leftarrow} \{0, 1\}$ 641 Set bad \leftarrow true with probability $1 - \delta_{min}^L$ 642 If bad = true Then $\beta' \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} \{0,1\}$ 643 If $\beta' = \beta$ Then return 1 644 Else return 0

procedure Extract(\overrightarrow{ID}): Games $G_3 - G_8$ $310 \quad X^{cnt} \leftarrow \overrightarrow{ID}$; $cnt \leftarrow cnt + 1$ $311 \quad let \ \ell = |\overrightarrow{ID}|, \ \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_\ell(ID_\ell))$ $312 \quad sk_{\overrightarrow{ID}} \stackrel{*}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{KeyDer}(msk', \overrightarrow{I})(\overrightarrow{I})$ $313 \quad \text{Return } sk_{\overrightarrow{ID}}$ **procedure LR**($\overrightarrow{ID}, m_0, m_1$): Game $G_3 - G_8$ $320 \quad X^0 \leftarrow \overrightarrow{ID}$ $321 \quad let \ \ell^* = |\overrightarrow{ID}|, \ \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_{\ell^*}(ID_{\ell^*}))$ $322 \quad C^* \stackrel{*}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{Enc}(mpk', \ \overrightarrow{I}, m_\beta)$ $323 \quad return \ C^*$

procedure Finalize(β'): Game G₄, G₅ 430 If $\exists i \in [L] : X_i \in bad_{H_i}$ Then $\beta' \stackrel{s}{\leftarrow} \{0, 1\}$ 431 $K_1, \ldots, K_L \stackrel{\$}{\leftarrow} \mathcal{K}^{(\lambda,\mu)}$ 432 for j = 1 to cnt do 433 let $\ell_i \leftarrow |ID^j|$ If $F_{H_i,K_i}(X_i^j) = \mathbb{R} \ \forall i = 1 \ \text{to} \ \ell_j$ Then 434 $\mathsf{bad} \gets \mathsf{true}$ 435 If $\exists i \in [\ell^*] : F_{H_i,K_i}(X_i^0) = B$ Then 436 437 $bad \leftarrow true$ 438 If bad \neq true 439 $\tilde{\eta} \leftarrow 0$ for j = 1 to $\lfloor kS/\delta_{min}^L \rfloor$ do 440
$$\begin{split} \tilde{K}_{1}, \dots, \tilde{K}_{L} &\stackrel{\text{s}}{\leftarrow} \mathcal{K}^{(\lambda, \mu)} \\ \text{If } \bigwedge_{i=1}^{L} (F_{\tilde{K}_{i}, H_{i}}(X_{i}^{0}) = \mathbf{R} \wedge F_{\tilde{K}_{i}, H_{i}}(X_{i}^{1}) = \mathbf{B} \\ & \wedge \dots \wedge F_{\tilde{K}_{i}, H_{i}}(X_{i}^{Q}) = \mathbf{B}) \text{ Then} \end{split}$$
441 442 443 $\tilde{\eta} \leftarrow \tilde{\eta} + 1$ $\tilde{\delta} \leftarrow \tilde{\eta} / \lceil kS / \delta_{min}^L \rceil$ 444Set bad \leftarrow true with probability $1 - \delta_{min}^L / \tilde{\delta}$ 445If bad = true Then $\beta' \stackrel{s}{\leftarrow} \{0, 1\}$ 446 447 If $\beta' = \beta$ Then return 1 448 Else return 0

procedure Finalize(β'): Game G₇ 740 If $\exists i \in [L] : X_i \in bad_{H_i}$ Then $\beta' \stackrel{\$}{\leftarrow} \{0, 1\}$ 741 If $\beta' = \beta$ Then return 1 742 Else return 0

procedure Finalize (β') : Game G₈ 840 If $\beta' = \beta$ Then return 1 841 Else return 0

Figure 7: Description of the Games from G_2 to G_8 .

procedure is shared by games G_i, G_j, \ldots, G_k , if G_i is boxed, then the code of the given procedure in G_i includes the boxed statements, whereas its code in the other games does not. To better understand the notation one may look at Figure 6 for an example. There, the **Finalize** procedure is shared by games G_0 and G_1 , and G_1 is written in a box. This means that **Finalize** in G_1 contains the statement $\beta'' \leftarrow \beta'$ of line 040, whereas this statement is not present in game G_0 .

 $\textbf{Lemma 3.3 Adv}^{\text{IND-sWID-CPA}}_{\mathcal{WIBE}}(\mathcal{B}) = 2 \cdot \Pr[G_4^{\mathcal{A}} \Rightarrow 1 \wedge \mathsf{Good}_4] - \Pr[\mathsf{Good}_4].$

Proof: To prove the lemma we will analyze the differences between each consecutive pair of games.

First, we focus on the code of \mathcal{B} and game G₀. The procedure **Initialize** contains in line 003 the code of **sW.Initialize**. Moreover, line 016 and line 027 contain the code of **sW.Extract** and **sW.LR** respectively. Finally, it is not hard to notice that the code of the **Finalize** procedure is an equivalent implementation of the way \mathcal{B} concludes its simulation and executes **sW.Finalize**. Therefore, we have:

$$Pr[IND-sWID-CPA^{\mathcal{B}} \Rightarrow 1] = Pr[G_0^{\mathcal{A}} \Rightarrow 1]$$

$$= Pr[G_0^{\mathcal{A}} \Rightarrow 1 | \mathsf{Bad}_0] Pr[\mathsf{Bad}_0] + Pr[G_0^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_0]$$

$$= \frac{1}{2} Pr[\mathsf{Bad}_0] + Pr[G_0^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_0]$$
(2)

where Equation (2) is justified from that the **Finalize** procedure of G_0 outputs a random bit when bad is set.

If we look at the differences between the games G_0 and G_1 we can observe that G_1 contains some additional lines of code (highlighted in the framed boxes). Such changes make sure that **Extract** and **LR** never return \perp . Also, in G_1 **Finalize** is modified in line 040 (by adding $\beta'' \leftarrow \beta'$) so that the procedure's output does not depend on **bad** = *true*. Since in game G_0 the events that **Extract** and **LR** return \perp and that **Finalize** takes β'' at random both occur only if **bad** is set, then we have that G_0 and G_1 are identical-until-bad. Thus, we can apply Lemma 2.1 to obtain:

$$\Pr[\mathsf{Bad}_0] = \Pr[\mathsf{Bad}_1] \quad \text{and} \quad \Pr[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_0] = \Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_1] \tag{3}$$

Now, let us compare games G_1 and G_2 . The changes in the **Extract** and **Finalize** procedures are only syntactical. Lines 015,016 (resp. 025,027) of G_1 have been moved to line 214 (resp. 224) of G_2 . So G_2 is equivalent to G_1 :

$$\Pr[\mathsf{Bad}_1] = \Pr[\mathsf{Bad}_2] \quad \text{and} \quad \Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_1] = \Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_2] \tag{4}$$

Let us now consider G_2 and G_3 . In game G_2 , both **Extract** and **LR** may set **bad** in lines 212 - 213 and 222 - 223 respectively. However, this operation does no longer influence the behavior of each procedures. So, in G_3 these lines are moved to the end of the game, into the procedure **Finalize**. Moreover, in order for this change to be described correctly, G_3 introduces a counter and a labeling for the queried identities. Again, these changes in the code are only syntactical. Thus the two games are identical, and we have:

$$\Pr[\mathsf{Bad}_2] = \Pr[\mathsf{Bad}_3] \quad \text{and} \quad \Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_2] = \Pr[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_3] \tag{5}$$

Finally, we show that G_3 and G_4 are identically distributed as well. The only change is that line 001 of G_3 is moved to line 431 of **Finalize** in G_4 . Since in G_3 the values K_1, \ldots, K_L are used only into **Finalize**, this code can be postponed there. Thus we have:

$$\Pr[\mathsf{Bad}_3] = \Pr[\mathsf{Bad}_4] \quad \text{and} \quad \Pr[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_3] = \Pr[\mathsf{G}_4^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_4] \tag{6}$$

Finally, if we put together Equations (2), (3), (4), (5) and (6) we obtain:

$$\mathbf{Adv}_{\mathcal{W}I\mathcal{B}\mathcal{E}}^{\mathrm{IND-sWID-CPA}}(\mathcal{B}) = 2 \cdot \Pr[\mathrm{IND-sWID-CPA}^{\mathcal{B}} \Rightarrow 1] - 1$$
$$= \Pr[\mathsf{Bad}_4] + 2 \cdot \Pr[\mathrm{G}_4^{\mathcal{A}} \wedge \mathsf{Good}_4] - 1$$
$$= 2 \cdot \Pr[\mathrm{G}_4^{\mathcal{A}} \wedge \mathsf{Good}_4] - \Pr[\mathsf{Good}_4]$$
(7)

which completes the proof of the Lemma.

Next, if we look at games G_4 and G_5 , we notice that the only difference is that G_5 changes the value of β' with a random bit when **bad** = **true**. Since this action is performed only if **bad** is set, then we have that games G_4 and G_5 are identical-until-**bad**, and thus we can apply the restatement of the fundamental Lemma of game-playing (i.e., Lemma 2.1) to obtain:

$$\Pr[\mathsf{Bad}_4] = \Pr[\mathsf{Bad}_5] \quad \text{and} \quad \Pr[\mathsf{G}_4^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_4] = \Pr[\mathsf{G}_5^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_5] \tag{8}$$

Now, let us focus on the games G_5 and G_6 . We observe that lines 431-445 of game G_5 are substituted with line 641 in game G_6 . In particular, in the latter game **bad** is set **true** with independent probability $1 - \delta_{min}^L$. Since $\Pr[\text{Good}_5] = \delta_{min}^L \cdot \frac{\Gamma(X)}{\delta}$, and the condition of Equation (1) holds, then we obtain that the difference

$$|\Pr[\mathsf{Good}_5] - \Pr[\mathsf{Good}_6]| = \delta_{\min}^L \cdot \frac{\tilde{\delta} - \Gamma(X)}{\tilde{\delta}} \le \frac{\delta_{\min}^L}{S}$$

holds with probability $1 - 1/2^k$. Thus we have:

$$\left|\Pr[\mathbf{G}_{5}^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathbf{G}_{6}^{\mathcal{A}} \Rightarrow 1]\right| \le \frac{\delta_{min}^{L}}{S} + \frac{1}{2^{k}} \tag{9}$$

Game G_7 is the same as G_6 except that the **Finalize** procedure does not set bad. So we have:

$$2 \cdot \Pr[\mathbf{G}_6^{\mathcal{A}} \Rightarrow 1] - 1 = \delta_{\min}^L (2 \cdot \Pr[\mathbf{G}_7^{\mathcal{A}} \Rightarrow 1] - 1)$$
(10)

Finally, observe that game G_8 differs from game G_7 as it does no longer contain line 740. So, it is easy to observe that a trivial reduction would show that any efficient distinguisher between the two games would reduce to the first condition of admissible hash functions, namely:

$$|\Pr[\mathbf{G}_{8}^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathbf{G}_{7}^{\mathcal{A}} \Rightarrow 1]| \le L \cdot \mathbf{Adv}_{\mathcal{H},\mathcal{C}}^{adm}(k)$$
(11)

Finally, one can easily note that game G_8 is essentially the same as the game IND-HID-CPA with some additional book-keeping. So we can write:

$$\mathbf{Adv}_{HIBE}^{\mathrm{IND-HID-CPA}}(\mathcal{A}) = 2 \cdot \Pr[\mathbf{G}_{8}^{\mathcal{A}} \Rightarrow 1] - 1$$

$$\leq 2 \cdot \Pr[\mathbf{G}_{7}^{\mathcal{A}} \Rightarrow 1] - 1 + 2L \cdot \mathbf{Adv}_{\mathcal{H}}^{adm}(\mathcal{C})$$
(12)

$$= \frac{2 \cdot \Pr[\mathbf{G}_{6}^{\mathcal{A}} \Rightarrow 1] - 1}{\delta_{min}^{L}} + 2L \cdot \mathbf{Adv}_{\mathcal{H}}^{adm}(\mathcal{C})$$
(13)

$$\leq \frac{2 \cdot \Pr[\mathbf{G}_{5}^{\mathcal{A}} \Rightarrow 1] - 1}{\delta_{min}^{L}} + 2\left(\frac{1}{S} + \frac{1}{2^{k}\delta_{min}^{L}}\right) + 2L \cdot \mathbf{Adv}_{\mathcal{H}}^{adm}(\mathcal{C})$$
(14)

$$\leq \quad \frac{2 \cdot \Pr[\mathbf{G}_4^{\mathcal{A}} \Rightarrow 1 \wedge \mathsf{Good}_4] - \Pr[\mathsf{Good}_4]}{\delta_{\min}^L} + 2\left(\frac{1}{S} + \frac{1}{2^k}\right) + \dots$$

$$+2L \cdot \mathbf{Adv}_{\mathcal{H}}^{adm}(\mathcal{C})$$
 (15)

$$\leq \frac{\mathbf{Adv}_{\mathcal{W}I\mathcal{B}\mathcal{E}}^{\mathrm{IND-sWID-CPA}}(\mathcal{B})}{\delta_{min}^{L}} + 2\left(\frac{1}{S} + \frac{1}{2^{k}}\right) + 2L \cdot \mathbf{Adv}_{\mathcal{H}}^{adm}(\mathcal{C})$$
(16)

Equation (12) is obtained by applying Equation (11), while Equation (13) derives from Equation (10). Equation (14) is obtained by applying the difference between game G_5 and G_6 noted in Equation (9). Equation (15) comes from that G_4 and G_5 are identical-until-bad (see Equation (8)), and finally the last result (16) is obtained by combining Equations (15) and (7).

This completes the proof of Theorem 3.2. Due to the exponential factor L, we notice that the reduction is meaningful when the maximum hierarchy's depth L is some fixed constant.

Remark 3.4 Even though our transformation requires a WIBE scheme with $\lambda \cdot L$ levels to get a HIBE with L levels, we observe that the HIBE key derivation algorithm will use the WIBE key derivation at most L times. The point is that while L is supposed to be a constant, λ can be instead non-constant, as it is the case for known constructions of admissible hash functions, whose output length depends on the number of secret key queries made by the adversary. This might have been a problem for those WIBE schemes that do not support key derivation (delegation) for a polynomial number of levels, such as our lattice-based scheme in Section 6.

3.2 Extensions of our transformation

Our transformation easily allows for two extensions.

Obtaining an IBE. If one is interested into constructing only an IBE, then our transformation easily works. In particular, we observe that to construct an IBE we can use a WIBE scheme with hierarchy of depth λ (instead of $\lambda \cdot L$). Furthermore the WIBE does not need to satisfy the delegation property. Therefore, we can state the following Corollary:

Corollary 3.5 Let IBE be the IBE scheme defined as $\mathcal{H}IBE$ using a scheme $\mathcal{W}IBE$ of depth λ . If $\mathcal{H} = \{H : \{0, 1\}^w \to \Sigma^\lambda\}$ is a family of (Q, δ_{min}) -admissible hash functions, and $\mathcal{W}IBE$ is IND-sWID-CPA-secure (even without the delegation property), then the scheme IBE described above is IND-ID-CPA-secure.

The transformation in the CML model. It is interesting to note that our transformation from a selective-secure WIBE to a fully-secure HIBE scheme works also in the CML model (whose

formal definition is given in Section 2.4). More precisely, let $\mathcal{W}I\mathcal{B}\mathcal{E}$ be a WIBE scheme that is (ρ_M, ρ_U) -CML-IND-sWID-CPA-secure, then we can build a HIBE scheme $\mathcal{H}I\mathcal{B}\mathcal{E}$ that is (ρ_M, ρ_U) -CML-IND-HID-CPA-secure. The construction is almost identical to the one described in this section. The only difference is that we have to define the Update_{user} algorithm of the HIBE. This can simply use the Update_{user} algorithm of the scheme $\mathcal{W}I\mathcal{B}\mathcal{E}$.

So, we can prove the following Corollary:

Corollary 3.6 Let \mathcal{HIBE} be the (H)IBE scheme defined as before with in addition the Update_{user} algorithm. If $\mathcal{H} = \{H : \{0,1\}^w \to \Sigma^\lambda\}$ is a family of (Q, δ_{min}) -admissible hash functions, and \mathcal{WIBE} is (ρ_M, ρ_U) -CML-IND-sWID-CPA-secure, then such scheme \mathcal{HIBE} is (ρ_M, ρ_U) -CML-IND-HID-CPA-secure.

The proof can be obtained by rewriting the proof of Theorem 3.2. In particular, observe that in the CML-IND-HID-CPA game the challenge identity is declared using the **Challenge** procedure, before running **LR**. So, the main observation here is that if the coloring induced by the admissible hash function on the set of identities queried by the adversary does not set **bad** \leftarrow **true**, then the simulator can forward all queries to the respective oracles of the WIBE game. In particular, consider the good event that the challenge identity \overrightarrow{ID}^* is red. In this case the adversary is allowed to query **Leak** and **Update** on identities \overrightarrow{ID} that would decrypt \overrightarrow{ID}^* . However, these identities, according to our transformation, will match the challenge pattern P^* . So the simulator can simply forward such queries to the **Leak** and **Update** procedures of the WIBE game.

Remark 3.7 In this work, we only consider the CML model with no leakage from the master secret. However, we notice that our transformation would easily extend even to the more general case where leakage from the master secret is allowed, as long as admissible hash functions are "public-coins", i.e., the randomness used to sample them can be publicly revealed (this is the case for the constructions considered in our work).

4 Selective WIBE schemes from selective HIBE

In this section we investigate methodologies that allow to build a selective-pattern secure WIBE scheme starting from a HIBE which is selective-identity secure. In particular, we identify conditions under which this transformation works, and then, in Appendix B, we will show that such conditions are satisfied by many known schemes, e.g., [8, 10, 35]. Then, by combining this result, i.e., a transformation from selective-identity secure HIBE to selective-pattern secure WIBE, with the result of Section 3, i.e., a conversion from selective-pattern secure WIBE to fully-secure HIBE, we obtain a methodology which allows to turn a selective-secure HIBE into a fully-secure one.

Towards this goal, our first contribution is a notion of security for HIBE schemes, called *secu*rity under correlated randomness that we describe in details in the following section. Informally speaking, a HIBE scheme is secure under correlated randomness if it satisfies two properties. First, when given the encryptions of the same message, using the same randomness, for two identities $(ID_1, \ldots, ID_j, \ldots, ID_L)$ and $(ID_1, \ldots, ID'_j, \ldots, ID_L)$ which differ only at position j, then one can efficiently generate a new valid ciphertext encrypting the same message, and intended for any identity matching the pattern $(ID_1, \ldots, *, \ldots, ID_L)$. Second, given the above property, we want to ensure that one can actually generate ciphertexts only for those identities matching the pattern $(ID_1, \ldots, *, \ldots, ID_L)$, meaning that an adversary should not be able to generate encryptions for any identity outside the pattern.

Next, in Section 4.2 we will show that HIBE schemes that are secure under correlated randomness can be used to build selective-pattern secure WIBE. Given the intuition above, one can easily imagine how this property can allow to create encryptions that are intended to patterns rather than to identities. Roughly speaking, what we will show in the next sections is a way to extend this intuition to a more general case which allows to describe patterns according to the WIBE notion.

4.1 Security under Correlated Randomness for HIBE

In this section, we introduce the notion of "security under correlated randomness" for HIBE schemes. Before going into the formal details, we first give an informal description of the notion.

The main idea follows the intuition sketched before as follows. Assume that one is given encryptions of the same message with the same randomness but for different identities $\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n$. Then there should be an efficient algorithm that allows to efficiently generate a new ciphertext encrypting the same message but intended to another identity $\overrightarrow{ID'} \in I\mathcal{D'} \subseteq I\mathcal{D}$. The first technical point is to delineate which is this subspace $I\mathcal{D'}$ of the identity space. So, our first contribution is to show that $I\mathcal{D'}$ follows from the differences between the identities $\overrightarrow{ID^0}, \ldots, \overrightarrow{ID^n}$. More technically, we will show that starting from any set of identities $\overrightarrow{ID^0}, \ldots, \overrightarrow{ID^n}$ one can define a matrix Δ whose column *i* contains the vector which is computed as the difference between $\overrightarrow{ID^0}$ and $\overrightarrow{ID^i}$ (i.e., $\Delta^{(i)} = \overrightarrow{ID^0} - \overrightarrow{ID^i}$). Then the identity subspace $I\mathcal{D'}$ fixed by $\overrightarrow{ID^0}, \ldots, \overrightarrow{ID^n}$ is the set of all identities that can be obtained by making affine operations over $\overrightarrow{ID^0}$ and Δ . (i.e., $\overrightarrow{ID^0}$ plus vectors obtained from integer linear combinations of vectors in Δ). Given this property, encrypting a message with the same randomness for $\overrightarrow{ID^0}, \ldots, \overrightarrow{ID^n}$ is equivalent to encrypting for the entire subspace $I\mathcal{D'}$. As one may guess, this is already a first step towards building a WIBE, in which the set of recipients of an encryption is actually a subspace of $I\mathcal{D}$ described by the pattern P.

With this intuition in mind, now we start to formalize these ideas. First, we introduce some useful definitions.

Let $\mathcal{H}I\mathcal{B}\mathcal{E}$ be an HIBE of depth $\leq L$ with identity space $I\mathcal{D} = \mathbb{Z}_q^{\lambda}$ (where $q \geq 2$ and $\lambda \geq 1$). Basically, in a very generic way, we assume that identities can be represented as integers (in \mathbb{Z}_q , for some integer q) or vectors of integers (e.g., binary strings). An identity at level ℓ in the hierarchy is represented as a vector $\overrightarrow{ID} = (ID_1, \ldots, ID_\ell)$ such that each $ID_i \in I\mathcal{D}$. When $I\mathcal{D} = \mathbb{Z}_q^{\lambda}$ and $\lambda > 1$, we further denote $ID_i = (ID_{i,1}, \ldots, ID_{i,\lambda})$.

Let $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$ be *n* identities such that each $\overrightarrow{ID}^i \in I\mathcal{D}^\ell$. We define the matrix

$$\Delta(\overrightarrow{ID}^{0},\ldots,\overrightarrow{ID}^{n}) = \left[\Delta^{(1)}||\cdots||\Delta^{(n)}\right] \in I\mathcal{D}^{(\ell \times n)} = \mathbb{Z}_{q}^{(\lambda\ell \times n)}$$

where each vector $\Delta^{(i)} = \overrightarrow{ID}^0 - \overrightarrow{ID}^i, \forall i = 1, ..., n$. For simplicity, when it is clear from the context, we simply write $\Delta(\overrightarrow{ID}^0, ..., \overrightarrow{ID}^n)$ as Δ . The matrix Δ essentially contains the differences between each vector \overrightarrow{ID}^i and the vector \overrightarrow{ID}^0 . We notice that the choice of \overrightarrow{ID}^0 for computing the differences is completely arbitrary. The same definition may be given w.r.t. \overrightarrow{ID}^k , for any $0 \le k \le n$. Moreover, given $\left[\overrightarrow{ID}^0||\cdots||\overrightarrow{ID}^n\right]$, notice that the matrix $\left[\overrightarrow{ID}^0||\Delta\right]$ is obtained through a simple transformation of the former that fixes the first column \overrightarrow{ID}^0 , and then makes linear operations (i.e., subtractions) over the other columns. Hence, the information contained in the two matrices is the same.

For any $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n) \in (I\mathcal{D}^{\ell})^{n+1}$, we define its Span as:

$$\mathsf{Span}(\overrightarrow{ID}^0,\ldots,\overrightarrow{ID}^n) = \{\overrightarrow{ID} = \overrightarrow{ID}^0 + \Delta \cdot \vec{k} : \vec{k} \in \mathbb{Z}^n\}.$$

Here, we remark that all the operations must be defined according to the identity space. For instance, if the identity space is $I\mathcal{D} = \mathbb{Z}_q^{\lambda}$, then we have to consider additions mod q.

Given the notion of Span described above, we can now define in a more formal way the property for HIBE schemes that we call *Ciphertext Conversion*. Intuitively, the property says that, given n+1ciphertexts (C_0, \ldots, C_n) encrypting the same message with the same randomness r, under identities $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$ respectively, one can generate a new ciphertext (encrypting the same message) intended to any $\overrightarrow{ID} \in \text{Span}(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$. More formally:

Property 1 (Ciphertext Conversion) A HIBE scheme satisfies Ciphertext Conversion if there exists an efficient algorithm $Convert(mpk, C_0, \overrightarrow{ID}^0, \ldots, C_n, \overrightarrow{ID}^n, \overrightarrow{ID})$ such that: for all honestly generated public keys mpk, for any identities $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n) \in (I\mathcal{D}^\ell)^{n+1}$, for all messages $m \in \mathcal{M}$, and all ciphertexts (C_0, \ldots, C_n) such that $C_i \stackrel{\$}{\leftarrow} Enc(mpk, \overrightarrow{ID}^i, m; r)$, it works as follows. If $\overrightarrow{ID} \in Span(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$ (or \overrightarrow{ID} is an ancestor of some $\overrightarrow{ID}' \in Span(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$), then it outputs a ciphertext C such that $C = Enc(mpk, \overrightarrow{ID}, m; r')$ for some randomness r' (not necessarily equal to r). Otherwise it outputs \bot .

$\underline{\mathbf{procedure \ Initialize}}(\overrightarrow{ID}^0,\ldots,\overrightarrow{ID}^n)$	procedure LR (m_0, m_1)
$(mpk, msk) \stackrel{\hspace{0.1em}\hspace{0.1em}\hspace{0.1em}}{\leftarrow} {\sf Setup} \ ; \ \beta \stackrel{\hspace{0.1em}\hspace{0.1em}\hspace{0.1em}}{\leftarrow} \{0,1\}$	Sample randomness r
Return mpk	for $i = 0$ to n do
$\frac{\textbf{procedure Extract}(\overrightarrow{ID})}{sk_{\overrightarrow{ID}} \stackrel{\$}{\leftarrow} KeyDer(msk,\overrightarrow{ID})}$ Return $sk_{\overrightarrow{ID}}$	$C_i \stackrel{\$}{\leftarrow} Enc(mpk, \overrightarrow{ID}^i, m_\beta; r)$ Return (C_0, \dots, C_n) procedure Finalize (β') Return $(\beta' = \beta)$

Figure 8: Game IND-sCR-CPA.

From the point of view of the security, however, this property itself may be "subtle". In fact, only the fact that a HIBE scheme satisfies Ciphertext Conversion does not guarantee that any user receiving a set of ciphertexts generated for different identities with the same randomness cannot get more than what the **Convert** algorithm produces. Therefore, we would like to be sure that identities in $\mathsf{Span}(\overrightarrow{ID}^0,\ldots,\overrightarrow{ID}^n)$ (and their ancestors) are the only ones that might recover the message m, when given (C_0, \ldots, C_n) . In order to formalize this idea, we introduce the notion of selective-security under correlated randomness (IND-sCR-CPA). To do this we define the game IND-sCR-CPA (see Figure 8) which consists of four procedures that can be run by an adversary \mathcal{A} . In particular, we assume that \mathcal{A} makes only one query (m_0, m_1) to the **LR** procedure such that $|m_0| = |m_1|$ (i.e., the two messages have the same length), and that \mathcal{A} makes only *legitimate* queries to the **Extract** procedure. In this setting, we say that a query $\mathbf{Extract}(\overrightarrow{ID})$ is not legitimate if $\overrightarrow{ID} \in \mathsf{Span}(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$ or \overrightarrow{ID} is an ancestor of some $\overrightarrow{ID}' \in \mathsf{Span}(\overrightarrow{ID}^0, \dots, \overrightarrow{ID}^n)$. Moreover, we consider the above game parametrized by a distribution \mathcal{R} over $I\mathcal{D}^{\ell \times (n+1)}$. This means that the game is valid whenever the adversary executes the **Initialize** procedure on input a set $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n) \in \mathcal{R}$. Notice that by properly varying the distribution \mathcal{R} one can obtain different notions of security. For instance, one can obtain the most generic definition by letting \mathcal{R} be all the space $I\mathcal{D}^{\ell \times (n+1)}$. Otherwise, one can impose some conditions on the distributions of the tuple $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$, and thus one obtains a more restricted notion of security.

We define the IND-sCR-CPA-advantage of an adversary \mathcal{A} against a scheme \mathcal{HIBE} as

$$\mathbf{Adv}_{\mathcal{HIBE}}^{\mathrm{IND-sCR-CPA}}(\mathcal{A}) = 2 \cdot \Pr[\mathrm{IND-sCR-CPA}^{\mathcal{A}} \Rightarrow 1] - 1.$$

Definition 4.1 [IND-sCR-CPA-security] A HIBE scheme \mathcal{HIBE} is IND-sCR-CPA-secure w.r.t. the distribution \mathcal{R} if for any PPT adversary \mathcal{A} , $\mathbf{Adv}_{\mathcal{HIBE}}^{\mathrm{IND-sCR-CPA}}(\mathcal{A})$ is at most negligible.

4.2 From HIBE selective-secure under Correlated Randomness to selective-secure WIBE

Now that we have defined the notion of selective security under correlated randomness (IND-sCR-CPA), we can show how to build a selective-pattern secure WIBE from an IND-sCR-CPA-secure HIBE. Let us first introduce some notation and basic definitions.

Let $I\mathcal{D} = \mathbb{Z}_q^{\lambda}$ be the identity space, for some $q \geq 2$ and $\lambda \geq 1$. For any pattern $P \in (I\mathcal{D} \cup \{*\})^{\ell}$ we define the function $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n) \leftarrow F(P)$ as follows. Let $\{j_1, \ldots, j_{n'}\} = W(P) \subseteq \{1, \ldots, \ell\}$ be the set of levels in which P contains *. Let $n = n' \cdot \lambda$, $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$ is defined as:

$$ID_i^0 = \begin{cases} P_i & if \ P_i \neq * \\ 0^{\lambda} & if \ P_i = * \end{cases}$$

$$ID_{i,m}^{k+l-1} = \begin{cases} -1 & \text{if } i = j_k \wedge m = l \\ ID_{i,m}^0 & \text{otherwise} \end{cases} : \begin{array}{c} 1 \le k \le n', \ 1 \le l \le \lambda \\ 1 \le i \le \ell, \ 1 \le m \le \lambda \end{cases}$$

Moreover, we let $B = [B^{(1)} || \cdots || B^{(\ell \lambda)}] \in \{0, 1\}^{\ell \lambda \times \ell \lambda}$ be the canonical basis of $\mathbb{R}^{\ell \lambda}$.

The function F(P) allows to specify a set of identities $(\overline{ID}^0, \ldots, \overline{ID}^n)$ such that the induced subspace $\text{Span}(\overline{ID}^0, \ldots, \overline{ID}^n)$ is exactly the same subspace described by the pattern P. Intuitively, this can be seen by looking at the way the identities are defined. \overline{ID}^0 is equal to P on all the positions different from * and 0 elsewhere. Instead each identity \overline{ID}^i is such that its difference with \overline{ID}^0 leads to a 1 in the *single* position where they differ and 0 elsewhere. Basically, this means that the matrix Δ obtained from F(P) contains a subset of vectors in B. In this way, adding linear combinations of these vectors to \overline{ID}^0 allows to reach identities \overline{ID} such that $ID_i = P_i$ where $P_i \neq *$, while ID_i can take any value in $I\mathcal{D}$ in those positions i where $P_i = *$. Notice that the number n of such linearly independent vectors strictly depends on the number of * in P. We formally show this property of $F(\cdot)$ by proving the following claim:

Claim 1 For any
$$P \in (I\mathcal{D} \cup \{*\})^{\ell}$$
 and any $\overrightarrow{ID} \in I\mathcal{D}^{\ell}$ it holds $\overrightarrow{ID} \in \mathsf{Span}(F(P))$ iff $\overrightarrow{ID} \in *P$.

Proof: By looking at the definitions of $F(\cdot)$ and Δ we can see that

$$\mathsf{Span}(F(P)) = \{ \overrightarrow{\textit{ID}} = \overrightarrow{\textit{ID}}^0 + \Delta(F(P)) \cdot \vec{k} : \vec{k} \in \mathbb{Z}^n \}$$

where

$$ID_i^0 = \begin{cases} P_i & if \ P_i \neq * \\ 0^\lambda & if \ P_i = * \end{cases}$$

Moreover, in the case where all the identities $\overrightarrow{ID}^1, \ldots, \overrightarrow{ID}^n$ are generated by F(P), we have

$$\Delta = \left[B^{(j_1)} || \cdots || B^{(j_1 + \lambda - 1)} || \cdots || B^{(j_{n'})} || \cdots || B^{(j_{n'} + \lambda - 1)} \right] \in \{0, 1\}^{\ell \lambda \times n' \lambda}$$

where $\{j_1, \ldots, j_{n'}\} = W(P)$ and the $B^{(i)}$'s are vectors of the basis B.

It is not hard to see that $\{\overrightarrow{ID} \in ID^{\ell} : \overrightarrow{ID} \in P, \ell \cdot \lambda = |P|\}$ describes the same space.

4.2.1 Our WIBE scheme

Let $\mathcal{H}I\mathcal{BE} = (\mathsf{Setup}', \mathsf{KeyDer}', \mathsf{Enc}', \mathsf{Dec}', \mathsf{Convert})$ be a HIBE scheme with identity space $I\mathcal{D} = \mathbb{Z}_q^{\lambda}$ (for $q \geq 2$ and $\lambda \geq 1$), and equipped with an efficient algorithm Convert satisfying Property 1. Then we construct the scheme $\mathcal{W}I\mathcal{BE} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ as follows.

Setup: Return the output of Setup'.

KeyDer $(sk_{\overrightarrow{ID}}, \overrightarrow{ID})$: Run $sk_{\overrightarrow{ID}} \stackrel{s}{\leftarrow} KeyDer'(sk_{\overrightarrow{ID}}, \overrightarrow{ID})$ and output $sk_{\overrightarrow{ID}}$.

Enc(mpk, P, m): Let $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n) \leftarrow F(P)$. For all i = 0 to n, compute $C_i \stackrel{\$}{\leftarrow} \text{Enc}'(mpk, \overrightarrow{ID}^k, m; r)$, where r is taken at random from the randomness space of \mathcal{HIBE} .Enc. Finally, output $C = (C_0, \ldots, C_n)$.

 $\mathsf{Dec}(sk_{\overrightarrow{ID}}, C, P)$: If $\overrightarrow{ID} \notin_* P$, then output \bot . Otherwise, compute $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n) \leftarrow F(P)$, run $C' \leftarrow \mathsf{Convert}(mpk, C_0, \overrightarrow{ID}^0, \ldots, C_n, \overrightarrow{ID}^n, \overrightarrow{ID})$ and then output $m \leftarrow \mathsf{Dec}'(sk_{\overrightarrow{ID}}, C')$.

Remark 4.2 We notice that our transformation assumes a HIBE scheme that works with the identities returned by our function $F(\cdot)$. This function is defined so that it assigns to the identities values P_i , 0 or -1. However, it may be the case that 0 and/or 1 are not considered valid values in some specific identity space (e.g., assume $I\mathcal{D} = \mathbb{Z}_q \setminus \{0\}$). This issue can be overcome by observing that everything still works if one takes any two different (and valid) values of the identity space, instead of 0 and 1. All we want is that when we compute the matrix Δ , if two identity components are equal, then their difference becomes 0, otherwise they lead to some value c (not necessarily 1). To see that everything works even with any constant c, observe that it is possible to consider our operations over Δ/c .

Now, we prove the security of our scheme via the following theorem.

Theorem 4.3 If \mathcal{HIBE} satisfies Property 1 and is IND-sCR-CPA-secure w.r.t. $\mathcal{R} = I\mathcal{D}^{\ell \times (n+1)}$, then the scheme \mathcal{WIBE} described above is correct and IND-sWID-CPA secure.

Proof: First, we show that the scheme is correct. Assume by contradiction that \mathcal{WIBE} is not correct. Then at least one the following cases holds:

- 1. the scheme \mathcal{HIBE} is not correct;
- 2. the algorithm **Convert** is not correct;
- 3. $\exists P \in (I\mathcal{D} \cup \{*\})^{\ell}, \overrightarrow{ID}' \in I\mathcal{D}^{\ell} \text{ such that } \overrightarrow{ID}' \in * P \text{ and } \overrightarrow{ID}' \notin \mathsf{Span}(F(P)).$

Since we know by assumption that \mathcal{HIBE} is correct and it satisfies Property 1, we can focus only on the last case. However, by the property proved in Claim 1, the span of F(P) and the identities matching P describe the same space, thus also the third case cannot occur. Furthermore, it is easy to see that this holds even for any ancestor of $\overrightarrow{ID'}$.

To prove the security of the scheme we show an algorithm \mathcal{B} that simulates the game IND-sWID-CPA to an adversary \mathcal{A} against the scheme $\mathcal{W}I\mathcal{B}\mathcal{E}$ while \mathcal{B} is playing the game IND-sCR-CPA against the scheme $\mathcal{H}I\mathcal{B}\mathcal{E}$. We describe the simulator \mathcal{B} as follows:

Initialize(P^*): \mathcal{B} receives as input a pattern P^* . It computes $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n) \leftarrow F(P^*)$, executes $mpk \leftarrow \textbf{Initialize}(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$, and returns mpk to \mathcal{A} .

Extract(\overrightarrow{ID}): \mathcal{B} runs $sk_{\overrightarrow{ID}} \leftarrow \mathbf{Extract}(\overrightarrow{ID})$ and returns $sk_{\overrightarrow{ID}}$ to \mathcal{A} .

LR(P, m_0, m_1): first, \mathcal{B} runs $(C_0, \ldots, C_n) \leftarrow \mathbf{LR}(m_0, m_1)$. Let P be a sub-pattern of P^* such that $\ell' = |P| \leq |P^*|$. If $P = P^*$, then \mathcal{B} returns $C = (C_0, \ldots, C_n)$. Otherwise, let $(\widehat{ID}^0, \ldots, \widehat{ID}^{n'}) \leftarrow F(P)$. By our definition of $F(\cdot)$, each of the identities in F(P) is also in $\mathsf{Span}(F(P^*))$. Therefore, we can use the convert algorithm to compute $\widehat{C}_i \leftarrow \mathsf{Convert}(mpk, C_0, \overrightarrow{ID}^0, \ldots, C_n, \overrightarrow{ID}^n, \widehat{ID}^i)$, for all i = 0 to n'. Finally, \mathcal{B} returns $C = (\widehat{C}_0, \ldots, \widehat{C}_{n'})$.

Finalize (β') : Let β' be the bit returned by \mathcal{A} . \mathcal{B} concludes the simulation by executing **Finalize** (β') .

If \mathcal{B} can simulate the game IND-sWID-CPA to \mathcal{A} in a perfect way, then it is easy to see that its advantage is the same as that of \mathcal{A} . The correctness of the simulation mainly follows from Claim 1, for which the pattern P^* and $\mathsf{Span}(F(P^*))$ describe exactly the same subspace of identities of length $|P^*|$. Thus the identities asked by \mathcal{A} to the key derivation oracle always fall into the set of key derivation queries that are "legitimate" for \mathcal{B} . Moreover, by construction and correctness, the challenge ciphertext is distributed as in the real case.

Therefore, we have:

$$\mathbf{Adv}_{\mathcal{HBE}}^{\mathrm{IND-sCR-CPA}}(\mathcal{B}) = \mathbf{Adv}_{\mathcal{WIBE}}^{\mathrm{IND-sWID-CPA}}(\mathcal{A})$$

4.2.2 A sufficient distribution for building a WIBE

In the previous section, we showed that an HIBE scheme satisfying Property 1 and the notion of selective-security under correlated randomness can be transformed into a WIBE. In particular, Theorem 4.3 considers the most general definition where the distribution \mathcal{R} is arbitrary, i.e., $\mathcal{R} = I\mathcal{D}^{\ell \times (n+1)}$. However, we observe that in order for the transformation to work, it is sufficient to consider a more restricted distribution that we call \mathcal{R}_{WIBE} .

Let $B = [B^{(1)}||\cdots||B^{(\ell\lambda)}] \in \{0,1\}^{\ell\lambda \times \ell\lambda}$ be the canonical basis defined in the previous section. We define the distribution

$$\mathcal{R}_{WIBE} = \{ (\overrightarrow{ID}^0, \dots, \overrightarrow{ID}^n) : \overrightarrow{ID}^0 \in \mathbb{Z}_q^{\lambda \ell}, \ \overrightarrow{ID}^i = \overrightarrow{ID}^0 + k_i \cdot B^{(j_i)}, \ 1 \le i \le n,$$

 $j_i \in \{1, \ldots, \lambda\ell\}, \vec{k} \in \mathbb{Z}^n\}$

It is interesting to observe that for any pattern P the identities obtained from F(P) follow the distribution \mathcal{R}_{WIBE} . We prove the following simple claim.

Claim 2 For any pattern $P \in (I\mathcal{D} \cup \{*\})^{\ell}$ we have $F(P) \in \mathcal{R}_{WIBE}$.

Proof: Since in \mathcal{R}_{WIBE} the first identity \overrightarrow{ID}^0 can take any value in $\mathbb{Z}_q^{\lambda \ell}$, we have only to check that the other identities are distributed correctly, namely, for all $1 \leq i \leq n$, there exist $k_i \in \mathbb{Z}$ and $j_i \in \{1, \ldots, \lambda \ell\}$ such that $\overrightarrow{ID}^i = \overrightarrow{ID}^0 + k_i \cdot B^{(j_i)}$. However, looking at the definition of F(P), one can notice that this is the case by setting $\vec{k} = 1^n$ and by taking $\{j_1, \ldots, j_n\} = \{j_1, \ldots, j_1 + \lambda - 1, j_2, \ldots, j_2 + \lambda - 1, \ldots, j_{n'}, \ldots, j_{n'} + \lambda - 1\}$ where $\{j_1, \ldots, j_{n'}\} = W(P)$.

Hence, we can combine the results of Theorem 4.3 and Claim 2 to obtain the following Corollary.

Corollary 4.4 If $\mathcal{H}I\mathcal{B}\mathcal{E}$ satisfies Property 1 and is secure under the IND-sCR-CPA notion w.r.t. \mathcal{R}_{WIBE} , then the scheme $\mathcal{W}I\mathcal{B}\mathcal{E}$ described above is correct and IND-sWID-CPA-secure.

5 A leakage-resilient WIBE scheme based on Decision Linear

In this section, we give the construction of a WIBE scheme that is selective-pattern-secure in the CML model based on the linear assumption. The scheme is obtained by adapting the selective-secure IBE construction given in [19]. Precisely, our WIBE works for hierarchies of identities of depth λ where each identity is one bit long, and it does not support delegation. Although this may seem a limitation, our result can be interpreted as a proof of concept to support the utility of our framework for obtaining fully-secure (H)IBEs. In fact, the given WIBE scheme satisfies the assumptions of Corollaries 3.5 and 3.6, and thus it allows to obtain a fully-secure IBE in the CML model.

Useful definitions. Before describing the scheme, we recall some notation and definitions. We use bold uppercases to denote matrices $\mathbf{A} \in \mathbb{Z}_p^{m \times n}$ and lowercases to denote vectors $v \in Z_p^n$. Row vectors are denoted by v^T . If $\mathbf{A} \in \mathbb{Z}_p^{m \times n}$ and g is a generator of a group \mathbb{G} of order p, then we denote by $g^{\mathbf{A}} \in \mathbb{G}^{m \times n}$ the matrix where each element $(g^{\mathbf{A}})_{i,j} = g^{(\mathbf{A})_{i,j}}$.

For a matrix $\mathbf{A} \in \mathbb{Z}_p^{m \times n}$ its rank is the maximum number of linearly independent rows or columns in the matrix. Let $Rk_r(\mathbb{Z}_p^{m \times n})$ be the set of all matrices in $\mathbb{Z}_p^{m \times n}$ that have rank r. For any matrix $\mathbf{A} \in \mathbb{Z}_p^{m \times n}$ we call the span, $\text{Span}(\mathbf{A})$, its row span, i.e., $\text{Span}(\mathbf{A}) = \{w^T \cdot \mathbf{A} : w \in \mathbb{Z}_p^m\}$. The kernel of a matrix \mathbf{A} is the linear space orthogonal to its span: $Ker(\mathbf{A}) = \{x \in \mathbb{Z}_p^n : \mathbf{A} \cdot x = 0\}$.

DECISION LINEAR ASSUMPTION. We recall the *Decision Linear assumption*, first introduced in [11]. In particular, we consider a general form in terms of a matrix that has been shown to be implied by the standard one [30]. Let \mathbb{G}, \mathbb{G}_T be bilinear groups of prime order p such that $p > 2^k$, where $k \in \mathbb{N}$ is the security parameter. For $r \in \{2,3\}$, let $g^{\mathbb{C}} \stackrel{*}{\leftarrow} D_r$ be the distribution defined by the following process: sample $\mathbb{C} \stackrel{*}{\leftarrow} Rk_r(\mathbb{Z}_p^{3\times 3})$ and output $g^{\mathbb{C}}$. Then, the decision linear assumption holds in \mathbb{G} if for any PPT adversary \mathcal{A} distributions D_2 and D_3 are computationally indistinguishable, that is

$$|\Pr[\mathcal{A}(g^{\mathbf{C}}) = 1|g^{\mathbf{C}} \stackrel{\$}{\leftarrow} D_2] - \Pr[\mathcal{A}(g^{\mathbf{C}}) = 1|g^{\mathbf{C}} \stackrel{\$}{\leftarrow} D_3]| \le \frac{1}{2} + negl(k)$$

The IBE scheme in [19]. Here we recall the IBE scheme proposed by Brakerski et al. [19, 18].

Setup: Sample $\mathbf{A}_0 \stackrel{\$}{\leftarrow} \mathbb{Z}_p^{2\times 2}$ (such that it is non-invertible). Sample $g^{\mathbf{A}_{i,b}} \stackrel{\$}{\leftarrow} \mathbb{G}^{2\times 2}$ for all $i \in [\lambda]$ and $b \in \{0,1\}$. Set $mpk = (g^{\mathbf{A}_0}, \{g^{\mathbf{A}_{i,b}}\}_{i\in[\lambda],b\in\{0,1\}})$ and $msk = \mathbf{A}_0^{-1}$. For any identity $ID \in \{0,1\}^{\lambda}$, we denote $\mathbf{A}_{ID} = \mathbf{A}_0 \|\mathbf{A}_{1,ID_1}\| \dots \|\mathbf{A}_{\lambda,ID_{\lambda}}$.

KeyDer(*msk*, *ID*): Sample vectors $x_1, x_2 \stackrel{\$}{\leftarrow} ker(\mathbf{A}_{ID})$ and set $sk_{ID} = g^{\mathbf{X}} = [g^{x_1}|g^{x_2}]$.

Update_{user}(sk_{ID}): Sample $\mathbf{T} \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} Rk_2(\mathbb{Z}_p^{2\times 2})$ and output $sk'_{ID} = g^{\mathbf{X}\cdot\mathbf{T}}$.

- Enc(*mpk*, *ID*, \mathfrak{m}): Let \mathbf{A}_{ID} be the matrix obtained from \mathbf{A} as described before. The ciphertext is $C = g^{v^T} \in \mathbb{G}^{2(\lambda+t^*+1)}$ where $v^T \stackrel{\hspace{0.1em}{\leftarrow}}{\leftarrow} \mathsf{Span}(\mathbf{A}_{ID})$ if one encrypts $\mathfrak{m} = 0$, and $v^T \stackrel{\hspace{0.1em}{\leftarrow}}{\leftarrow} \mathbb{Z}_p^{2(\lambda+t^*+1)}$ if one encrypts $\mathfrak{m} = 1$.
- Dec (sk_{ID}, C, P) : To decrypt, compute $e(C, sk_{ID}) = e(g^{v^T}, g^{\vec{x}})$. If the result is $e(g, g)^{\vec{0}}$, then output 0, otherwise output 1.

This scheme is proved selective-secure under the decision linear assumption with leakage rate $(\rho_U, \rho_M) = \left(\frac{c \cdot \log k}{(2\lambda+2) \cdot \log p}, \frac{\lambda-2-\gamma}{2(\lambda+1)}\right)$ (for all $\gamma, c > 0$).

Our WIBE scheme. In what follows we describe the algorithms of our WIBE. We observe that the Setup, KeyDer and Update_{user} algorithms are the same as the ones of the IBE scheme by Brakerski *et al.* [19]. The only difference is in encryption and decryption.

Setup: Sample $\mathbf{A}_0 \stackrel{\$}{\leftarrow} \mathbb{Z}_p^{2\times 2}$ (such that it is invertible). Sample $g^{\mathbf{A}_{i,b}} \stackrel{\$}{\leftarrow} \mathbb{G}^{2\times 2}$ for all $i \in [\lambda]$ and $b \in \{0,1\}$. Set $mpk = (g^{\mathbf{A}_0}, \{g^{\mathbf{A}_{i,b}}\}_{i \in [\lambda], b \in \{0,1\}})$ and $msk = \mathbf{A}_0^{-1}$. For any identity $ID \in \{0,1\}^{\lambda}$, we denote $\mathbf{A}_{ID} = \mathbf{A}_0 \|\mathbf{A}_{1,ID_1}\| \dots \|\mathbf{A}_{\lambda,ID_{\lambda}}$.

KeyDer(*msk*, *ID*): Sample vectors $x_1, x_2 \stackrel{\$}{\leftarrow} ker(\mathbf{A}_{ID})$ and set $sk_{ID} = g^{\mathbf{X}} = [g^{x_1}|g^{x_2}]$.

Update_{user}(sk_{ID}): Sample $\mathbf{T} \stackrel{s}{\leftarrow} Rk_2(\mathbb{Z}_p^{2\times 2})$ and output $sk'_{ID} = g^{\mathbf{X} \cdot \mathbf{T}}$.

- $\mathsf{Enc}(mpk, P, \mathfrak{m}): \text{ Let } \mathbf{A}_P = \mathbf{A}_0 \|\mathbf{A}_{1,P_1}\| \dots \|\mathbf{A}_{\lambda,P_{\lambda}} \in \mathbb{Z}_p^{2 \times 2(\lambda+t^*+1)} \text{ where } \mathbf{A}_{i,*} = \mathbf{A}_{i,0} |\mathbf{A}_{i,1} \text{ and } t^* \text{ is the number of wildcards } * \text{ in } P. \text{ The ciphertext is } C = g^{v^T} \in \mathbb{G}^{2(\lambda+t^*+1)}, \text{ where } v^T \stackrel{\$}{\leftarrow} \mathsf{Span}(\mathbf{A}_P) \text{ if one encrypts } \mathfrak{m} = 0, \text{ and } v^T \stackrel{\$}{\leftarrow} \mathbb{Z}_p^{2(\lambda+t^*+1)} \text{ to encrypt } \mathfrak{m} = 1.$
- $\mathsf{Dec}(sk_{ID}, C, P)$: For every $ID \in P$, notice that \mathbf{A}_{ID} is contained in \mathbf{A}_P . From the ciphertext C we extract the ciphertext components C' that correspond to the matrix \mathbf{A}_{ID} (notice that $C' \in \mathsf{Span}(\mathbf{A}_{ID})$). Compute $e(C', sk_{ID})$. If the result is $e(g, g)^0$, then output 0, otherwise output 1.

By looking at the way encryption works, this scheme can also be seen as though it follows the correlated-randomness paradigm described in Section 4. Indeed, encrypting to a pattern Pconsists into making an encryption w.r.t. to the pattern's components different from *, plus the encryption (with the same linear combination, i.e., the same randomness) for both the bits of the pattern's components that are equal to *. For some technical reasons (related to the specific IBE construction), it does not seem possible to prove the security of the WIBE by showing a direct black-box reduction to the selective-security of the IBE. Thus, we prove directly that this WIBE scheme is selective-secure in the CML model under the decision linear assumption.

Theorem 5.1 If the Decision Linear assumption holds in \mathbb{G} , then, for any polynomial $\lambda \geq 3$, the WIBE scheme described above is (ρ_M, ρ_U) -CML-IND-sWID-CPA-secure with leakage-rate $(\rho_U, \rho_M) = \left(\frac{c \cdot \log k}{(4\lambda+6) \log p}, \frac{2\lambda-4-\gamma}{4(\lambda+1)}\right)$ for all $\gamma, c > 0$.

Proof: The proof of this theorem is obtained by adapting the proof of selective-security of the IBE scheme given in [18].

Assume by contradiction there exists an adversary \mathcal{A} that is able to break the security of the given $\mathcal{W}I\mathcal{B}\mathcal{E}$ scheme with non-negligible probability $1/2 + \epsilon$. Then we show how to build an algorithm \mathcal{B} that solves the Decision Linear problem with probability $1/2 + \epsilon^2/32 - negl(k)$.

 \mathcal{B} is given as input a matrix $g^{\mathbf{C}} \in \mathbb{G}^{3\times 3}$ such that either $C \stackrel{\$}{\leftarrow} Rk_2(\mathbb{Z}_p^{3\times 3})$ or $C \stackrel{\$}{\leftarrow} Rk_3(\mathbb{Z}_p^{3\times 3})$. In particular, we assume that the first two rows of C have always rank 2, and the last row is either a linear combination of the first two rows or it is chosen independently at random.

 \mathcal{B} simulates the game procedures as follows.

Initialize (P^*) : Let $P^* \in (\{0,1\} \cup \{*\})^{\lambda}$. Without loss of generality and for ease of exposi-

tion, assume that the wildcard symbols are all at the last t positions, and let $\lambda = \lambda' + t$, $\ell = \lambda + 1, \ell' = \lambda' + 1$. \mathcal{B} samples a full-rank matrix $\mathbf{B} \stackrel{*}{\leftarrow} \mathbb{Z}_p^{3 \times 2\ell' + 4t}$ and it sets $g^{\mathbf{V}} = g^{\mathbf{C} \cdot \mathbf{B}} \in \mathbb{G}^{3 \times 2\ell' + 4t}$. We interpret $\mathbf{B} = [\mathbf{B}_0 || \mathbf{B}_1 || \cdots || \mathbf{B}_{\lambda'} || \mathbf{B}_{\lambda'+1,0} || \mathbf{B}_{\lambda'+1,1} || \cdots || \mathbf{B}_{\lambda,0} || \mathbf{B}_{\lambda,1}]$ and $\mathbf{V} = \begin{bmatrix} \mathbf{A} \\ v^T \end{bmatrix}$ where $\mathbf{A} \in \mathbb{Z}_p^{2 \times 2\ell' + 4t}$ are the first two rows. Moreover, we interpret $g^{\mathbf{A}}$ as $\begin{bmatrix} g^{\mathbf{A}_0} || g^{\mathbf{A}_{1,P_1^*} || \cdots || g^{\mathbf{A}_{\lambda',P_{\lambda'}^*} || g^{\mathbf{A}_{\lambda'+1,0} || g^{\mathbf{A}_{\lambda'+1,1} || \cdots || g^{\mathbf{A}_{\lambda,0} || g^{\mathbf{A}_{\lambda,1}}} \end{bmatrix}$, and $g^{v^T} = \begin{bmatrix} g^{v_0^T} || g^{v_1^T} || g^{v_{\lambda'}^T} || g^{v_{\lambda'+1,0}^T || g^{v_{\lambda'+1,1}^T || \cdots || g^{v_{\lambda,0}^T || g^{v_{\lambda,1}^T}} \end{bmatrix}$

For all i = 1 to λ' , \mathcal{B} chooses $\mathbf{A}_{i,1-P_i^*} \stackrel{\$}{\leftarrow} \mathbb{Z}_p^{2\times 2}$ full-rank.

The master public key is $mpk = (g^{\mathbf{A}_0}, g^{\mathbf{A}_{1,0}}, g^{\mathbf{A}_{1,1}}, \dots, g^{\mathbf{A}_{\lambda,0}}, g^{\mathbf{A}_{\lambda,1}})$

- **Extract** (\overrightarrow{ID}) : Let $ID \in \{0,1\}^{\lambda}$ be the queried identity such that $ID \notin P^*$. Then, there exists an index $j \in [\lambda']$ such that $ID_j \neq P_j^*$. So, \mathcal{B} can use the knowledge of \mathbf{A}_{j,ID_j} to sample a secret key sk_{ID} in $Ker(\mathbf{A}_{ID})$.
- **LR**(*P*): Let *P* be a pattern matching *P*^{*} and wlog assume that it has the first $\lambda'' \ge \lambda'$ components that are not wildcards. \mathcal{B} sets the challenge ciphertext as

$$C^* = \left[g^{v_0^T} || g^{v_1^T} || g^{v_{\lambda'}^T} || g^{v_{\lambda'+1, P_{\lambda'+1}}^T} || \cdots || g^{v_{\lambda'', P_{\lambda''}}^T} || g^{v_{\lambda''+1, 0}^T} || g^{v_{\lambda''+1, 1}^T} || \cdots || g^{v_{\lambda, 0}^T} || g^{v_{\lambda, 1}^T} \right]$$

Finalize(β'): Let β' be the bit received by \mathcal{A} . \mathcal{B} returns the same bit β' .

- Leak(f, ID) If a secret key for identity ID has been already generated (and maybe also updated), answer with $f(sk_{ID,j})$ where j is its most updated version. Otherwise, first proceed with the generation of the secret key.
 - Let $ID \in P^*$. The simulator proceeds as follows.

Let $\mathbf{B}_{ID} = \left[\mathbf{B}_{0}||\cdots||\mathbf{B}_{\lambda'}||\mathbf{B}_{\lambda'+1,ID_{\lambda'+1}}||\cdots||\mathbf{B}_{\lambda,ID_{\lambda}}\right] \in \mathbb{Z}_{p}^{3\times 2\ell}$ be the matrix obtained by extracting the columns related to ID from \mathbf{B} .

Sample a random $\mathbf{X}_{ID} \in \mathbb{Z}_p^{2\ell \times (2\ell-3)}$ such that $\mathbf{B}_{ID} \cdot \mathbf{X}_{ID} = 0$ (notice that \mathbf{X}_{ID} is of rank $2\ell-3$). Next, pick $\mathbf{T}_{ID,0} \stackrel{*}{\leftarrow} \mathbb{Z}_p^{(2\ell-3)\times 2}$ of rank 2 and compute $sk_{ID,0} = g^{\mathbf{X}_{ID} \cdot \mathbf{T}_{ID,0}} \in \mathbb{G}^{2\ell \times 2}$ and return $f(sk_{ID,0})$.

Notice that the columns of \mathbf{X}_{ID} are within negligible statistical distance from a set of $2\ell - 3$ random vectors in $Ker(\mathbf{A}_{ID})$. Moreover, we have the special property that \mathbf{X}_{ID} is orthogonal to the challenge ciphertext components that correspond to identity ID, that is $v_{ID}^T \cdot \mathbf{X}_{ID} = 0$ (v_{ID}^T) is defined in the usual way by extracting the columns related to ID). Clearly, this means that such secret keys have a malformed distribution. However, we will argue that due to the fact that subspaces are leakage-resilient this change is not noticeable by the adversary.

Update(f, ID): Assume that for the identity ID we have already generated a secret key (as described in the **Leak** procedure) using a matrix \mathbf{X}_{ID} . Let j be a counter for the total number of update queries asked by \mathcal{A} . We denote by $sk_{ID,j} \stackrel{\$}{\leftarrow} D_{\mathbf{X}_{ID}}$ the following process: choose a random $\mathbf{T}_{ID,j} \stackrel{\$}{\leftarrow} \mathbb{Z}_p^{(2\ell-3)\times 2}$ and compute $sk_{ID,j} = g^{\mathbf{X}_{ID}\cdot\mathbf{T}_{ID,j}}$ (using the same matrix \mathbf{X}_{ID} sampled to generate the secret key).

To answer the j-th update query for identity ID, the simulator proceeds as follows.

- 1. First, sample an updated secret key $sk_{ID,j} \stackrel{\$}{\leftarrow} D_{\mathbf{X}_{ID}}$.
- 2. For all leakage values $\alpha \in \{0,1\}^{c \log k}$ check if α is "good" w.r.t. $sk_{ID,j}$.
 - For $\mu = 1$ to M, \mathcal{B} runs a fresh emulation of \mathcal{A} in which all the update queries for time periods n > j are answered according to the real distribution. That is, $\forall n > j$, \mathcal{B} computes $sk_{ID,n} \stackrel{\$}{\leftarrow} \mathsf{Update}_{user}(sk_{ID,n-1}, \mathbf{R}_n)$ where $\mathbf{R}_n \stackrel{\$}{\leftarrow} \mathbb{Z}_p^{2\times 2}$ and answers with $f_n(sk_{ID,n}, \mathbf{R}_n)$. Moreover, in this emulated game \mathcal{A} is given a legal challenge ciphertext, namely \mathcal{B} flips a random bit β and gives to \mathcal{A} a legal encryption of β . All **Leak** queries (f_n, ID) in time periods n > j are answered with $f_n(sk_{ID,n})$ as described before.
 - If in the μ -th emulation \mathcal{A} returns $\beta' = \beta$, then \mathcal{B} sets $Z_{\mu} = 1$, otherwise $Z_{\mu} = 0$.
 - Let $\overline{Z}_{(j,\alpha,sk_{ID,j})} = \frac{1}{M} \sum_{\mu=1}^{M} Z_{\mu}$.
 - If j = 1 and $\overline{Z}_{(1,\alpha,sk_{ID,1})} \geq \frac{1}{2} + \frac{3}{4}\epsilon \frac{\epsilon}{2(\tau+1)}$, then α is "good" and set $\eta_1 = \overline{Z}_{(1,\alpha,sk_{ID,1})}$.
 - Otherwise, if j > 1 and $\overline{Z}_{(j,\alpha,sk_{ID,j})} \geq \eta_{j-1} \frac{\epsilon}{2(\tau+1)}$, then α is "good" and set $\eta_j = \overline{Z}_{(j,\alpha,sk_{ID,j})}$.
- 3. If no good α is found and L = 1, then abort and return a random bit.
- 4. If no good α is found and L > 1, then repeat again from Step 1. If after $J = \tau k/\epsilon$ times no good α is found, then abort the simulation and return a random bit.

To prove the theorem we have to show that the simulator succeeds with non-negligible probability whenever \mathcal{A} is successful with non-negligible probability. As we mentioned earlier, our proof is an extension of the one given by Brakerski *et al.* in [18] to prove their IBE².

Let Λ be the event that all the estimations (of \mathcal{A} 's probability of success in the emulated game) made by \mathcal{B} during the simulation are close enough to the real value. More precisely, for $j = 1, \ldots, \tau$, all $\alpha \in \{0,1\}^{c \log k}$ and for every secret key $sk_{ID,j}$ generated during the simulation, let $P_{(j,\alpha,sk_{id,j})}$ be defined as follows. Fix the coin tosses of \mathcal{B} until it receives the *j*-th update query from the adversary. $P_{(j,\alpha,sk_{id,j})}$ is the probability, conditioned on this fixing, that \mathcal{A} wins in an emulated game where in the *j*-th update query \mathcal{B} generates the secret key $sk_{ID,j}$ and returns α as the corresponding leakage value. The rest of the emulated game is defined as described before, i.e., updating the secret keys with the correct distribution. Let P_j be the value of $P_{(j,\alpha,sk_{id,j})}$ for the values $(\alpha, sk_{ID,j})$ that are selected by \mathcal{B} in the simulation.

Basically, by making M emulations, \mathcal{B} tries to estimate the probability $P_{(j,\alpha,sk_{id,j})}$ as $\overline{Z}_{(j,\alpha,sk_{ID,j})}$. A is the event that all such estimations are close enough. Formally, Λ is true if for all $1 \leq j \leq \tau$, all identities ID asked to **Update**, for every $\alpha \in \{0,1\}^{c \log k}$ and all the values $sk_{ID,j}$ generated by \mathcal{B} ,

$$\left|\bar{Z}_{(j,\alpha,sk_{ID,j})} - P_{(j,\alpha,sk_{ID,j})}\right| \le \frac{\epsilon}{8\tau}$$

Notice that Λ implies that $\forall j = 1, \dots, \tau$ it holds $|\eta_j - P_j| \leq \frac{\epsilon}{8\tau}$. To prove the theorem, we will show that the following claims hold.

Claim 3 $Pr[\Lambda] = 1 - negl(k).$

^{2}More precisely, the technical proof is that of the public-key encryption scheme, [18], Section 6.

Proof: This proof is essentially the same as that of Claim 6.4 in [18]. We recall it here for completeness.

The idea is to apply a Chernoff bound to every tuple $(j, \alpha, sk_{ID,j})$ used in the game to get:

$$\Pr\left[\left|\bar{Z}_{(j,\alpha,sk_{ID,j})} - P_{(j,\alpha,sk_{ID,j})}\right| > \frac{\epsilon}{8\tau}\right] \le 2e^{-\epsilon^2 M/(32\tau^2)} = 2e^{-k/32}.$$

Since the total number of tuples $(j, \alpha, sk_{ID,j})$ is at most $\tau \cdot k^c \cdot J = poly(k)$, the claim follows by union bound.

Claim 4 $\Pr[b = b' | \mathcal{B} \text{ does not abort } \wedge \Lambda] \geq \frac{1}{2} + \frac{\epsilon}{8} - negl(k).$

Proof: Since the event Λ occurs, we have that $\eta_{\tau} \geq \frac{1}{2} + \frac{\epsilon}{4}$. Also, Λ implies that $P_{\tau} \geq \eta_{\tau} - \frac{\epsilon}{2(\tau+1)} \geq \frac{1}{2} + \frac{\epsilon}{8}$. However, notice that P_{τ} is the probability that \mathcal{A} wins in the last emulated game where the challenge ciphertext is a "legal" encryption of a bit β which does not depend on \mathbf{C} , that is it is independent of b. Instead, we are interested into bounding the probability that \mathcal{A} wins in the simulation provided by \mathcal{B} when it is given the malformed challenge ciphertext $g^{v_P^T}$ where $v_P^T \cdot \mathbf{Y}_j$ holds for all secret keys $sk_{ID,j} = g^{\mathbf{Y}_j}$ of the identities ID matching P. Thus, we show that such malformed ciphertext cannot change \mathcal{A} 's probability of success by a non-negligible amount.

If b = 0 (i.e., **C** is rank 2), then g^{v^T} is a valid encryption of 0 and the secret keys all have the correct distribution w.r.t. such ciphertext. The more tricky case is when b = 1 (i.e., **C** is rank 3). To show that the distribution seen by \mathcal{A} in the last emulated game is not too different from the one seen in the simulation, we apply the result of Theorem 5.3 in [18] to the subspace $Ker(\mathbf{A}_{ID})$ for all the identities ID that have been asked to the **Leak** procedure. First, notice that in our simulation \mathbf{X}_{ID} is generated as a set of column vectors that are within negligible statistical distance from a set of $2\ell - 3$ random vectors in $Ker(\mathbf{A}_{ID})$. Let v^T be the ciphertext seen by \mathcal{A} in the simulation, and let $u \stackrel{\$}{\leftarrow} \mathbb{Z}_p^{2\ell}$ be a legal ciphertext encrypting 1 (like the one seen by \mathcal{A} in the last emulated game, when $\beta = 1$).

We can describe the view of the adversary in the simulation as

$$View_{S} = \left\{ \mathbf{A}, v, \{ h_{ID_{k},0}(g^{\mathbf{X}_{ID_{k}} \cdot \mathbf{T}_{ID_{k},0}}), h_{ID_{k},1}(g^{\mathbf{X}_{ID_{k}} \cdot \mathbf{T}_{ID_{k},1}}), \ldots \}_{k=1}^{Q} \right\}$$

where ID_1, \ldots, ID_Q $(Q \leq \tau)$ are all the identities for which \mathcal{A} asked a **Leak** query to \mathcal{B} , and for each identity ID we consider all the leakage from the memory and the updates, represented by the functions $h_{ID,0}, h_{ID,1}, \ldots$, etc. Moreover, recall that in $View_S$ each \mathbf{X}_{ID} is taken uniformly at random in $Ker(\mathbf{B}_{ID})$.

Our goal, is to show that this view is indistinguishable from

$$View_{R} = \left\{ \mathbf{A}, u, \{ h_{ID_{j},0}(g^{\mathbf{X}_{ID_{j}} \cdot \mathbf{T}_{ID_{j},0}}), h_{ID_{j},1}(g^{\mathbf{X}_{ID_{j}} \cdot \mathbf{T}_{ID_{j},1}}), \ldots \}_{j=1}^{Q} \right\}$$

which is \mathcal{A} 's view in the last emulated game, where the only change is that \mathcal{A} is given a legal challenge ciphertext u.

To show indistinguishability we consider some intermediate distributions. Let $View_{S,j,r}$ be a modification of $View_S$ in which the matrix \mathbf{X}_{ID} corresponding to the first j identities asked to **Leak** is sampled as $\mathbf{X}_{ID_k} \stackrel{\$}{\leftarrow} Ker^{(2\ell-3)}(\mathbf{B}_{ID_k})$ ($\forall k \leq j$), whereas $\forall k > j \ \mathbf{X}_{ID_k} \stackrel{\$}{\leftarrow} Ker^{(2\ell-3)}(\mathbf{A}_{ID})$. Moreover, for $r \in \{0, 1\}$, the key updates of the first j - 1 + r identities are answered legally, i.e., $Z_{ID_k,i} \stackrel{\$}{\leftarrow} Ker^2(\mathbf{A}_{ID_k})$:

$$View_{S,j,r} = \left\{ \mathbf{A}, v, \{ h_{ID_k,0}(g^{\mathbf{Z}_{ID_k,0}}), \dots \}_{k=1}^{j-1+r}, \{ h_{ID_k,0}(g^{\mathbf{X}_{ID_k} \cdot \mathbf{T}_{ID_k,0}}), \dots \}_{k=j+r}^Q \right\}$$

Clearly, $View_S = View_{S,1,0}$. Consider $View_{S,Q,1}$, and let $View_{R,Q,1}$ be the same distribution except that it includes a random $u \stackrel{\$}{\leftarrow} \mathbb{Z}_p^{2\ell}$ instead of v:

$$View_{R,Q,1} = \left\{ \mathbf{A}, u, \{ h_{ID_k,0}(g^{\mathbf{Z}_{ID_k,0}}), h_{ID_k,1}(g^{\mathbf{Z}_{ID_k,1}}), \ldots \}_{k=1}^Q \right\}$$

Notice that the distributions $View_{S,Q,1}$ and $View_{R,Q,1}$ are the same as in $View_{S,Q,1}$ the vector v is independent of **A** (since it is an encryption of b = 1).

Then, similarly to the previous case, for $j \in \{1, ..., Q\}$, $r \in \{0, 1\}$ we define the distributions $View_{R,j,r}$. Observe that $View_{R,1,0} = View_R$.

To show that the view of the adversary does not change when it is given a malformed ciphertext v, we have to show that $View_S \approx View_R$. First, observe that $\forall j = 1$ to Q we have that $View_{S,j-1,1} \approx View_{S,j,0}$ because the only change is in the sampling of \mathbf{X}_{ID_j} . Then, for all $j \in \{1, \ldots, Q\}$, we can apply the result of Theorem 5.3 in [18] to show that $View_{S,j,0} \approx View_{S,j,1}$ (resp. $View_{R,j,0} \approx$ $View_{R,j,1}$), both with statistical distance δ obtained as follows. First, we apply Theorem 5.3 with parameters $\hat{m} = 2\ell - 2$, $\hat{\ell} = 2\ell - 3$ and the functions h_i whose range is $|W| = p^{4\ell \cdot \rho_M} \cdot k^c$ (assuming that they contain the maximum amount of leakage from memory and updates). The value δ can be derived from Theorem 5.2:

$$\delta = J \cdot \tau \cdot \sqrt{\frac{|W|}{p^{2\ell-6}}} = J \cdot \tau \cdot \sqrt{\frac{p^{4\ell \cdot \rho_M} \cdot k^c}{p^{2\ell-6}}}$$

If we assign our value of $\rho_M = \frac{2\ell - 6 - \gamma}{4\ell}$, then we obtain $\delta = J \cdot \tau \cdot p^{-\gamma/2} \cdot k^{c/2} = negl(k)$. If we finally consider that the total number of view transitions is at most $\tau = poly(k)$, and we sum up, then we obtain that the statistical distance between the views $View_S$ and $View_R$ is negligible.

Claim 5 $\Pr[\mathcal{B} \text{ does not abort}|\Lambda] \ge \epsilon/4 - negl(k).$

Proof: The proof of this claim is essentially a rewriting of that of Claim 6.2 in [18]. We provide it here for completeness.

To show that the simulator \mathcal{B} does not abort with sufficient probability, we show that it could be replaced by another algorithm \mathcal{B}' whose output distribution is computationally indistinguishable from that of \mathcal{B} . Then, we will bound the probability that \mathcal{B}' aborts.

 \mathcal{B}' is identical to \mathcal{B} except that in the simulation of the **Update** procedure, instead of sampling the updated secret keys as $sk_{ID,j} \stackrel{\$}{\leftarrow} D_{\mathbf{X}_{ID}}$, it uses a different distribution D' defined as follows. Let $sk_{ID,0} = g^{\mathbf{Y}_0}$ be the "first" secret key of ID. D' consists into sampling a matrix $\mathbf{R} \stackrel{\$}{\leftarrow} Rk_2(\mathbb{Z}_p^{2\times 2})$ and output $g^{\mathbf{Y}_0\cdot\mathbf{R}}$. So, this means that \mathcal{B}' samples the updated keys using the correct distribution of the Update_{user} algorithm. Observe that in \mathcal{B}' the leakage value α is still found in the same way

as in \mathcal{B} . However, if the keys are sampled from the correct distribution, then a good leakage value α should exist.

If we look at the difference between the distributions produced by \mathcal{B} and \mathcal{B}' respectively, this is essentially the difference between $D_{\mathbf{X}_{ID}}$ and D'. As it is shown in [18] (Claim 6.2), these are computationally indistinguishable under the decision linear assumption. So, one obtains that

 $\Pr[\mathcal{B} \text{ does not abort}|\Lambda] \ge \Pr[\mathcal{B}' \text{ does not abort}|\Lambda] - negl(k)$

Finally, we bound the probability that \mathcal{B}' does not abort, conditioned on that the event Λ occurs. First, consider the first update query. By Markov's inequality we have that

$$\Pr\left[P_1 \geq \frac{1}{2} + \frac{3}{4}\epsilon\right] \geq \frac{\epsilon}{4}$$

holds when the good leakage value α is used. Observe that the probability is taken over the random choices in the generation of $sk_{ID,1}$, that the first keys are generated with a distribution which is within negligible statistical distance from the real one, and that the updates are done with the correct distribution. Thus, a good α has to exist (with all but negligible probability), and for that $\alpha sk_{ID,1}$ is found with probability at least $\epsilon/4$. Since Λ occurs, \mathcal{B}' 's estimation in that case is close enough to P_1 , and thus \mathcal{B}' does not abort in the first query with at least probability $\epsilon/4$.

Then, we can apply the same argument recursively to query i (for i = 2 to τ). Recall that in query i we did not abort so far, and thus the value η_{i-1} is well defined, and $P_{i-1} \ge \eta_{i-1} - \frac{\epsilon}{8\tau}$. Again, by Markov's inequality, when the good leakage value α is found, it holds

$$\Pr\left[P_i \ge \eta_{i-1} - \frac{\epsilon}{4\tau}\right] \ge \frac{\epsilon}{8\tau}$$

Moreover, J trials guarantee that a secret key $sk_{ID,i}$ for such good α is found with all but negligible probability. Since Λ holds, the estimation of P_i is close enough, and thus \mathcal{B}' does not abort in step i with all but negligible probability.

Therefore, we have that

$$\Pr[\mathcal{B}' \text{ does not abort}|\Lambda] \ge \frac{\epsilon}{4} - negl(k)$$

L

This concludes the proof of the Theorem.

6 Lattice-Based WIBE

In this section, we give a construction of a lattice-based selectively-secure WIBE, based on the hardness of the LWE Problem [32], that very closely resembles the selectively-secure HIBE construction from [22]. In fact, the master/secret key generation and delegation procedures are exactly the same for the HIBE and the WIBE. The only difference lies in the encryption and decryption procedures; yet even there, the distinction is fairly minor. For the benefit of those readers familiar with the HIBE of [22], we present the constructions of the WIBE along with the construction of the HIBE and also try to use the same notational conventions.

6.1 Lattices and the LWE Problem

LATTICES AND GAUSSIAN DISTRIBUTIONS. An *m*-dimensional integer lattice Λ is an additive subgroup of \mathbb{Z}^m . Every *m*-dimensional lattice can be described by a column basis $\mathbf{B} \in \mathbb{Z}^{m \times m}$. We will denote by $\tilde{\mathbf{B}}$, the *Gram-Schmidt* orthogonalization of the matrix \mathbf{B} . For any matrix \mathbf{B} , we define $\|\mathbf{B}\|$ to be the largest ℓ_2 norm of any column vector of \mathbf{B} . We define the lattice $\Lambda^{\perp}(\mathbf{A})$ as

$$\Lambda^{\perp}(\mathbf{A}) = \{ \mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} \equiv 0 \bmod q \}$$

and a "shifted lattice" $\Lambda_{\mathbf{y}}^{\perp}(\mathbf{A})$ as

$$\Lambda_{\mathbf{v}}^{\perp}(\mathbf{A}) = \{ \mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} \equiv \mathbf{y} \bmod q \}.$$

For any $\mathbf{A} \in \mathbb{Z}^{n \times m}$, $\mathbf{y} \in \mathbb{Z}_q^n$, and any s > 0, the distribution $D_{\Lambda_{\mathbf{y}}^{\perp}(\mathbf{A}),s}$ assigns a probability proportional to $e^{-\pi \|\mathbf{x}\|^2/s^2}$ to every $\mathbf{x} \in \Lambda_{\mathbf{y}}^{\perp}(\mathbf{A})$ and 0 everywhere else.

THE LWE PROBLEM. For any distribution χ over \mathbb{Z} , and any vector $\mathbf{x} \in \mathbb{Z}_q^n$ we define $\operatorname{Noisy}_{\chi}(\mathbf{x})$ to be the distribution obtained by first creating a vector $\mathbf{y} \in \mathbb{Z}^n$ each of whose coordinates is independently sampled according to χ , and then outputting $\mathbf{x} + \mathbf{y} \mod q$.

The decisional Learning With Errors Problem $(\mathsf{LWE}_{n,q,\chi})$ is to distinguish between the following two oracles: the oracle \mathcal{O}_0 outputs random elements in \mathbb{Z}_q^{n+1} , whereas the oracle \mathcal{O}_1 has a uniformly random secret \mathbf{s} , and whenever it is queried, it chooses a uniformly random $\mathbf{a} \in \mathbb{Z}_q^n$ and outputs $(\mathbf{a}, \mathsf{Noisy}_{\chi}(\mathbf{a}^T\mathbf{s}))$. In general, the distribution χ is set so that it produces values that are considerably smaller than q. When χ is taken to be a discrete Gaussian distribution with a particular standard deviation with respect to q, it is known that the $\mathsf{LWE}_{n,q,\chi}$ problem is as hard as certain worst-case lattice problems [32, 31].

6.2 Algorithms used in constructing the HIBE and WIBE

We now briefly describe the algorithms that were used in [22] to construct the HIBE, which we will be using in this section for constructing the WIBE.

- 1. GenBasis $(1^n, 1^m, q)$: This algorithm generates a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ (where $m = \Omega(n \log q)$) and a basis $\mathbf{S} \in \mathbb{Z}^{m \times m}$ of $\Lambda^{\perp}(\mathbf{A})$ such that the distribution of \mathbf{A} is negligibly close to uniform over $\mathbb{Z}_q^{n \times m}$ and $\|\tilde{\mathbf{S}}\| = O(\sqrt{n \log q})$.
- 2. ExtBasis($\mathbf{S}, \mathbf{A}' = \mathbf{A} || \bar{\mathbf{A}}$) : This algorithm takes as input a matrix $\mathbf{A}' = \mathbf{A} || \bar{\mathbf{A}} \in \mathbb{Z}_q^{n \times (m+\bar{m})}$ and a matrix $\mathbf{S} \in \mathbb{Z}^{m \times m}$, which is basis of $\Lambda^{\perp}(\mathbf{A})$, and outputs a matrix $\mathbf{S}' \in \mathbb{Z}^{(m+\bar{m}) \times (m+\bar{m})}$ that is a basis of $\Lambda^{\perp}(\mathbf{A}')$ such that $\|\mathbf{\tilde{S}}\| = \|\mathbf{\tilde{S}'}\|$.
- 3. SampleD(S, A, y, s): This algorithm takes as input a basis $\mathbf{S} \in \mathbb{Z}^{m \times m}$ of the lattice $\Lambda^{\perp}(\mathbf{A})$, a vector $\mathbf{y} \in \mathbb{Z}_q^n$, and a real number $s \geq \|\tilde{\mathbf{S}}\| \cdot \omega(\sqrt{\log n})$ and outputs a vector $\mathbf{z} \sim D_{\Lambda^{\perp}_{\mathbf{x}}(\mathbf{A}), s}$.
- 4. RandBasis(\mathbf{S}, s): This algorithm takes as input an $m \times m$ lattice basis \mathbf{S} and a real number $s \geq \|\tilde{\mathbf{S}}\| \cdot \omega(\sqrt{\log n})$, and outputs a basis \mathbf{S}' of the same lattice such that $\|\mathbf{S}'\| \leq s\sqrt{m}$. Furthermore, if $\mathbf{S}_0, \mathbf{S}_1$ are bases of the same lattice and $s > \max\{\|\tilde{\mathbf{S}}_0\|, \|\tilde{\mathbf{S}}_1\|\}$, then the distributions of RandBasis(\mathbf{S}_0, s) and RandBasis(\mathbf{S}_1, s) are statistically close.

6.3 Our Lattice-Based WIBE scheme

We now describe the master key generation, key derivation, encryption and decryption algorithms of our WIBE scheme. For any distribution χ over \mathbb{Z} , and any vector $\mathbf{x} \in \mathbb{Z}_q^n$ let $\text{Noisy}_{\chi}(\mathbf{x})$ be the distribution obtained by first creating a vector $\mathbf{y} \in \mathbb{Z}^n$ each of whose coordinates is independently sampled according to χ , and then outputting $\mathbf{x} + \mathbf{y} \mod q$.

Master Key Generation. We assume that the identities are of the form $\{0,1\}^t$, for all $t \leq L$. The generation of the master public and secret keys is done exactly in the same fashion in the HIBE and in the WIBE. The WIBE is parametrized by the integers n, m, q where n is the security parameter, m is an integer of size $\Omega(n \log q)$ and q is some prime whose size is related to the number of allowable key derivations, and is discussed in Section 6.3. We first run the GenBasis $(1^n, 1^m, q)$ procedure to obtain a matrix $\mathbf{A}_0 \in \mathbb{Z}_q^{n \times m}$ and a basis $\mathbf{S}_0 \in \mathbb{Z}^{m \times m}$ of $\Lambda^{\perp}(\mathbf{A})$. Then for all $(i, j) \in \{0, 1\} \times \{1, \ldots, L\}$, we generate a uniformly random matrix $\mathbf{A}_j^{(i)} \in \mathbb{Z}_q^{n \times m}$, and choose a uniformly-random $\mathbf{y} \in \mathbb{Z}_q^n$. The master public key is

$$\left[\mathbf{A}_0, \mathbf{A}_1^{(0)}, \mathbf{A}_1^{(1)}, \dots, \mathbf{A}_L^{(0)}, \mathbf{A}_L^{(1)}, \mathbf{y}\right],\$$

and the master secret key is S_0 .

Key Derivation. The key derivation procedure is again performed exactly the same for the HIBE and the WIBE. The public key of identity $id = (id_1, \ldots, id_t)$ is $(\mathbf{A}_{id}, \mathbf{y})$, where $\mathbf{A}_{id} = \mathbf{A}_0 \|\mathbf{A}_1^{(id_1)}\| \dots \|\mathbf{A}_t^{(id_t)}$. The secret key of user *id* is $(\mathbf{S}_{id}, \mathbf{x}_{id})$ where \mathbf{S}_{id} is a "short" basis of the lattice $\Lambda^{\perp}(\mathbf{A}_{id})$ and \mathbf{x}_{id} is a "short" vector satisfying $\mathbf{A}_{id}^T \mathbf{x}_{id} = \mathbf{y}$. The matrix \mathbf{S}_{id} will be used for delegation, while the vector \mathbf{x}_{id} will be used for decryption.

If a user with $id = (id_1, \ldots, id_t)$ would like to generate a secret key for a user $id' = (id_1, \ldots, id_t, id_{t+1}, \ldots, id_{t'})$ whose public key is $(\mathbf{A}_{id'} = \mathbf{A}_{id} || \mathbf{\bar{A}}, \mathbf{y})$, where $\mathbf{\bar{A}} = \mathbf{A}_{t+1}^{(id_{t+1})} || \ldots || \mathbf{A}_{t'}^{(id_{t'})}$, he computes the following:

$$\begin{split} \mathbf{S}_{id'} &\leftarrow \texttt{RandBasis}(\texttt{ExtBasis}(\mathbf{S}_{id}, \mathbf{A}_{id'}), s) \\ \mathbf{x}_{id'} &\leftarrow \texttt{SampleD}(\texttt{ExtBasis}(\mathbf{S}_{id}, \mathbf{A}_{id'}), \mathbf{A}_{id'}, \mathbf{y}, s) \end{split}$$

where $s \geq \|\widetilde{\mathbf{S}_{id}}\| \cdot \omega(\sqrt{\log n})$. We point out that with every key derivation, the value of $\|\widetilde{\mathbf{S}_{id}}\|$ increases by a factor of $\tilde{O}(\sqrt{n})$. When the norm of the secret key gets too large, decryption becomes impossible, and so, just like in [22], it is important to adjust the ratio of the size of the secret key \mathbf{S}_0 and the prime q based on how many levels of delegations one wishes to have. With each level of delegation increasing the norm of the user id by a factor of $\tilde{O}(\sqrt{n})$, the ratio between $\|\widetilde{\mathbf{S}_0}\|$ and qshould be on the order of \sqrt{n}^d , where d is the maximum allowable levels in the hierarchy. Since the LWE_{n,q,χ} problem becomes easier as q gets larger (and the distribution χ stays the same), there is a trade-off between security and the maximum number of delegation levels. We direct the reader to [22] for the precise parameters.

Encryption and Decryption. In the HIBE, encryption of a message $\kappa \in \{0, 1\}$ is performed to identity $id = (id_1, \ldots, id_t)$ by picking a random $\mathbf{r} \in \mathbb{Z}_q^n$ and outputting the pair $(\mathbf{u}_{id}, v) \in \mathbb{Z}_q^{m(t+1)+1}$, where

$$\left(\mathbf{u}_{id}, v\right) = \left(\texttt{Noisy}_{\chi}\left(\mathbf{A}_{id}^T\mathbf{r}\right), \texttt{Noisy}_{\chi}\left(\mathbf{y}^T\mathbf{r} + \kappa \cdot \lfloor q/2 \rfloor\right)\right)$$

where

$$\mathbf{A}_{id} = \mathbf{A}_0 \|\mathbf{A}_1^{(id_1)}\| \dots \|\mathbf{A}_t^{(id_t)} \tag{17}$$

and χ is some "narrow" distribution such that the LWE_{n,q, χ} problem is hard.

The decryption of the HIBE ciphertext by the identity $id = (id_1, \ldots, id_t)$ is performed as follows: for a ciphertext (\mathbf{u}_{id}, v) and secret key \mathbf{x}_{id} , the algorithm computes $v - \mathbf{x}_{id}^T \mathbf{u}_{id} \mod q$ and outputs 0 if the result is closer to 0 than to q/2, and outputs 1 otherwise.

In our WIBE, encryption is defined in essentially the same way as in the HIBE. To encrypt to a pattern $pat = (pat_1, \ldots, pat_t) \in \{0, 1, *\}^t$, we pick a random $\mathbf{r} \in \mathbb{Z}_q^n$, define

$$\mathbf{A}_{pat} = \mathbf{A}_0 \|\mathbf{A}_1^{(pat_1)}\| \dots \|\mathbf{A}_t^{(pat_t)}$$
(18)

where $\mathbf{A}_i^* := \mathbf{A}_i^{(0)} \| \mathbf{A}_i^{(1)}$, and output the pair $(\mathbf{u}_{pat}, v) \in \mathbb{Z}_q^{m(t+t_*+1)+1}$ (where t_* is the number of * in the pattern *pat*),

$$(\mathbf{u}_{pat}, v) = \left(\texttt{Noisy}_{\chi}(\mathbf{A}_{pat}^T \mathbf{r}), \texttt{Noisy}_{\chi}\left(\mathbf{y}^T \mathbf{r} + \kappa \cdot \lfloor q/2 \rfloor \right) \right).$$

Notice that instead of the matrix \mathbf{A}_{pat} being $n \times mt$ as in the HIBE, it can be as large as $n \times 2mt$ because every position pat_i that contains the wildcard * results in the concatenation of both $\mathbf{A}_i^{(0)}$ and $\mathbf{A}_i^{(1)}$ into the matrix \mathbf{A}_{pat} . Therefore the ciphertext of the WIBE could be twice as large as the HIBE ciphertext.

The decryption procedure of the WIBE is also very similar to that of the HIBE. For every $id = (id_1, \ldots, id_t) \in \{0, 1\}^t$, the matrix \mathbf{A}_{pat} contains the matrix \mathbf{A}_{id} , where \mathbf{A}_{id} is defined as in (17). Therefore, since we know $\mathbf{u}_{pat} = \operatorname{Noisy}_{\chi}(\mathbf{A}_{pat}^T\mathbf{r})$, we can retrieve from it $\mathbf{u}_{id} = \operatorname{Noisy}_{\chi}(\mathbf{A}_{id}^T\mathbf{r})$. And now, using the secret key \mathbf{x}_{id} , the user can decrypt the ciphertext (\mathbf{u}_{id}, v) the same way as in the HIBE scheme by computing $v - \mathbf{x}_{id}^T\mathbf{u}_{id} \mod q$ and outputting 0 if the result is closer to 0 than to q/2, and 1 otherwise.

6.4 Security Proof

The security proof of our scheme is a simple adaptation of the HIBE security proof in [22].

Theorem 6.1 Given an adversary \mathcal{A} who breaks the WIBE with parameters n, m, q allowing d key derivations, there exists an algorithm \mathcal{S} that solves the LWE_{n,q,χ} problem where $q > \sigma \cdot n^{d/2} \cdot poly(n)$ where σ is the standard deviation of the distribution χ and poly(n) is some fixed polynomial function in n.

Proof: S is given access to an adversary \mathcal{A} who breaks the WIBE and to an LWE oracle that either outputs uniformly random samples in \mathbb{Z}_q^{n+1} or samples from $\mathsf{LWE}_{n,q,\chi}$. When the adversary \mathcal{A} presents the pattern $pat = (pat_1, \ldots, pat_i)$, S proceeds to create the public key and the eventual "challenge" ciphertext as follows. First, the algorithm S calls the LWE oracle once to obtain a sample $(\mathbf{y}, v) \in \mathbb{Z}_q^{n+1}$. Secondly, S calls the LWE oracle m times to obtain a matrix $\mathbf{A}_0 \in \mathbb{Z}_q^{n \times m}$ and a vector $\mathbf{u}_0 \in \mathbb{Z}_q^m$. Then for all $1 \leq i \leq t$, if pat_i is either 0 or 1, S calls the LWE oracle m times to obtain a matrix $\mathbf{A}_i^{(pat_i)} \in \mathbb{Z}_q^{n \times m}$ and a vector $\mathbf{u}_i^{(pat_i)} \in \mathbb{Z}_q^m$, and if $pat_i = *$, it calls the LWE oracle 2mtimes and obtains two matrices $\mathbf{A}_i^{(0)}, \mathbf{A}_i^{(1)} \in \mathbb{Z}_q^{n \times m}$ and two vectors $\mathbf{u}_i^{(0)}, \mathbf{u}_i^{(1)} \in \mathbb{Z}_q^m$. For all pairs $(i, j) \in \{1, \ldots, L\} \times \{0, 1\}$, if the matrix $\mathbf{A}_i^{(j)}$ hasn't been defined yet, S generates a random matrix $\mathbf{A}_i^{(j)}$ along with an $m \times m$ "trapdoor" matrix $\mathbf{S}_i^{(j)}$ using the algorithm GenBasis $(1^n, 1^m, q)$. The public key is the collection

$$\left[\mathbf{A}_{0}, \mathbf{A}_{1}^{(0)}, \mathbf{A}_{1}^{(1)}, \dots, \mathbf{A}_{L}^{(0)}, \mathbf{A}_{L}^{(1)}, \mathbf{y}\right].$$

After S sends the public key to the A, A proceeds to query the secret key of any identity $id = (id_1, \ldots, id_t)$ that does not match the pattern *pat*. The public key of identity *id* is the matrix/vector

pair $(\mathbf{A}_{id}, \mathbf{y})$ where $\mathbf{A}_{id} = \mathbf{A}_0 \| \mathbf{A}_1^{(id_1)} \| \dots \| \mathbf{A}_t^{(id_t)}$. Because *id* does not match *pat*, there must be at least one *i* such that $id_i = 1 - pat_i$, and by the construction of the public key, \mathcal{S} knows a "short" matrix $\mathbf{S}_i^{(id_i)}$ that is basis for the lattice $\Lambda^{\perp} \left(\mathbf{A}_i^{(id_i)} \right)$. Therefore \mathcal{S} can use the key derivation procedure in Section 6.3 to construct a secret key $(\mathbf{S}_{id}, \mathbf{x}_{id})$ for the identity id^3 .

After \mathcal{A} finishes querying for keys, he sends a pattern pat' that matches pat and \mathcal{S} will send him an encryption of either 0 or 1 encrypted to the pattern pat'. \mathcal{S} selects a random bit $\kappa \stackrel{\$}{\leftarrow} \{0,1\}$ and sends the encryption pair $(\mathbf{u}_{pat'}, v + \kappa \cdot \lfloor q/2 \rfloor)$ where

$$\mathbf{u}_{pat'}^{T} = (\mathbf{u}_{0})^{T} \| \left(\mathbf{u}_{1}^{(pat_{1}')} \right)^{T} \| \dots \| \left(\mathbf{u}_{t}^{(pat_{t}')} \right)^{T},$$
(19)

where we define $\left(\mathbf{u}_{i}^{(*)}\right)^{T} = \left(\mathbf{u}_{i}^{(0)}\right)^{T} \|\left(\mathbf{u}_{i}^{(1)}\right)^{T}$. Notice that if the LWE oracle outputs valid pairs from the LWE_{*n*,*q*, χ} distribution for some secret vector **r**, then we have

$$(\mathbf{u}_{pat'}, v + \kappa \cdot \lfloor q/2 \rfloor) = \left(\operatorname{Noisy}_{\chi}(\mathbf{A}_{pat'}^T \mathbf{r}), \operatorname{Noisy}_{\chi}\left(\mathbf{y}^T \mathbf{r} + \kappa \cdot \lfloor q/2 \rfloor\right) \right),$$

where $\mathbf{A}_{pat'}$ is as defined in (18). Thus $(\mathbf{u}_{pat'}, v + \kappa \cdot \lfloor q/2 \rfloor)$ is a valid encryption of κ . On the other hand, if the LWE oracle just outputs random vectors, then $(\mathbf{u}_{pat'}, v + \kappa \cdot \lfloor q/2 \rfloor)$ is uniformly random and independent of κ and thus no adversary will succeed with probability > 1/2 in guessing κ . Thus if \mathcal{A} returns the correct value of κ , we guess that the LWE oracle is outputting from the LWE_{n,q,\chi} distribution, and otherwise we guess that it is outputting random values.

7 Future Directions

First, in its most general form (i.e., without restrictions on the distribution \mathcal{R}), our notion of security under correlated randomness gives a generic methodology for encrypting messages to sets S of recipients that are defined by $\mathsf{Span}(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$. In this sense, a WIBE can be seen as a special case of this notion in which the recipients' sets always have a fixed form specified by the pattern P, i.e., $S = \mathsf{Span}(F(P))$. However, one may think of a more general notion in which these sets can have a more "irregular" form that we can express using a set of identities $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$ and its Span.

Since we were mostly interested in building WIBE schemes in this work, we considered security under correlated randomness w.r.t. the distribution \mathcal{R}_{WIBE} . However, as a future direction, it would be interesting to explore whether there exist HIBE schemes that are IND-sCR-CPA-secure according to the most generic notion, i.e., without any restriction on \mathcal{R} . Perhaps more interestingly, the resulting primitive could be seen as the dual version of the notion of Spatial Encryption proposed by Boneh and Hamburg in [14]. In Spatial Encryption, ciphertexts are associated to points in \mathbb{Z}_p^{ℓ} , while secret keys correspond to affine subspaces of \mathbb{Z}_p^{ℓ} . In this setting, a ciphertext for $x \in \mathbb{Z}_p^{\ell}$ can be decrypted by any secret key for $W \subseteq \mathbb{Z}_p^{\ell}$ as long as $x \in W$. In contrast, our new notion would consider ciphertexts that are associated to affine subspaces of ID^{ℓ} .

As a second direction, it would be interesting to investigate whether our techniques could be applied to other cryptographic primitives. Indeed, the problem of selective vs. full security has already been considered in the context of other cryptographic notions, such as attribute-based encryption or

³Because S has the master public key, he has the ability to derive a rather short secret key for any identity *id* by using a small $s \ge \|\widetilde{\mathbf{S}_0}\| \cdot \omega(\sqrt{\log n})$. But it's possible that *id* is at a level in the hierarchy that has keys with larger norms, in which case S will run the key derivation procedure with a larger s. The exact size of the parameter s depends on the hierarchy structure of the WIBE.

verifiable random functions (VRFs). In the particular case of VRFs, finding a fully secure scheme has been a long standing open problem until the very recent works by Hohenberger and Waters [27] and by Boneh *et al.* [16]. In fact, both of these works can be seen as obtaining a fully secure VRF from a selective secure one. While the work of Boneh *et al.* explicitly builds a selective-secure VRF and then turns it into a fully secure one, the work of Hohenberger and Waters can be interpreted as a fully secure version of the selective-secure VRF scheme of Abdalla *et al.* [2].

Acknowledgments

This work was done while the second author was at École normale supérieure. The authors were supported in part by the European Research Council, in part by the European Commission through the ICT Program under Contract ICT-2007-216676 ECRYPT II, and in part by the French ANR-10-SEGI-15 PRINCE Project.

References

- M. Abdalla, D. Catalano, A. Dent, J. Malone-Lee, G. Neven, and N. Smart. Identity-based encryption gone wild. In M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, editors, *ICALP* 2006, Part II, volume 4052 of *LNCS*, pages 300–311. Springer, July 2006.
- [2] M. Abdalla, D. Catalano, and D. Fiore. Verifiable random functions from identity-based key encapsulation. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 554–571. Springer, Apr. 2009.
- [3] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 553–572. Springer, May 2010.
- [4] S. Agrawal, D. Boneh, and X. Boyen. Lattice basis delegation in fixed dimension and shorterciphertext hierarchical IBE. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 98–115. Springer, Aug. 2010.
- [5] M. Bellare and T. Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for Waters' IBE scheme. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 407–424. Springer, Apr. 2009.
- [6] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, ACM CCS 93, pages 62–73. ACM Press, Nov. 1993.
- [7] M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, May / June 2006.
- [8] D. Boneh and X. Boyen. Efficient selective-ID secure identity based encryption without random oracles. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, May 2004.
- [9] D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In M. Franklin, editor, CRYPTO 2004, volume 3152 of LNCS, pages 443–459. Springer, Aug. 2004.

- [10] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In R. Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 440–456. Springer, May 2005.
- [11] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, CRYPTO 2004, volume 3152 of LNCS, pages 41–55. Springer, Aug. 2004.
- [12] D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. SIAM Journal on Computing, 36(5):1301–1328, 2007.
- [13] D. Boneh and M. K. Franklin. Identity based encryption from the Weil pairing. SIAM Journal on Computing, 32(3):586–615, 2003.
- [14] D. Boneh and M. Hamburg. Generalized identity based and broadcast encryption schemes. In J. Pieprzyk, editor, ASIACRYPT 2008, volume 5350 of LNCS, pages 455–470. Springer, Dec. 2008.
- [15] D. Boneh and J. Katz. Improved efficiency for CCA-secure cryptosystems built using identitybased encryption. In A. Menezes, editor, CT-RSA 2005, volume 3376 of LNCS, pages 87–103. Springer, Feb. 2005.
- [16] D. Boneh, H. W. Montgomery, and A. Raghunathan. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In E. Al-Shaer, A. D. Keromytis, and V. Shmatikov, editors, ACM CCS 10, pages 131–140. ACM Press, Oct. 2010.
- [17] Z. Brakerski and Y. T. Kalai. A framework for efficient signatures, ring signatures and identity based encryption in the standard model. IACR Cryptology ePrint Archive, Report 2010/278, 2010. http://eprint.iacr.org/2010/086.
- [18] Z. Brakerski, Y. T. Kalai, J. Katz, and V. Vaikuntanathan. Cryptography resilient to continual memory leakage. Cryptology ePrint Archive, Report 2010/278, 2010. http://eprint.iacr. org/.
- [19] Z. Brakerski, Y. T. Kalai, J. Katz, and V. Vaikuntanathan. Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In 51st FOCS, pages 501–510. IEEE Computer Society Press, 2010.
- [20] R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. In E. Biham, editor, EUROCRYPT 2003, volume 2656 of LNCS, pages 255–271. Springer, May 2003.
- [21] R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 207–222. Springer, May 2004.
- [22] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 523–552. Springer, May 2010.
- [23] C. Cocks. An identity based encryption scheme based on quadratic residues. In B. Honary, editor, 8th IMA International Conference on Cryptography and Coding, volume 2260 of LNCS, pages 360–363. Springer, Dec. 2001.

- [24] Y. Dodis, K. Haralambiev, A. López-Alt, and D. Wichs. Cryptography against continuous memory attacks. In 51st FOCS, pages 511–520. IEEE Computer Society Press, 2010.
- [25] C. Gentry and S. Halevi. Hierarchical identity based encryption with polynomially many levels. In O. Reingold, editor, TCC 2009, volume 5444 of LNCS, pages 437–456. Springer, Mar. 2009.
- [26] C. Gentry and A. Silverberg. Hierarchical ID-based cryptography. In Y. Zheng, editor, ASI-ACRYPT 2002, volume 2501 of LNCS, pages 548–566. Springer, Dec. 2002.
- [27] S. Hohenberger and B. Waters. Constructing verifiable random functions with large input spaces. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 656–672. Springer, May 2010.
- [28] A. B. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 62–91. Springer, May 2010.
- [29] A. B. Lewko and B. Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In D. Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 455–479. Springer, Feb. 2010.
- [30] M. Naor and G. Segev. Public-key cryptosystems resilient to key leakage. In S. Halevi, editor, CRYPTO 2009, volume 5677 of LNCS, pages 18–35. Springer, Aug. 2009.
- [31] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In M. Mitzenmacher, editor, *41st ACM STOC*, pages 333–342. ACM Press, May / June 2009.
- [32] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In H. N. Gabow and R. Fagin, editors, 37th ACM STOC, pages 84–93. ACM Press, May 2005.
- [33] A. Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and D. Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 47–53. Springer, Aug. 1985.
- [34] B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In S. Halevi, editor, CRYPTO 2009, volume 5677 of LNCS, pages 619–636. Springer, Aug. 2009.
- [35] B. R. Waters. Efficient identity-based encryption without random oracles. In R. Cramer, editor, EUROCRYPT 2005, volume 3494 of LNCS, pages 114–127. Springer, May 2005.

A A proof without artificial abort

In this section we show an alternative proof for the transformation given in Section 3 which does not use an artificial abort step. In order to get this, we adopt a definition of admissible hash functions which is slightly different from both the one introduced by Boneh and Boyen [9] and the one by Cash *et al.* [22]. The main difference with the other definitions is that we require the functions to explicitly provide a lower bound and an upper bound for the probability of the simulation-enabling event, i.e., the event that the challenge identity is marked red and all the secret key queries identities are marked blue. The point of this explicit requirement is that having these bounds allows to make a reduction that avoids to introduce artificial aborts in the simulation. Moreover, we notice that the construction of admissible hash functions proposed by Boneh and Boyen in [9] already provides such bounds, and thus it can be employed in our transformation. The formal definition follows. **Definition A.1** [Admissible Hash Functions] $\mathcal{H} = \{H : \{0, 1\}^w \to \Sigma^\lambda\}$ is a family of $(Q, \delta_{min}, \delta_{max})$ admissible hash functions if for every polynomial Q = Q(k), there exists an efficiently computable function $\mu = \mu(k)$, efficiently recognizable sets $bad_H \subseteq (\{0, 1\}^w)^*$, and two inverses of polynomials $\delta_{min} = 1/\delta_{min}(k, Q)$ and $\delta_{max} = 1/\delta_{max}(k, Q)$ such that the following properties holds:

- 1. For every PPT algorithm \mathcal{A} that, on input $H \in \mathcal{H}$, outputs $\vec{x} \in (\{0,1\}^w)^{Q+1}$, there exists a negligible function $\epsilon(k)$ such that: $\mathbf{Adv}_{\mathcal{H}}^{adm}(\mathcal{A}) = \Pr[\vec{x} \in bad_H : H \leftarrow \mathcal{H}, \vec{x} \leftarrow \mathcal{A}(H)] \leq \epsilon(k)$
- 2. For every $H \in \mathcal{H}, K \stackrel{\hspace{0.1em}\hspace{0.1em}}{\leftarrow} \mathcal{K}^{(\lambda,\mu)}$, and every vector $\vec{x} \in (\{0,1\}^w)^{Q+1} \setminus bad_H$ such that $x_0 \notin \{x_1,\ldots,x_Q\}$ we have: $\delta_{min} \leq \gamma(\vec{x}) \leq \delta_{max}$
- 3. There exists a negligible function $\nu(k)$ such that: $(\delta_{max} \delta_{min}) \leq \nu(k)$

Now we can prove the following theorem:

Theorem A.2 If $\mathcal{H} = \{H : \{0, 1\}^w \to \Sigma^\lambda\}$ is a family of $(Q, \delta_{min}, \delta_{max})$ -admissible hash functions, and $\mathcal{W}I\mathcal{BE}$ is IND-sWID-CPA-secure, then the scheme $\mathcal{H}I\mathcal{BE}$ described in Section 3 is IND-HID-CPA-secure.

Proof: To prove Theorem A.2 we describe a sequence of games that allows to show that an adversary for the game IND-HID-CPA can be efficiently turned into an adversary for the game IND-sWID-CPA.

THE SIMULATOR ALGORITHM \mathcal{B} . The proof proceeds in a similar way to the one of Theorem 3.2. In Figure 9 we describe an adversary \mathcal{B} that plays game IND-sWID-CPA against the scheme \mathcal{WIBE} , by simulating the game IND-HID-CPA to an adversary \mathcal{A} . To avoid confusion between the games IND-sWID-CPA and IND-HID-CPA, we prepend the prefix **sW** to the procedures of IND-sWID-CPA.

In order to show that such simulation can be carried on efficiently, we proceed by describing a sequence of games G_0-G_5 , where G_0 is the game simulated by our algorithm \mathcal{B} , and G_5 is essentially IND-HID-CPA with some additional code that, however, does not condition the output. Our approach is based on code-based games where each game is defined as a set of procedures that can be run by the adversary.

Before focusing on the game sequence, we first show that the simulation provided by \mathcal{B} is correct whenever **bad** is not set, and that \mathcal{B} plays the game IND-sWID-CPA correctly. For ease of exposition we assume that the adversary always outputs identities of the same (maximum) length L. However, this can be formalized by assuming that for any set of identities $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^Q)$ output by \mathcal{A} , for i = 1 to Q all those \overrightarrow{ID}^i such that $|\overrightarrow{ID}^i| < L$ are padded to reach length L using some special symbol so that $F_{H_i,K_i}(ID_j^i)$ always returns B on positions j such that $|\overrightarrow{ID}^i| < j \leq L$. On the other hand, if the challenge identity has length $\ell^* < L$, then it is padded with some symbol so that $F_{H_i,K_i}(ID_j^0)$ always returns R on positions $j > \ell^*$.

First, observe that all the identities \vec{I} for which \mathcal{B} runs $\mathbf{sW.Extract}(\vec{I})$ are legitimate queries, namely they do not match the challenge pattern P^* declared by \mathcal{B} to $\mathbf{sW.Initialize}$. In the code of \mathcal{B} , if $\mathbf{sW.Extract}(\vec{I})$ is called, then there exists an index $i \in \{1, \ldots, \ell\}$ for which $F_{K_i, H_i}(ID_i) = B$, namely $I_i \neq P_i$ (and $P_i \neq *$), thus $\vec{I} \notin_* P^*$. Second, note that the ciphertext C^* is distributed as the challenge ciphertext in the game IND-HID-CPA for the scheme \mathcal{HIBE} . However, we have also to check that the procedure $\mathbf{sW.LR}$ be run on an identity $\vec{I} \in_* P^*$. To see this, observe that

Algorithm \mathcal{B} : $K_1,\ldots,K_L \stackrel{\stateseq}{\leftarrow} \mathcal{K}^{(\lambda,\mu)}$ $P^* \leftarrow (K_1, \dots, K_L)$ Run $mpk' \leftarrow \mathbf{sW.Initialize}(P^*)$ $H_1,\ldots,H_L \stackrel{s}{\leftarrow} \mathcal{H}$ $mpk \leftarrow (mpk', H_1, \ldots, H_L)$ Run $\mathcal{A}'(mpk)$, answering queries as follows: $\mathbf{Extract}(\overrightarrow{ID})$: let $\ell = |\overrightarrow{ID}|$ $sk_{\overrightarrow{ID}} \leftarrow \bot$ $\vec{I} \leftarrow (H_1(ID_1), \dots, H_\ell(ID_\ell))$ If $F_{H_i,K_i}(ID_i) = \mathbb{R} \ \forall i = 1 \ \text{to} \ \ell \ \text{Then}$ $\mathsf{bad} \leftarrow \mathsf{true}$ Else $sk_{\overrightarrow{ID}} \stackrel{*}{\leftarrow} \mathbf{sW.Extract}(\overrightarrow{I})$ Return $sk_{\overrightarrow{ID}}$ $LR(ID, m_0, m_1):$ let $\ell^* = |\overrightarrow{ID}|, \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_{\ell^*}(ID_{\ell^*}))$ $C^* \leftarrow \bot$ If $\exists i \in [\ell^*] : F_{H_i,K_i}(ID_i) = B$ Then $\mathsf{bad} \gets \mathsf{true}$ Else $C^* \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} \mathbf{sW.LR}(\vec{I}, m_0, m_1)$ return C^* let β' be \mathcal{A}' 's output If $\exists i \in [L] : X_i \in bad_{H_i}$ Then $\beta' \stackrel{\$}{\leftarrow} \{0, 1\}$ If bad = true Then $\beta' \stackrel{\$}{\leftarrow} \{0, 1\}$ $sW.Finalize(\beta')$

procedure Initialize: Games $G_0 - G_2$, G_3 001 $K_1, \ldots, K_L \stackrel{\$}{\leftarrow} \mathcal{K}^{(\lambda,\mu)}$ 002 $P^* \leftarrow (K_1, \ldots, K_L)$ 003 $(mpk', msk') \stackrel{\hspace{0.1em}\hspace{0.1em}\hspace{0.1em}}{\leftarrow} \mathcal{W}I\mathcal{B}\mathcal{E}.\mathsf{Setup}; \beta \stackrel{\hspace{0.1em}\hspace{0.1em}\hspace{0.1em}\hspace{0.1em}}{\leftarrow} \{0, 1\}$ 004 $H_1, \ldots, H_L \stackrel{\$}{\leftarrow} \mathcal{H}$ 005 $mpk \leftarrow (mpk', H_1, \ldots, H_L)$ $006 \quad cnt \leftarrow 1$ 007 return mpkprocedure Extract(\overrightarrow{ID}): Games G₀, G₁ 010 let $\ell = |\overrightarrow{ID}|, \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_\ell(ID_\ell))$ 011 $sk_{\overrightarrow{ID}} \leftarrow \bot$ 012 If $\overline{F}_{H_i,K_i}(ID_i) = \mathbb{R} \ \forall i = 1 \text{ to } \ell \text{ Then}$ 013 $\mathsf{bad} \gets \mathsf{true}$ 014 $sk_{\overrightarrow{ID}} \stackrel{\$}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{KeyDer}(msk', \vec{I})$ 015 Else $sk_{\overrightarrow{ID}} \stackrel{\$}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{KeyDer}(msk', \vec{I})$ 016 Return $sk_{\overrightarrow{ID}}$ procedure LR($\overrightarrow{ID}, m_0, m_1$): Games G₀, G₁ 020 let $\ell^* = |\overrightarrow{ID}|, \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_{\ell^*}(ID_{\ell^*}))$ 021 $C^* \leftarrow \bot$ 022 If $\exists i \in [\ell^*] : F_{H_i,K_i}(ID_i) = B$ Then 023 $bad \leftarrow true$ $C^* \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} \mathcal{W}I\mathcal{B}\mathcal{E}.\mathsf{Enc}(mpk', \vec{I}, m_\beta)$ 024 025 Else $C^* \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} \mathcal{W}I\mathcal{B}\mathcal{E}.\mathsf{Enc}(mpk', \vec{I}, m_\beta)$ 026 027 return C^* procedure Finalize(β'): Games G₀, G₁ 030 If $\exists i \in [L] : X_i \in bad_{H_i}$ Then $\beta' \stackrel{\$}{\leftarrow} \{0, 1\}$ 031 $\beta'' \leftarrow \beta'$ 032 If bad = true Then $\beta'' \stackrel{*}{\leftarrow} \{0,1\}, |\beta'' \leftarrow \beta'$ 033 If $\beta'' = \beta$ Then return 1 034 Else return 0

Figure 9: Algorithm \mathcal{B} and description of the first games of the sequence. In each procedure, if G_i is boxed, then the given procedure in G_i includes the boxed statements, whereas the other games do not include them.

the procedure is run only if bad is not set, namely when $F_{H_i,K_i}(ID_i) = \mathbb{R}$ for all $i \in [\ell^*]$, which is equivalent to say $\vec{I} \in P^*$.

THE SEQUENCE OF GAMES. Now, let us focus on the sequence of games $G_0 - G_5$. In particular, the Lemma 3.3 given below proves that we can move from the game IND-sWID-CPA played by \mathcal{B} to game G_5 . A critical part in \mathcal{B} 's simulation is that it may set **bad** \leftarrow **true** and, as a consequence, \mathcal{B} returns a random bit (basically, it fails its simulation). Such bad event depends on the values K_1, \ldots, K_L chosen by \mathcal{B} as well as on the set of identities asked by \mathcal{A} to **Extract** and **LR**. As shown in other works, such as [35], these cases are problematic as the event that the simulation fails is not independent of the adversary's view. This difficulty has been usually overcome by introducing an "artificial" abort event in the simulation that allows to balance the probability of failing so that it is sufficiently independent of the adversary's view. However, the artificial abort step needs additional computation and degrades the final concrete security of the reduction. For the case of Waters' IBE scheme, Bellare and Ristenpart proposed in [5] an alternative proof of security that avoids the need of the artificial abort and thus results in a more efficient reduction.

In the first part of our proof we extend Bellare and Ristenpart's techniques to the case of admissible hash functions. Similarly to [5], the most significant part of this proof is that we move from a game, i.e., G_0 , where K_1, \ldots, K_L are chosen at the beginning, to another game, i.e., G_4 , where these values are chosen at the end, that is after the adversary has output its set of queries. So, this choice is made independently of the adversary's queries. As we will notice later, this is not itself sufficient to conclude that the bad event is independent of the adversary's view as this event is jointly determined by both the adversary's and simulator's choices. However, the techniques in [5] helps to find the sufficient conditions to analyze the success probability of the experiment without having to consider the specific adversary's choices.

Following the notation given in Section 2, we write $G_i^{\mathcal{A}} \Rightarrow b$ to denote that an execution of game G_i by \mathcal{A} returns b. Also, let Bad_i (resp. Good_i) be the event that G_i sets (resp. does not set) $\mathsf{bad} \leftarrow \mathsf{true}$.

Our adversary \mathcal{B} and the games G_0-G_5 are described in Figures 9 and 10. When some games share a procedure with very similar code we use a compact description with boxed statements. If a procedure is shared by games G_i, G_j, \ldots, G_k , if G_i is boxed, then the code of the given procedure in G_i includes the boxed statements, whereas its code in the other games does not. To better understand the notation one may look at Figure 9 for an example. There, the **Finalize** procedure is shared by games G_0 and G_1 , and G_1 is written in a box. This means that **Finalize** in G_1 contains the statement $\beta'' \leftarrow \beta'$ of line 032, whereas this statement is not present in game G_0 .

Lemma A.3 $\operatorname{Adv}_{\mathcal{W}I\mathcal{BE}}^{\operatorname{IND-sWID-CPA}}(\mathcal{B}) = 2 \cdot \Pr[G_4^{\mathcal{A}} \Rightarrow 1 \wedge \mathsf{Good}_4] - \Pr[\mathsf{Good}_4].$

Proof: To prove the lemma we will analyze the differences between each consecutive pair of games.

First, we focus on the code of \mathcal{B} and game G_0 . The procedure **Initialize** contains in line 003 the code of **sW.Initialize**. Moreover, line 015 and line 026 contain the code of **sW.Extract** and **sW.LR** respectively. Finally, it is not hard to notice that the code of the **Finalize** procedure is an equivalent implementation of the way \mathcal{B} concludes its simulation and executes **sW.Finalize**. Therefore, we

procedure Extract(ID): Game G₂ 210 let $\ell = |I\dot{D}|, \ \vec{I} \leftarrow (H_1(ID_1), \dots, H_\ell(ID_\ell))$ 211 If $F_{H_i,K_i}(ID_i) = \mathbb{R} \ \forall i = 1 \ \text{to} \ \ell \ \text{Then}$ 212 $\mathsf{bad} \gets \mathsf{true}$ 213 $sk_{\overrightarrow{ID}} \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{KeyDer}(msk', \vec{I})(\vec{I})$ 214 Return $sk_{\overrightarrow{ID}}$ procedure $LR(\overline{ID}, m_0, m_1)$: Game G₂ 220 let $\ell^* = |\overrightarrow{ID}|, \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_{\ell^*}(ID_{\ell^*}))$ 221 If $\exists i \in [\ell^*] : F_{H_i,K_i}(ID_i) = B$ Then 222 $\mathsf{bad} \leftarrow \mathsf{true}$ 223 $C^* \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} \mathcal{W}I\mathcal{B}\mathcal{E}.\mathsf{Enc}(mpk', \vec{I}, m_\beta)$ 224 return C^* procedure Finalize(β'): Game G₂, G₃ 230 If $\exists i \in [L] : X_i \in bad_{H_i}$ Then $\beta' \stackrel{\$}{\leftarrow} \{0, 1\}$ 231 for j = 1 to cnt do 232 let $\ell_i \leftarrow |\overrightarrow{ID}^j|$ If $F_{H_i,K_i}(ID_i^j) = \mathbb{R} \ \forall i = 1 \ \text{to} \ \ell_j \ \text{Then}$ 233 $bad \leftarrow true$ 234235 If $\exists i \in [\ell^*] : F_{H_i,K_i}(ID_i^0) = B$ Then $\mathsf{bad} \gets \mathsf{true}$ 236237 If $\beta' = \beta$ Then return 1 238 Else return 0

procedure Initialize: Games $G_4 - G_5$ $\overline{400 \ (mpk', msk') \stackrel{\$}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{Setup}; \beta \stackrel{\$}{\leftarrow} \{0, 1\}}$ 401 $H_1, \ldots, H_L \stackrel{\$}{\leftarrow} \mathcal{H}$ 402 $mpk \leftarrow (mpk', H_1, \ldots, H_L)$ 403 $cnt \leftarrow 1$ 404 return mpkprocedure Extract(ID): Games $G_3 - G_5$ 310 $\overrightarrow{ID}^{cnt} \leftarrow \overrightarrow{ID}$; $cnt \leftarrow cnt + 1$ 311 let $\ell = |\overrightarrow{ID}|, \ \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_\ell(ID_\ell))$ 312 $sk_{\overrightarrow{ID}} \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} \mathcal{W}I\mathcal{B}\mathcal{E}.\mathsf{KeyDer}(msk', \overrightarrow{I})(\overrightarrow{I})$ 313 Return $sk_{\overrightarrow{ID}}$ procedure $LR(ID, m_0, m_1)$: Games $G_3 - G_5$ 320 $\overrightarrow{ID}^0 \leftarrow \overrightarrow{ID}$ 321 let $\ell^* = |\overrightarrow{ID}|, \overrightarrow{I} \leftarrow (H_1(ID_1), \dots, H_{\ell^*}(ID_{\ell^*}))$ 322 $C^* \stackrel{*}{\leftarrow} \mathcal{W}I\mathcal{BE}.\mathsf{Enc}(mpk', \vec{I}, m_\beta)$ 323 return C^* procedure Finalize(β'): Games G_4 , G_5 If $\exists i \in [L] : X_i \in bad_{H_i}$ Then $\beta' \stackrel{*}{\leftarrow} \{0, 1\}$ 430 431 $\overline{K_1,\ldots,K_L} \stackrel{\hspace{0.1em}\hspace{0.1em}\hspace{0.1em}}\leftarrow \mathcal{K}^{(\lambda,\mu)}$ 432 for j = 1 to cnt do let $\ell_i \leftarrow |\overrightarrow{ID}^j|$ 433If $F_{H_i,K_i}(ID_i^j) = \mathbb{R} \ \forall i = 1 \ \text{to} \ \ell_j$ Then 434 435 $\mathsf{bad} \gets \mathsf{true}$ 436 If $\exists i \in [\ell^*] : F_{H_i,K_i}(ID_i^0) = B$ Then 437 $bad \leftarrow true$ 438 If $\beta' = \beta$ Then return 1 439 Else return 0

Figure 10: Description of the games $G_2 - G_5$. In each procedure, if G_i is boxed, then the given procedure in G_i includes the boxed statements, whereas the other games do not include them.

have:

$$Pr[IND-sWID-CPA^{\mathcal{B}} \Rightarrow 1] = Pr[G_0^{\mathcal{A}} \Rightarrow 1]$$

$$= Pr[G_0^{\mathcal{A}}|\mathsf{Bad}_0] Pr[\mathsf{Bad}_0] + Pr[G_0^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_0]$$

$$= \frac{1}{2} Pr[\mathsf{Bad}_0] + Pr[G_0^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_0]$$
(20)

where the last line is justified from that the **Finalize** procedure of G_0 outputs a random bit when bad is set.

If we look at the differences between the games G_0 and G_1 we can observe that G_1 contains some additional lines of code (highlighted in the framed boxes). Such changes make sure that **Extract** and **LR** never return \perp . Also, in G_1 **Finalize** is modified in line 032 (by adding $\beta'' \leftarrow \beta'$) so that the procedure's output does not depend on **bad** = *true*. Since in game G_0 the events that **Extract** and **LR** return \perp and that **Finalize** takes β'' at random occur only if **bad** is set, then we have that G_0 and G_1 are identical-until-**bad**. Thus, we can apply Lemma 2.1 to obtain:

$$\Pr[\mathsf{Bad}_0] = \Pr[\mathsf{Bad}_1] \quad \text{and} \quad \Pr[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_0] = \Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_1] \tag{21}$$

Now, let us compare games G_1 and G_2 . The changes in the **Extract** and **Finalize** procedures are only syntactical. Lines 014,015 (resp. 024,026) of G_1 have been moved to line 213 (resp. 223) of G_2 . So G_2 is equivalent to G_1 :

$$\Pr[\mathsf{Bad}_1] = \Pr[\mathsf{Bad}_2] \quad \text{and} \quad \Pr[G_1^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_1] = \Pr[G_2^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_2] \tag{22}$$

Let us now consider G_2 and G_3 . In game G_2 , both **Extract** and **LR** may set **bad** in lines 211 - 212 and 221 - 222 respectively. However, this operation does no longer influence the behavior of each procedures. So, in G_3 these lines are moved to the end of the game, into the procedure **Finalize**. Moreover, in order for this change to be described correctly, G_3 introduces a counter and a labeling for the queried identities. Again, these changes in the code are only syntactical. Thus the two games are identical, and we have:

$$\Pr[\mathsf{Bad}_2] = \Pr[\mathsf{Bad}_3] \quad \text{and} \quad \Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_2] = \Pr[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_3] \tag{23}$$

Finally, we show that G_3 and G_4 are identically distributed as well. The only change is that line 001 of G_3 is moved to line 431 of **Finalize** in G_4 . Since in G_3 the values K_1, \ldots, K_L are used only into **Finalize**, this code can be postponed there. Thus we have:

$$\Pr[\mathsf{Bad}_3] = \Pr[\mathsf{Bad}_4] \quad \text{and} \quad \Pr[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_3] = \Pr[\mathsf{G}_4^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_4] \tag{24}$$

Finally, if we put together Equations (20), (21), (22), (23) and (24) we obtain:

$$\mathbf{Adv}_{\mathcal{W}I\mathcal{B}\mathcal{E}}^{\mathrm{IND-sWID-CPA}}(\mathcal{B}) = 2 \cdot \Pr[\mathrm{IND-sWID-CPA}^{\mathcal{B}} \Rightarrow 1] - 1$$
$$= \Pr[\mathsf{Bad}_4] + 2 \cdot \Pr[\mathsf{G}_4^{\mathcal{A}} \land \mathsf{Good}_4] - 1$$
$$= 2 \cdot \Pr[\mathsf{G}_4^{\mathcal{A}} \land \mathsf{Good}_4] - \Pr[\mathsf{Good}_4]$$
(25)

which completes the proof of the Lemma.

CONDITIONAL INDEPENDENCE. With the result of Lemma A.3 we reached a game where the values K_1, \ldots, K_L are chosen only at the end, after the adversary's set of queries as well as the game's output are fixed. However, to conclude the analysis we would need to evaluate $\Pr[\mathsf{Good}_4]$ which still depends on the adversary's queries. Therefore, to complete the proof we use the "Conditional Independence Lemma" stated by Bellare and Ristenpart in [5]. We restate the lemma below using our notation and we adapt it to our case. Indeed, in [5] the bad event is related to the behavior of the specific Waters' hash function, while in our case it is related to the "coloring" function F associated to the family \mathcal{H} . However, its proof remains essentially the same as the one in [5]. For completeness, we show this at the end of this section.

Let \overline{ID} be the space of identities $(\overline{ID}^0, \ldots, \overline{ID}^Q) \in (ID^L)^{Q+1}$ such that $\overline{ID}^0 \notin \{\overline{ID}^1, \ldots, \overline{ID}^Q\}$, namely for all $i \in \{1, \ldots, Q\}$, $\exists j \in [L] : ID_j^0 \neq ID_j^i$. For ease of exposition we still assumes that the queried identities are properly padded as described before.

For each hash function H_i , for $1 \le i \le L$, and any $X = (\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^Q) \in \overrightarrow{ID}$, we define the function γ_i , which is the same as the γ given in Definition A.1:

$$\gamma_i(X_i) = \Pr[F_{H_i,K_i}(ID_i^0) = \mathbb{R} \land F_{H_i,K_i}(ID_i^1) = \mathbb{B} \land \dots \land F_{H_i,K_i}(ID_i^Q) = \mathbb{B}]$$

Also, we let $\Gamma(X) = \prod_{i=1}^{L} \gamma_i(X_i)$. For any $X \in \overline{ID}$, let Q(X) be the event that an execution of $G_4^{\mathcal{A}}$ returns 1 and that \mathcal{A} outputs X as its set of queries.

Lemma A.4 [Conditional Independence (restated from [5])] For any $X \in \overline{ID}$

$$\Pr[\mathcal{G}_4^{\mathcal{A}} \Rightarrow 1 \land \mathsf{Good}_4 \land Q(X)] = \Gamma(X) \cdot \Pr[\mathcal{G}_4^{\mathcal{A}} \Rightarrow 1 \land Q(X)]$$
(26)

$$\Pr[\mathsf{Good}_4 \wedge Q(X)] = \Gamma(X) \cdot \Pr[Q(X)] \tag{27}$$

where the probabilities are taken over all the random coins of both game G_4 and the adversary.

Proof of Lemma A.4. For completeness, we give the proof of the Conditional Independence Lemma by Bellare and Ristenpart [5] adapted to the notation of our case.

Let Ω be the set of all coin tosses made in the execution of G_4 with the adversary \mathcal{A} . This can be seen as a cross product $\Omega = \Omega' \times \mathcal{K}^{(\lambda,\mu)}$. Namely, the coin tosses of an execution are a pair (ω, κ) where κ represents the choice of the vectors $K_1, \ldots, K_L \in \mathcal{K}^{(\lambda,\mu)}$ made in line 431, and ω contains all the other random coins (of both the game and the adversary).

For any $X \in \overline{ID}$, let $\Omega'(X)$ be the set of all $\omega' \in \Omega'$ such that the execution with ω' produces X(notice that when X is chosen, the vectors K_1, \ldots, K_L have not been chosen yet). Let Ω'_1 be the set of all $\omega' \in \Omega'$ such that the execution outputs 1 (again, this is determined only by random coins ω'). Let $\mathcal{K}^{(\lambda,\mu)}_{\mathsf{Good}}(X)$ be the set of $\kappa \in \mathcal{K}^{(\lambda,\mu)}$ such that it holds

$$\bigwedge_{i=1}^{L} (F_{K_i,H_i}(X_i^0) = \mathbf{R} \wedge F_{K_i,H_i}(X_i^1) = \mathbf{B} \wedge \dots \wedge F_{K_i,H_i}(X_i^Q) = \mathbf{B})$$

Notice that the set of coins for which $G_4^{\mathcal{A}} \Rightarrow 1$ is $\Omega'_1 \times \mathcal{K}^{(\lambda,\mu)}$, whereas the set of coins leading to $(\mathsf{Good}_4 \wedge Q(X))$ is $\Omega'(X) \times \mathcal{K}^{(\lambda,\mu)}_{\mathsf{Good}}(X)$. Thus

$$\begin{split} \Pr[\mathbf{G}_{4}^{\mathcal{A}} \Rightarrow 1 \wedge \mathsf{Good}_{4} \wedge Q(X)] &= \frac{|(\Omega_{1}' \times \mathcal{K}^{(\lambda,\mu)}) \cap (\Omega'(X) \times \mathcal{K}^{(\lambda,\mu)}_{\mathsf{Good}}(X))|}{|\Omega' \times \mathcal{K}^{(\lambda,\mu)}|} \\ &= \frac{|(\Omega_{1}' \cap \Omega'(X)) \times \mathcal{K}^{(\lambda,\mu)}_{\mathsf{Good}}(X)|}{|\Omega' \times \mathcal{K}^{(\lambda,\mu)}|} \\ &= \frac{|(\Omega_{1}' \cap \Omega'(X))| \cdot |\mathcal{K}^{(\lambda,\mu)}|}{|\Omega'| \cdot |\mathcal{K}^{(\lambda,\mu)}|} \cdot \frac{|\mathcal{K}^{(\lambda,\mu)}_{\mathsf{Good}}(X)|}{|\mathcal{K}^{(\lambda,\mu)}|} \\ &= \frac{|(\Omega_{1}' \cap \Omega'(X))| \cdot |\mathcal{K}^{(\lambda,\mu)}|}{|\Omega'| \times \mathcal{K}^{(\lambda,\mu)}|} \cdot \frac{|\mathcal{K}^{(\lambda,\mu)}_{\mathsf{Good}}(X)|}{|\mathcal{K}^{(\lambda,\mu)}|} \\ &= \frac{|(\Omega_{1}' \cap \Omega'(X)) \times \mathcal{K}^{(\lambda,\mu)}|}{|\Omega' \times \mathcal{K}^{(\lambda,\mu)}|} \cdot \frac{|\mathcal{K}^{(\lambda,\mu)}_{\mathsf{Good}}(X)|}{|\mathcal{K}^{(\lambda,\mu)}|} \\ &= \Pr[\mathbf{G}_{4}^{\mathcal{A}} \Rightarrow 1 \wedge Q(X)] \cdot \Gamma(X). \end{split}$$

which establishes Equation (26) of Lemma A.4. Similarly, we have:

$$\begin{aligned} \Pr[\mathsf{Good}_4 \wedge Q(X)] &= \frac{|(\Omega'(X) \times \mathcal{K}_{\mathsf{Good}}^{(\lambda,\mu)}(X))|}{|\Omega' \times \mathcal{K}^{(\lambda,\mu)}|} \\ &= \frac{|(\Omega'(X)|}{|\Omega'|} \cdot \frac{|\mathcal{K}_{\mathsf{Good}}^{(\lambda,\mu)}(X))|}{|\mathcal{K}^{(\lambda,\mu)}|} \\ &= \frac{|(\Omega'(X)| \cdot |\mathcal{K}^{(\lambda,\mu)}|}{|\Omega'| \cdot |\mathcal{K}^{(\lambda,\mu)}|} \cdot \frac{|\mathcal{K}_{\mathsf{Good}}^{(\lambda,\mu)}(X))|}{|\mathcal{K}^{(\lambda,\mu)}|} \\ &= \frac{|(\Omega'(X) \times \mathcal{K}^{(\lambda,\mu)})|}{|\Omega' \times \mathcal{K}^{(\lambda,\mu)}|} \cdot \frac{|\mathcal{K}_{\mathsf{Good}}^{(\lambda,\mu)}(X))|}{|\mathcal{K}^{(\lambda,\mu)}|} \\ &= \Pr[Q(X)] \cdot \Gamma(X). \end{aligned}$$

which establishes Equation (27) of Lemma A.4.

CONCLUDING THE PROOF. We can apply Lemma A.4 to the result of Lemma 3.3 to obtain:

$$\begin{split} \mathbf{Adv}_{\mathcal{W}\mathcal{IBE}}^{\mathrm{IND-SWID-CPA}}(\mathcal{B}) &= 2 \cdot \Pr[\mathrm{G}_{4}^{\mathcal{A}} \Rightarrow 1 \wedge \mathsf{Good}_{4}] - \Pr[\mathsf{Good}_{4}] \\ &= \sum_{X \in \overline{I\mathcal{D}}} 2\Pr[\mathrm{G}_{4}^{\mathcal{A}} \Rightarrow 1 \wedge \mathsf{Good}_{4} \wedge Q(X)] - \sum_{X \in \overline{I\mathcal{D}}} \Pr[\mathsf{Good}_{4} \wedge Q(X)] \\ &= \sum_{X \in \overline{I\mathcal{D}}} 2 \cdot \Gamma(X) \cdot \Pr[\mathrm{G}_{4}^{\mathcal{A}} \Rightarrow 1 \wedge Q(X)] - \sum_{X \in \overline{I\mathcal{D}}} \Gamma(X) \cdot \Pr[Q(X)] \end{split}$$

Now, since the functions H_1, \ldots, H_L are taken from a family of $(Q, \delta_{min}, \delta_{max})$ -admissible hash functions (and $X_i \notin bad_{H_i}$), then we have that each $\gamma_i(X_i) \in [\delta_{min}, \delta_{max}]$. Moreover, let $\Gamma_{min} = \delta_{min}^L$ and $\Gamma_{max} = \delta_{max}^L$ be a lower bound and an upper bound respectively for the function $\Gamma(X)$. Thus,

we have:

$$\mathbf{Adv}_{\mathcal{W}I\mathcal{B}\mathcal{E}}^{\mathrm{IND-SWID-CPA}}(\mathcal{B}) \geq 2 \cdot \Gamma_{min} \sum_{X \in \overline{\mathcal{ID}}} \Pr[\mathbf{G}_{4}^{\mathcal{A}} \Rightarrow 1 \wedge Q(X)] - \Gamma_{max} \sum_{X \in \overline{\mathcal{ID}}} \cdot \Pr[Q(X)]$$
$$= 2 \cdot \Gamma_{min} \Pr[\mathbf{G}_{4}^{\mathcal{A}} \Rightarrow 1] - \Gamma_{max}$$
(28)

Since Game G_5 is essentially the IND-HID-CPA' game we have:

$$\mathbf{Adv}_{\mathcal{HIBE}}^{\mathrm{IND-HID-CPA}}(\mathcal{A}) = 2 \operatorname{Pr}[\mathbf{G}_{5}^{\mathcal{A}'} \Rightarrow 1] - 1$$

$$\leq 2 \cdot \operatorname{Pr}[\mathbf{G}_{4}^{\mathcal{A}'} \Rightarrow 1] - 1 + 2L\mathbf{Adv}_{\mathcal{H}}^{adm}(\mathcal{C})$$
(29)

where Equation (29) is justified by observing that with a straightforward reduction one can show that

$$|\Pr[\mathbf{G}_{5}^{\mathcal{A}'} \Rightarrow 1] - \Pr[\mathbf{G}_{4}^{\mathcal{A}'} \Rightarrow 1]| \le L \cdot \mathbf{Adv}_{\mathcal{H}}^{adm}(\mathcal{C})),$$

by the property of admissible hash functions.

Therefore, by putting together Equations (28) and (29), we obtain:

$$\begin{aligned} \mathbf{Adv}_{\mathcal{HIBE}}^{\mathrm{IND-HID-CPA}}(\mathcal{A}) &\leq \frac{\mathbf{Adv}_{\mathcal{WIBE}}^{\mathrm{IND-sWID-CPA}}(\mathcal{B})}{\Gamma_{min}} + \frac{\Gamma_{max} - \Gamma_{min}}{\Gamma_{min}} + 2L\mathbf{Adv}_{\mathcal{H}}^{adm}(\mathcal{C}) \\ &\leq \frac{\mathbf{Adv}_{\mathcal{WIBE}}^{\mathrm{IND-sWID-CPA}}(\mathcal{B})}{\delta_{min}^{L}} + \frac{\nu(k)}{\delta_{min}^{L}} + 2L\mathbf{Adv}_{\mathcal{H}}^{adm}(\mathcal{C}) \end{aligned}$$
(30)

where Equation (30) follows from the property of admissible hash functions, together with the fact that $L \ge 1$ and $0 \le \delta_{min} \le \delta_{max} \le 1$.

This completes the proof of Theorem A.2. Due to the exponential factor L, notice that the reduction is meaningful when the maximum hierarchy's depth L is some fixed constant.

B HIBE Schemes Selective-Secure under Correlated Randomness

In this section we motivate the notion of selective-security under correlated randomness, by showing that several known HIBE schemes already satisfy this notion. In particular we show this for the pairing-based schemes by Boneh and Boyen [8], Boneh, Boyen and Goh [10], and Waters [35]. As the reader will see, our proofs show direct black-box reductions to the standard selective-security (IND-sHID-CPA) of the respective schemes. Essentially, this means that the schemes already satisfy our new notion without having to tweak the scheme or rely on other assumptions.

B.1 The case of the Boneh-Boyen HIBE [8, 1]

In this section we prove the IND-sCR-CPA-security of a variant of the Boneh-Boyen HIBE scheme that is proposed in [1]. At the end of the section we will discuss why this is not possible for the original BB scheme given in [8]. We briefly recall the scheme in Figure 11.

The scheme has identity space $I\mathcal{D} = \mathbb{Z}_p$ where p is a large prime. We first show that the scheme satisfies Property 1 by describing the following algorithm:

BB.Convert $(mpk, C^0, \overrightarrow{ID}^0, \dots, C^n, \overrightarrow{ID}^n, \overrightarrow{ID})$. Assume that $|\overrightarrow{ID}^0| = \dots = |\overrightarrow{ID}^n| = \ell$ and $|\overrightarrow{ID}| = \ell' \leq \ell$. Recall that each C^i has form $(C_1^i, \{C_{2,j}^i\}_{j=1}^\ell, C_3^i)$. First, consider the case when $\ell' = \ell$. The algorithm works as follows. Setup: $g_1, g_2 \stackrel{\$}{\leftarrow} \mathbb{G} ; \alpha \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ $h_1 \leftarrow g_1^{\alpha} ; h_2 \leftarrow g_2^{\alpha}$ $u_{i,j} \stackrel{\hspace{0.1em}\mathsf{\scriptscriptstyle\$}}{\leftarrow} \mathbb{G} ext{ for } i = 1 \dots L, j = 0, 1$ $mpk \leftarrow (g_1, g_2, h_1, u_{1,0}, \dots, u_{L,1})$ $msk \leftarrow h_2$ Return (mpk, msk)

KeyDer $(d_{(ID_1,...,ID_{\ell})}, ID_{\ell+1})$: Parse $d_{(ID_1,\ldots,ID_\ell)}$ as (d_0,\ldots,d_ℓ) $r_{\ell+1} \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ $\begin{array}{c} \overset{-_{P}}{d_{0}'} \leftarrow \overset{-_{P}}{d_{0}} \cdot \begin{pmatrix} u_{\ell+1,0} \cdot u_{\ell+1,1}^{ID_{\ell+1}} \end{pmatrix}^{r_{\ell+1}} \\ \overset{d_{\ell+1}'}{d_{\ell+1}'} \leftarrow \overset{-_{P}}{g_{1}^{r_{\ell+1}}} \end{array}$ Return $(d'_0, d_1, \ldots, d_\ell, d'_{\ell+1})$

 $Enc(mpk, \overrightarrow{ID}, m)$: $Dec(d_{(ID_1,...,ID_{\ell})}, C):$ Parse \overrightarrow{ID} as (ID_1, \ldots, ID_ℓ) $r \stackrel{s}{\leftarrow} \mathbb{Z}_p; C_1 \leftarrow g_1^r$ For $i = 1, \dots, \ell$ do $C_{2,i} \leftarrow (u_{i,0} \cdot u_{i,1}^{ID_i})^r$ $C_3 \leftarrow m \cdot \hat{e}(h_1, g_2)^r$ Return m'Return $(C_1, C_{2,1}, \ldots, C_{2,\ell}, C_3)$

Parse $d_{(ID_1,\ldots,ID_l)}$ as (d_0,\ldots,d_ℓ) Parse C as $(C_1,C_{2,1},\ldots,C_{2,\ell},C_3)$ $m' \leftarrow C_3 \cdot \frac{\prod_{i=1}^{\ell} \hat{e}(d_i, C_{2,i})}{\hat{e}(C_1, d_0)}$

Figure 11: The variant of the Boneh-Boyen HIBE scheme in [1].

Set $C_1 = C_1^0$ and $C_3 = C_3^0$. Find $\vec{k} \in \mathbb{Z}^n$ such that $\Delta \vec{k} = (\vec{ID} - \vec{ID^0})$, and compute

$$\tilde{\Delta} = \begin{pmatrix} C_{2,1}^0 / C_{2,1}^1 & C_{2,1}^0 / C_{2,1}^2 & \cdots & C_{2,1}^0 / C_{2,1}^n \\ \vdots & \vdots & & \vdots \\ C_{2,\ell}^0 / C_{2,\ell}^1 & C_{2,\ell}^0 / C_{2,\ell}^2 & \cdots & C_{2,\ell}^0 / C_{2,\ell}^n \end{pmatrix} = \left[\tilde{\Delta}^{(1)} || \cdots || \tilde{\Delta}^{(n)} \right] \in \mathbb{G}^{\ell \times n}$$

Finally, for all j = 1 to ℓ , set $C_{2,j} = C_{2,j}^0 \cdot \prod_{i=1}^n (\tilde{\Delta}^{(i)})^{k_i}$, and output $C = (C_1, \{C_{2,j}\}_{j=1}^\ell, C_3)$.

If $\ell' < \ell$, then one can first pad \overrightarrow{ID} to get $\overrightarrow{ID'}$ of length ℓ (e.g., by setting $ID'_i = ID^0_i$ for $\ell' < i \leq \ell$), and then use the above procedure to generate a ciphertext C' for $\overrightarrow{ID'}$. Finally, a valid ciphertext for ID can be obtained by removing the elements $C_{2,j}$ for $j > \ell'$.

The correctness of the algorithm can be verified by inspection.

Theorem B.1 If there exists an adversary \mathcal{A} that has IND-sCR-CPA-advantage $\geq \epsilon$ against the BB-HIBE scheme w.r.t. \mathcal{R}_{WIBE} , then there exists an adversary \mathcal{B} that has IND-sHID-CPA-advantage ϵ against the same scheme BB-HIBE. Namely:

$$\mathbf{Adv}_{BB-HIBE}^{\mathrm{IND}\text{-}\mathrm{sHID}\text{-}\mathrm{CPA}}(\mathcal{B}) = \mathbf{Adv}_{BB-HIBE}^{\mathrm{IND}\text{-}\mathrm{sCR}\text{-}\mathrm{CPA}}(\mathcal{A})$$

Proof: We prove the theorem by describing the adversary \mathcal{B} that plays the game IND-sHID-CPA against the scheme BB - HIBE by simulating the game IND-sCR-CPA to the adversary \mathcal{A} . The idea

Initialize $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$: \mathcal{B} takes as input a set of identities $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n) \in \mathcal{R}_{WIBE}$ such that $|\overrightarrow{ID}^i| = \ell^*$ for all $0 \le i \le n$. Let $\Delta = \Delta(\overrightarrow{ID}^0, \dots, \overrightarrow{ID}^n)$ and define

$$Z(\Delta) = \{j : \left[\Delta_j^1, \cdots, \Delta_j^n\right] = 0^{1 \times n}\} \subseteq \{1, \dots, \ell^*\}.$$

So, $Z(\Delta) = \{j_1, \ldots, j_{\nu^*}\}$ is the set of indices j such that the j-th row of Δ has all zeros (i.e., all the identity components at level j are equal). For all $i \in \{1, \ldots, L\}$ we define the map:

$$\pi(i) = \begin{cases} k & \text{if } j_k \in Z(\Delta) \land i \le \ell^* \\ \nu^* + (i - \ell^*) & \text{if } i > \ell^* \\ \bot & \text{if } i \notin Z(\Delta) \land i \le \ell^* \end{cases}$$

 \mathcal{B} computes \widetilde{ID}^* by setting $\widetilde{ID}_i^* = ID_i^0$ for all $i = 1, \ldots, \ell^*$ such that $\pi(i) \neq \bot$, and executes the procedure $\widetilde{mpk} \leftarrow \mathbf{Initialize}(\widetilde{ID}^*)$ of the game IND-sHID-CPA. The received master public key is of the form $\widetilde{mpk} = (\widetilde{g}_1, \widetilde{g}_2, \widetilde{h}_1, \widetilde{u}_{0,1}, \widetilde{u}_{1,1}, \ldots, \widetilde{u}_{0,L}, \widetilde{u}_{1,L})$. \mathcal{B} constructs another master public key $mpk = (g_1, g_2, h_1, u_{0,1}, u_{1,1}, \ldots, u_{0,L}, u_{1,L})$ as follows:

$$g_1 = \tilde{g}_1, \ g_2 = \tilde{g}_2, \ h_1 = h_1$$

$$u_{b,i} = \begin{cases} \tilde{u}_{b,\pi(i)} & \text{if } \pi(i) \neq \bot \\ g_1^{\alpha_{b,i}} \text{ for } \alpha_{b,i} \stackrel{\$}{\leftarrow} \mathbb{Z}_p & \text{if } \pi(i) = \bot \end{cases}, \quad \forall i = 1, \dots, L, \ \forall b \in \{0,1\}$$

Finally, \mathcal{B} returns mpk to \mathcal{A} .

We give some hints about our technique, that may help the reader to follow our proof. Given in input the set of identities $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$, \mathcal{B} is going to play the selective-identity game by committing on the identity \overrightarrow{ID}^* which is equal to \overrightarrow{ID}^0 in those positions where all the identities are equal (i.e., $\pi(i) \neq \bot$), and eliminates the other positions. This requires the remapping that is formally defined by $\pi(\cdot)$. Then, when \mathcal{B} receives the master public key, it keeps in the new public key (that it returns to \mathcal{A}) only some elements. Precisely, it changes the position of all the elements $u_{b,i}$ of the given public key to some other positions (determined by $\pi(i)$), and then it creates new elements $u_{b,i}$ for those positions *i* where the identities $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$ are not all equal. In particular, for the latter elements notice that \mathcal{B} knows their discrete log in base g_1 .

Extract(\overrightarrow{ID}): Let \overrightarrow{ID} be the queried identity, and let $\ell = |\overrightarrow{ID}|$. \mathcal{B} defines the identity \widetilde{ID} by setting $\widetilde{ID}_{\pi(i)} = ID_i$ for all $1 \leq i \leq \ell$ such that $\pi(i) \neq \bot$ and asks for the secret key of \widetilde{ID} by executing $sk_{\widetilde{ID}} \leftarrow \mathbf{Extract}(\widetilde{ID})$. Let $sk_{\widetilde{ID}} = (\widetilde{d}_0, \widetilde{d}_1, \dots, \widetilde{d}_\nu)$ be the received secret key. \mathcal{B} computes:

$$d_{0} = \tilde{d}_{0} \cdot \left(\prod_{i=1,\pi(i)=\perp}^{\ell} u_{0,i} u_{1,i}^{ID_{i}}\right)^{r_{i}}, d_{i} = \begin{cases} \tilde{d}_{\pi(i)} & \text{if } \pi(i) \neq \perp \\ g_{1}^{r_{i}} & \text{if } \pi(i) = \perp \end{cases}, \ \forall i = 1, \dots, \ell$$

where $r_i \stackrel{s}{\leftarrow} \mathbb{Z}_p$. Finally, \mathcal{B} returns $(d_0, d_1, \ldots, d_\ell)$.

Before continuing the description of the simulation, we quickly pause to show that the returned secret key is valid, namely if $(\tilde{d}_0, \tilde{d}_1, \ldots, \tilde{d}_{\nu})$ is a valid secret key for the identity \widetilde{ID} , then $(d_0, d_1, \ldots, d_{\ell})$ is a valid secret key for \overrightarrow{ID} (under the HIBE system with master public key mpk). We also observe that \widetilde{ID} is a legitimate query for \mathcal{B} . Recall that by definition \overrightarrow{ID} falls into one of the following cases:

|*ID*| = ℓ* and *ID* ∉ Span(*ID*⁰,...,*ID*ⁿ)
 |*ID*| < ℓ* and *ID* is not an ancestor of any *ID*' ∈ Span(*ID*⁰,...,*ID*ⁿ)
 |*ID*| > ℓ*

If $|\overrightarrow{ID}| > \ell^*$, then $|\widetilde{ID}| > \nu^*$, and thus \widetilde{ID} is a legitimate query w.r.t. \widetilde{ID}^* .

If $|\overrightarrow{ID}| = \ell^*$ and $\overrightarrow{ID} \notin \mathsf{Span}(\overrightarrow{ID}^0, \dots, \overrightarrow{ID}^n)$, then notice that there exists $j \in Z(\Delta)$ such that $ID_j \neq ID_j^0$. So, we have that $\widetilde{ID}_{\pi(j)} = ID_j \neq ID_j^0 = \widetilde{ID}_{\pi(j)}^*$.

If $|\overrightarrow{ID}| < \ell^*$ and \overrightarrow{ID} is not an ancestor of any $\overrightarrow{ID'} \in \mathsf{Span}(\overrightarrow{ID^0}, \ldots, \overrightarrow{ID^n})$, then a similar argument as the one in the previous case applies. Indeed, notice that this case is equivalent to saying that any $\overrightarrow{ID'}$'s descendant of length ℓ^* is not in $\mathsf{Span}(\overrightarrow{ID^0}, \ldots, \overrightarrow{ID^n})$. In particular, this must hold even for $\overrightarrow{ID'}$ defined such that $ID'_i = ID_i$ for all $1 \le i \le \ell$, and $ID'_i = ID^0_i$ for $\ell < i \le \ell^*$. Recall that $\overrightarrow{ID'}$ is a descendant of \overrightarrow{ID} and it holds $\overrightarrow{ID'} \notin \mathsf{Span}(\overrightarrow{ID^0}, \ldots, \overrightarrow{ID^n})$. This means that there exists $j \in Z(\Delta)$, such that $ID'_j \ne ID^0_j$. However, by definition of $\overrightarrow{ID'}$, this must hold for $j \le \ell < \ell^*$. So, as in the previous case, we have that $\overrightarrow{ID}_{\pi(j)} \ne \widetilde{ID}^*_{\pi(j)}$.

LR (m_0, m_1) : \mathcal{B} executes the procedure $\tilde{C}^* \leftarrow \mathbf{LR}(m_0, m_1)$ of the game IND-sHID-CPA. Let $\tilde{C}^* = (\tilde{C}_1^*, \tilde{C}_{2,1}^*, \dots, \tilde{C}_{2,\nu^*}^*, \tilde{C}_3^*)$ be an encryption of m_β (for some $\beta \in \{0, 1\}$) for the identity \widetilde{ID}^* . \mathcal{B} computes and outputs the ciphertexts (C^0, \dots, C^n) as it is described below.

For all i = 0 to n, \mathcal{B} sets:

$$C_{1}^{i} = \tilde{C}_{1}^{*}, \ C_{3}^{i} = \tilde{C}_{3}^{*}$$

$$C_{2,j}^{i} = \begin{cases} \tilde{C}_{2,\pi(j)}^{*} & \text{if } i \in Z(\Delta) \\ (\tilde{C}_{1}^{*})^{\alpha_{0,j} + ID_{j}^{i} \cdot \alpha_{1,j}} & \text{if } i \notin Z(\Delta) \end{cases}, \ \forall j = 1, \dots, \ell^{*}$$

By looking at the definition of $\pi(i)$ and $Z(\Delta)$, one can see that the produced ciphertexts follow the correct distribution.

Finalize(β'): Let β' be the bit received by \mathcal{A} . \mathcal{B} concludes its simulation by executing **Finalize**(β').

Since \mathcal{B} can perfectly simulate the game IND-sCR-CPA to the adversary \mathcal{A} , we have:

$$\mathbf{Adv}_{BB-HIBE}^{\mathrm{IND}\text{-}\mathrm{sHID}\text{-}\mathrm{CPA}}(\mathcal{B}) = \mathbf{Adv}_{BB-HIBE}^{\mathrm{IND}\text{-}\mathrm{sCR}\text{-}\mathrm{CPA}}(\mathcal{A})$$

which concludes the proof.

ABOUT THE ORIGINAL BB SCHEME. As we mentioned at the beginning of this section, we proved the IND-sCR-CPA security of a variant of the original BB scheme. The reason of this is that the scheme proposed in [8] is not secure under correlated randomness. This is not an issue of the proof. As we show below, the scheme completely breaks when many encryptions with the same randomness and only two different identities are released.

To see this, we recall that the original scheme is the same as that given in Figure 11 except that the public key contains only one element u_1 in common for all the levels (instead of many $u_{i,1}$).

Now, assume that one receives two ciphertexts generated with the same randomness r and for two identities (ID_1^0, ID_2^0) , (ID_1^1, ID_2^1) such that $ID_1^0 = ID_1^1$ and $ID_2^0 \neq ID_2^1$. Then, one can recover $u_1^{r.c} = C_{2,2}^0/C_{2,2}^1$ (where $c = ID_2^0 - ID_2^1$). This value then allows to generate ciphertexts for any identity, i.e., for the pattern P = (*, *). So, in particular, an adversary in the IND-sCR-CPA game can ask the challenge ciphertexts for the identities $(\overrightarrow{ID}^0, \overrightarrow{ID}^1)$ mentioned before, then it generates a ciphertext for some $\overrightarrow{ID}' \notin \text{Span}(\overrightarrow{ID}^0, \overrightarrow{ID}^1)$, and finall ask for the secret key of \overrightarrow{ID}' (this is a legitimate query), which allows to decrypt and recover the message.

B.2 The case of the Boneh-Boyen-Goh HIBE [10]

In this section we consider the HIBE scheme proposed by Boneh, Boyen and Goh in [10], and we show that it satisfies Property 1 and that it is IND-sCR-CPA-secure w.r.t. \mathcal{R}_{WIBE} . We recall the scheme in Figure 12. The scheme has identity space $I\mathcal{D} = \mathbb{Z}_p^*$ (i.e., 0 is not a valid identity) where p is a prime of suitable length.

Setup:	$KeyDer(d_{(ID_1,\ldots,ID_\ell)}, ID_{\ell+1}):$
$g_1, g_2 \stackrel{\$}{\leftarrow} \mathbb{G} ; \alpha \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ $h_1 \leftarrow g_1^{\alpha} ; h_2 \leftarrow g_2^{\alpha}$ $u_i \stackrel{\$}{\leftarrow} \mathbb{G} \text{ for } i = 1, \dots, L$ $mpk \leftarrow (g_1, g_2, h_1, u_0, \dots, u_L)$ $d_0 \leftarrow h_2$ For $i = 1, \dots, L + 1$ do $d_i \leftarrow 1$ $msk \leftarrow (d_0, d_1, \dots, d_L, d_{L+1})$ Return (mpk, msk)	Parse $d_{(ID_1,,ID_{\ell})}$ as $(d_0, d_{\ell+1},, d_L, d_{L+1})$ $r_{\ell+1} \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ $d'_0 \leftarrow d_0 \cdot d_{\ell+1}^{ID_{\ell+1}} \cdot (u_0 \prod_{i=1}^{\ell} u_i^{ID_i})^{r_{\ell+1}}$ For $i = \ell + 2,, L$ do $d'_i \leftarrow d_i \cdot u_i^{r_{\ell+1}}$ $d'_{L+1} \leftarrow d_{L+1} \cdot g_1^{r_{\ell+1}}$ Return $(d'_0, d'_{\ell+2},, d'_L, d'_{L+1})$
Enc $(mpk, \overrightarrow{ID}, m)$: Parse \overrightarrow{ID} as (ID_1, \dots, ID_ℓ) $r \stackrel{s}{\leftarrow} \mathbb{Z}_p$; $C_1 \leftarrow g_1^r$ $C_2 \leftarrow (u_0 \prod_{i=1}^{\ell} u_i^{ID_i})^r$ $C_3 \leftarrow m \cdot \hat{e}(h_1, g_2)^r$ Return (C_1, C_2, C_3)	$\begin{aligned} Dec(d_{(ID_1,\ldots,ID_\ell)},C):\\ & \text{Parse } d_{(ID_1,\ldots,ID_\ell)} \text{ as } (d_0,d_{\ell+1},\ldots,d_{L+1})\\ & \text{Parse } C \text{ as } (C_1,C_2,C_3)\\ & m' \leftarrow C_3 \cdot \frac{\hat{e}(C_2,d_{L+1})}{\hat{e}(C_1,d_0)}\\ & \text{Return } m' \end{aligned}$

Figure 12: The Boneh-Boyen-Goh HIBE scheme.

To show that the BBG-HIBE scheme is IND-sCR-CPA-secure we first show that it satisfies Property 1 by describing the following algorithm for the ciphertext conversion. For ease of exposition, we give our description using identities that may take value 0, even though 0 is not a valid identity value. However, following Remark 4.2, everything can be defined by choosing any two other values of \mathbb{Z}_p^* , e.g., 1 and 2, instead of 0 and 1.

BBG.Convert $(mpk, C^0, \overrightarrow{ID}^0, \ldots, C^n, \overrightarrow{ID}^n, \overrightarrow{ID})$ Assume that $|\overrightarrow{ID}^0| = \ldots = |\overrightarrow{ID}^n| = \ell$ and $|\overrightarrow{ID}| = \ell' \leq \ell$. Recall that each C^i has form (C_1^i, C_2^i, C_3^i) . First, consider the case when $\ell' = \ell$. The algorithm works as follows.

First, set $C_1 = C_1^0$ and $C_3 = C_3^0$. Find $\vec{k} \in \mathbb{Z}^n$ such that $\Delta \vec{k} = (\overrightarrow{ID} - \overrightarrow{ID}^0)$, and compute

$$\tilde{\Delta} = \left[C_2^0 / C_2^1 || C_2^0 / C_2^2 || \cdots || C_2^0 / C_2^n \right] = \left[\tilde{\Delta}^{(1)} || \cdots || \tilde{\Delta}^{(n)} \right] \in \mathbb{G}^{1 \times r}$$

Finally, set $C_2 = C_2^0 \cdot \prod_{i=1}^n (\tilde{\Delta}^{(i)})^{k_i}$, and output $C = (C_1, C_2, C_3)$.

If $\ell' < \ell$, then one can first pad \overrightarrow{ID} to get \overrightarrow{ID}' of length ℓ by setting $ID'_i = 0$ for $\ell' < i \leq \ell$, and then use the above procedure. It is not hard to see that a ciphertext for such \overrightarrow{ID}' is a valid ciphertext for \overrightarrow{ID} as well.

The correctness of the algorithm can be verified by inspection.

We prove its security via the following theorem.

Theorem B.2 If there exists an adversary \mathcal{A} that has IND-sCR-CPA-advantage $\geq \epsilon$ against the BBG-HIBE scheme w.r.t. \mathcal{R}_{WIBE} , then there exists an adversary \mathcal{B} that has IND-sHID-CPA-advantage ϵ against the same scheme BBG-HIBE. Namely:

$$\mathbf{Adv}_{BBG-HIBE}^{\mathrm{IND}\text{-}\mathrm{sHID}\text{-}\mathrm{CPA}}(\mathcal{B}) = \mathbf{Adv}_{BBG-HIBE}^{\mathrm{IND}\text{-}\mathrm{sCR}\text{-}\mathrm{CPA}}(\mathcal{A})$$

Proof: We make the proof by describing an adversary \mathcal{B} that plays the game IND-sHID-CPA against the scheme BBG-HIBE by simulating the game IND-sCR-CPA to the adversary \mathcal{A} . The main idea of the proof is very similar to the one used in the proof of Theorem B.1. Namely, our simulator is going to play a selective-identity game by declaring an identity \widetilde{ID}^* which is equal to \overline{ID}^0 in all positions where all the identities are equal. Then for these positions, the corresponding elements of the public key are kept in the new public key (but relocated), and new elements, for which \mathcal{B} knows the discrete log, are introduced. This knowledge, basically, allows the simulator to convert the secret keys and the challenge ciphertext received by its challenger into the ones to give to \mathcal{A} .

Initialize $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$: \mathcal{B} takes as input a set of identities $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n) \in \mathcal{R}_{WIBE}$ such that $|\overrightarrow{ID}^i| = \ell^*$ for all $0 \le i \le n$. Let $\Delta = \Delta(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$ and define

$$Z(\Delta) = \{j : \left[\Delta_j^1, \cdots, \Delta_j^n\right] = 0^{1 \times n}\} \subseteq \{1, \dots, \ell^*\}.$$

So, $Z(\Delta) = \{j_1, \ldots, j_{\nu^*}\}$ is the set of indices j such that the j-th row of Δ has all zeros. For all $i \in \{1, \ldots, L\}$ we define the map:

$$\pi(i) = \begin{cases} k & \text{if } j_k \in Z(\Delta) \land i \le \ell^* \\ \nu^* + (i - \ell^*) & \text{if } i > \ell^* \\ \bot & \text{if } i \notin Z(\Delta) \land i \le \ell^* \end{cases}$$

 \mathcal{B} computes \widetilde{ID}^* by setting $\widetilde{ID}_i^* = ID_i^0$ for all $i = 1, \ldots, \ell^*$ such that $\pi(i) \neq \bot$, and executes the procedure $\widetilde{mpk} \leftarrow \operatorname{Initialize}(\widetilde{ID}^*)$ of the game IND-sHID-CPA. The received master public key is of the form $\widetilde{mpk} = (\widetilde{g}_1, \widetilde{g}_2, \widetilde{h}_1, \widetilde{u}_0, \widetilde{u}_1, \ldots, \widetilde{u}_L)$. \mathcal{B} constructs a master public key $mpk = (g_1, g_2, h_1, u_0, u_1, \ldots, u_L)$ as follows:

$$g_1 = \tilde{g}_1, \ g_2 = \tilde{g}_2, \ h_1 = h_1, \ u_0 = \tilde{u}_0$$
$$u_i = \begin{cases} \tilde{u}_{\pi(i)} & \text{if } \pi(i) \neq \bot \\ g_1^{\alpha_i} \text{ for } \alpha_i \stackrel{\$}{\leftarrow} \mathbb{Z}_p & \text{if } \pi(i) = \bot \end{cases}, \ \forall i = 1, \dots, L$$

Finally, \mathcal{B} returns mpk to \mathcal{A} .

Extract (\overrightarrow{ID}) : Let \overrightarrow{ID} be the identity asked by the adversary, and let $\ell = |\overrightarrow{ID}|$. The simulator defines the identity \widetilde{ID} by setting $\widetilde{ID}_{\pi(i)} = ID_i$ for all $1 \leq i \leq \ell$ such that $\pi(i) \neq \bot$ and asks for the secret key of \widetilde{ID} by executing $sk_{\widetilde{ID}} \leftarrow \mathbf{Extract}(\widetilde{ID})$. Let $sk_{\widetilde{ID}} = (\widetilde{d}_0, \widetilde{d}_{\nu+1}, \ldots, \widetilde{d}_{L+1})$ be the received secret key. \mathcal{B} picks a random $r' \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ and computes:

$$d_{0} = \tilde{d}_{0} \cdot \left(u_{0} \prod_{i=1}^{\ell} u_{i}^{ID_{i}} \right)^{r'} \cdot (\tilde{d}_{L+1})^{\sum_{i=1,\pi(i)=\perp}^{\ell} \alpha_{i}ID_{i}}, \quad d_{L+1} = \tilde{d}_{L+1} \cdot g_{1}^{r'}$$
$$d_{i} = \begin{cases} \tilde{d}_{\pi(i)} \cdot u_{\pi(i)}^{r'} & \text{if } \pi(i) \neq \bot \\ (\tilde{d}_{L+1} \cdot g_{1}^{r'})^{\alpha_{i}} & \text{if } \pi(i) = \bot \end{cases}, \quad \forall i = \nu + 1, \dots, L$$

Finally, \mathcal{B} returns $(d_0, d_{\nu+1}, \ldots, d_{L+1})$ to \mathcal{A} .

Before describing the rest of the proof we show that if $(\tilde{d}_0, \tilde{d}_{\nu+1}, \ldots, \tilde{d}_{L+1})$ is a valid secret key for the identity \widetilde{ID} , then $(d_0, d_{\nu+1}, \ldots, d_{L+1})$ is a valid secret key for \overrightarrow{ID} (under the HIBE system with master public key mpk). Assume that

$$\tilde{d}_0 = \tilde{h}_2 \left(\tilde{u}_0 \prod_{i=1}^{\nu} \tilde{u}_i^{\widetilde{ID}_i} \right)', \quad \tilde{d}_{L+1} = \tilde{g}_1^r, \quad \tilde{d}_i = u_i^r$$

for some randomness $r \in \mathbb{Z}_p$. Since we can write $\tilde{d}_0 = h_2 \left(u_0 \prod_{i=1,\pi(i) \neq \perp}^{\ell} u_i^{ID_i} \right)^r$, we have

$$\begin{aligned} d_0 &= h_2 \left(u_0 \prod_{i=1,\pi(i)\neq\perp}^{\ell} u_i^{ID_i} \right)^r \cdot \left(u_0 \prod_{i=1}^{\ell} u_i^{ID_i} \right)^{r'} \cdot (\tilde{d}_{L+1})^{\sum_{i=1,\pi(i)=\perp}^{\ell} \alpha_i ID_i} \\ &= h_2 \left(u_0 \prod_{i=1,\pi(i)\neq\perp}^{\ell} u_i^{ID_i} \right)^{r+r'} \cdot \left(\prod_{i=1,\pi(i)=\perp}^{\ell} u_i^{ID_i} \right)^{r'} \cdot \left(\prod_{i=1,\pi(i)=\perp}^{\ell} u_i^{ID_i} \right)^r \\ &= h_2 \left(u_0 \prod_{i=1}^{\ell} u_i^{ID_i} \right)^{r+r'} \end{aligned}$$

which is a valid secret key element for the identity \overrightarrow{ID} and randomness r + r'. Furthermore, as one can observe, $d_{L+1} = g_1^{r+r'}$ and $d_i = u_i^{r+r'}$ that are also valid elements for randomness r + r'.

Finally, following the same argument in the proof of Theorem B.1 we can see that ID is a legitimate query for \mathcal{B} .

LR (m_0, m_1) : \mathcal{B} executes the procedure $\tilde{C}^* \leftarrow \mathbf{LR}(m_0, m_1)$ of the game IND-sHID-CPA. Let $\tilde{C}^* = (\tilde{C}_1^*, \tilde{C}_2^*, \tilde{C}_3^*)$ be an encryption of m_β (for some $\beta \in \{0, 1\}$) for the identity \widetilde{ID}^* .

 \mathcal{B} computes and outputs the ciphertexts (C^0, \ldots, C^n) as it is described below.

For all i = 0 to n, \mathcal{B}_2 sets:

$$C_1^i = \tilde{C}_1^*, \ C_3^i = \tilde{C}_3^*, \ C_2^i = \tilde{C}_2^* \cdot (\tilde{C}_1^*)^{\sum_{j=1,\pi(j)=\perp}^{\ell^*} \alpha_j I D_j^i}$$

To see that the computed ciphertexts are correct, assume s be the randomness used to create \tilde{C}^* . So, we have: $\tilde{C}^*_1 = \tilde{g}^s_1$ and

$$\tilde{C}_2^* = \left(\tilde{u}_0 \prod_{j=1}^{\nu^*} \tilde{u}_j^{\widetilde{ID}_j^*}\right)^s = \left(u_0 \prod_{j=1,\pi(j)\neq \perp}^{\ell^*} u_j^{ID_j^0}\right)^s$$

Thus $\forall i = 0, \ldots, n$ we obtain:

$$C_{2}^{i} = \tilde{C}_{2}^{*} \cdot \left(\prod_{j=1,\pi(j)=\perp}^{\ell^{*}} u_{j}^{ID_{j}^{i}}\right)^{s} = \left(u_{0}\prod_{j=1,\pi(j)\neq\perp}^{\ell^{*}} u_{j}^{ID_{j}^{0}}\right)^{s} \cdot \left(\prod_{j=1,\pi(j)=\perp}^{\ell^{*}} u_{j}^{ID_{j}^{i}}\right)^{s}$$

Since the target identities are distributed according to \mathcal{R}_{WIBE} , we can see that the elements C_2^i are distributed correctly as well.

Finalize(β'): Let β' be the bit received by \mathcal{A} . \mathcal{B} concludes its simulation by executing **Finalize**(β').

Since \mathcal{B} can perfectly simulate the game IND-sCR-CPA to the adversary \mathcal{A} , we have:

$$\mathbf{Adv}_{BBG-HIBE}^{\mathrm{IND-sHID-CPA}}(\mathcal{B}) \geq \mathbf{Adv}_{BBG-HIBE}^{\mathrm{IND-sCR-CPA}}(\mathcal{A})$$

which concludes the proof.

B.3 The case of the Waters HIBE [35]

In this section we show that the HIBE scheme proposed by Waters in [35] has Property 1 and it is IND-sCR-CPA-secure. We recall the scheme in Figure 13

Setup:

 $\begin{array}{l} \overset{\cdot}{g_1,g_2} \stackrel{\ast}{\leftarrow} \mathbb{G} \ ; \ \alpha \stackrel{\ast}{\leftarrow} \mathbb{Z}_p \\ h_1 \leftarrow g_1^{\alpha} \ ; \ h_2 \leftarrow g_2^{\alpha} \\ u_{i,j} \stackrel{\ast}{\leftarrow} \mathbb{G} \ \text{for} \ i = 1, \ldots, L; j = 0 \ldots n \\ mpk \leftarrow (g_1,g_2,h_1,u_{1,0},\ldots,u_{L,n}) \\ msk \leftarrow h_2 \\ \text{Return} \ (mpk,msk) \end{array}$

 $\begin{aligned} \mathsf{KeyDer}(d_{(ID_1,\ldots,ID_\ell)}, ID_{\ell+1}): \\ & \text{Parse } d_{(ID_1,\ldots,ID_\ell)} \text{ as } (d_0,\ldots,d_\ell) \\ & r_{\ell+1} \stackrel{\$}{\leftarrow} \mathbb{Z}_p \\ & d'_0 \leftarrow d_0 \cdot F_{\ell+1} (ID_{\ell+1})^{r_{\ell+1}} \\ & d'_{\ell+1} \leftarrow g_1^{r_{\ell+1}} \\ & \text{Return } (d'_0, d_1,\ldots,d_\ell, d'_{\ell+1}) \end{aligned}$

 $\begin{array}{lll} \operatorname{Enc}(mpk, ID, m): & \operatorname{Dec}(d_{(ID_1, \dots, ID_{\ell})}, C): \\ \operatorname{Parse} ID \text{ as } (ID_1, \dots, ID_{\ell}) & \operatorname{Parse} d_{(ID_1, \dots, ID_{\ell})} \text{ as } (d_0, \dots, d_{\ell}) \\ \operatorname{Parse} i = 1 \dots \ell \text{ do} & \operatorname{Parse} C \text{ as } (C_1, C_{2,1}, \dots, C_{2,\ell}, C_3) \\ \operatorname{For} i = 1 \dots \ell \text{ do} & m' \leftarrow C_3 \cdot \frac{\prod_{i=1}^{\ell} \hat{e}(d_i, C_{2,i})}{\hat{e}(C_1, d_0)} \\ \operatorname{Return} m' & \operatorname{Return} m' \end{array}$

Figure 13: The Waters HIBE scheme.

The scheme has identity space \mathbb{Z}_2^{λ} where λ is sufficiently long so that one can hash identities into strings of length λ avoiding collisions. We first show that the scheme satisfies Property 1 by describing the following algorithm:

Waters.Convert $(mpk, C^0, \overrightarrow{ID}^0, \ldots, C^n, \overrightarrow{ID}^n, \overrightarrow{ID})$ Assume that $|\overrightarrow{ID}^0| = \ldots = |\overrightarrow{ID}^n| = \ell$ and $|\overrightarrow{ID}| = \ell' \leq \ell$. Recall that each C^i has form $(C_1^i, \{C_{2,j}^i\}_{j=1}^\ell, C_3^i)$. First, consider the case when $\ell' = \ell$. The algorithm works as follows.

Set $C_1 = C_1^0$ and $C_3 = C_3^0$.

Find $\vec{k} \in \mathbb{Z}^n$ such that $\Delta \vec{k} = (\vec{ID} - \vec{ID}^0)$, and compute

$$\tilde{\Delta} = \begin{pmatrix} C_{2,1}^0 / C_{2,1}^1 & C_{2,1}^0 / C_{2,1}^2 & \cdots & C_{2,1}^0 / C_{2,1}^n \\ \vdots & \vdots & & \vdots \\ C_{2,\ell}^0 / C_{2,\ell}^1 & C_{2,\ell}^0 / C_{2,\ell}^2 & \cdots & C_{2,\ell}^0 / C_{2,\ell}^n \end{pmatrix} = \left[\tilde{\Delta}^{(1)} || \cdots || \tilde{\Delta}^{(n)} \right] \in \mathbb{G}^{\ell \times n}$$

Finally, for all j = 1 to ℓ , set $C_{2,j} = C_{2,j}^0 \cdot \prod_{i=1}^n (\tilde{\Delta}^{(i)})^{k_i}$, and output $C = (C_1, \{C_{2,j}\}_{j=1}^\ell, C_3)$.

If $\ell' < \ell$, then one can first pad \overrightarrow{ID} to get $\overrightarrow{ID'}$ of length ℓ (e.g., by setting $ID'_i = ID^0_i$ for $\ell' < i \leq \ell$), and then use the above procedure to generate a ciphertext C' for $\overrightarrow{ID'}$. Finally, a valid ciphertext for \overrightarrow{ID} can be obtained by removing the elements $C_{2,j}$ for $j > \ell'$.

The algorithm's correctness can be verified by inspection.

We prove the security via the following theorem.

Theorem B.3 If there exists an adversary \mathcal{A} that has IND-sCR-CPA-advantage $\geq \epsilon$ against the Wat-HIBE scheme w.r.t. \mathcal{R}_{WIBE} , then there exists an adversary \mathcal{B} that has IND-sHID-CPA-advantage ϵ against the same scheme Wat-HIBE. Namely:

$$\mathbf{Adv}_{Wat-HIBE}^{\mathrm{IND-sHID-CPA}}(\mathcal{B}) = \mathbf{Adv}_{Wat-HIBE}^{\mathrm{IND-sCR-CPA}}(\mathcal{A})$$

Proof: We prove the theorem by describing the adversary \mathcal{B} that plays the game IND-sHID-CPA against the scheme by simulating the game IND-sCR-CPA to the adversary \mathcal{A} . Again, the idea behind the techniques used in the proof is very similar to what we did in the proofs of Theorem B.1 and Theorem B.2.

Initialize $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$: \mathcal{B} takes as input a set of identities $(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n) \in \mathcal{R}_{WIBE}$ such that $\overrightarrow{ID}^i \in \mathbb{Z}_2^{\lambda \ell^*}$ for all $0 \leq i \leq n$. Let $\Delta = \Delta(\overrightarrow{ID}^0, \ldots, \overrightarrow{ID}^n)$ and define

$$Z(\Delta) = \{j : \left[\Delta^{1}_{[j,j\lambda-1]}, \cdots, \Delta^{n}_{[j,j\lambda-1]}\right] = 0^{\lambda \times n}\} \subseteq \{1, \dots, \ell^*\}$$

where $\Delta_{[j,k]}^i$ denotes that we consider vector Δ^i restricted to rows from j to k. So, $Z(\Delta) = \{j_1, \ldots, j_{\nu^*}\}$ is the set of levels j such that all the identities agree at that level. For all $i \in \{1, \ldots, L\}$ we define the following map:

$$\pi(i) = \begin{cases} k & \text{if } j_k \in Z(\Delta) \land i \le \ell^* \\ \nu^* + (i - \ell^*) & \text{if } i > \ell^* \\ \bot & \text{if } i \notin Z(\Delta) \land i \le \ell^* \end{cases}$$

 \mathcal{B} computes \widetilde{ID}^* by setting $\widetilde{ID}_i^* = ID_i^0$ for all $i = 1, \ldots, \ell^*$ such that $\pi(i) \neq \bot$, and executes the procedure $\widetilde{mpk} \leftarrow \mathbf{Initialize}(\widetilde{ID}^*)$ of the game IND-sHID-CPA. The received master public key is of the form $\widetilde{mpk} = (\widetilde{g}_1, \widetilde{g}_2, \widetilde{h}_1, \widetilde{u}_{0,1}, \widetilde{u}_{1,1}, \ldots, \widetilde{u}_{0,L}, \widetilde{u}_{1,L})$. \mathcal{B} constructs another master public key $mpk = (g_1, g_2, h_1, \{u_{i,j}\}_{i=1,\ldots,L,j=0,\ldots,\lambda})$ as follows:

$$g_1 = \tilde{g}_1, \ g_2 = \tilde{g}_2, \ h_1 = \tilde{h}_1$$

$$u_{i,j} = \begin{cases} \tilde{u}_{\pi(i),j} & \text{if } \pi(i) \neq \bot \\ g_1^{\alpha_{i,j}} & \text{for } \alpha_{i,j} \stackrel{\$}{\leftarrow} \mathbb{Z}_p & \text{if } \pi(i) = \bot \end{cases}, \quad \forall i = 1, \dots, L, \; \forall j = 1, \dots, \lambda$$

Finally, \mathcal{B} returns mpk to \mathcal{A} .

Extract(\overrightarrow{ID}): Let \overrightarrow{ID} be the queried identity, and let $\ell = |\overrightarrow{ID}|$. \mathcal{B} defines the identity \widetilde{ID} by setting $\widetilde{ID}_{\pi(i)} = ID_i$ for all $1 \leq i \leq \ell$ such that $\pi(i) \neq \bot$ and asks for the secret key of \widetilde{ID} by executing $sk_{\widetilde{ID}} \leftarrow \mathbf{Extract}(\widetilde{ID})$. Let $sk_{\widetilde{ID}} = (\widetilde{d}_0, \widetilde{d}_1, \dots, \widetilde{d}_{\nu})$ be the received secret key. \mathcal{B} computes:

$$d_0 = \tilde{d}_0 \cdot \left(\prod_{i=1,\pi(i)=\perp}^{\ell} \left(u_{i,0} \prod_{j=1}^{\lambda} u_{i,j}^{ID_{i,j}}\right)^{r_i}\right), d_i = \begin{cases} \tilde{d}_{\pi(i)} & \text{if } \pi(i) \neq \perp \\ g_1^{r_i} & \text{if } \pi(i) = \perp \end{cases}, \ \forall i = 1, \dots, \ell$$

where $r_i \stackrel{\$}{\leftarrow} \mathbb{Z}_p$. Finally, \mathcal{B} returns $(d_0, d_1, \ldots, d_\ell)$ to \mathcal{A} .

It is not hard to see that if $(\tilde{d}_0, \tilde{d}_1, \ldots, \tilde{d}_\nu)$ is a valid secret key for the identity \widetilde{ID} , then $(d_0, d_1, \ldots, d_\ell)$ is a valid secret key for \overrightarrow{ID} (under the HIBE system with master public key mpk). Moreover, following the same argument showed in the proof of Theorem B.1, one can see that \widetilde{ID} is a legitimate query for \mathcal{B} .

LR (m_0, m_1) : \mathcal{B} executes the procedure $\tilde{C}^* \leftarrow \mathbf{LR}(m_0, m_1)$ of the game IND-sHID-CPA. Let $\tilde{C}^* = (\tilde{C}_1^*, \tilde{C}_{2,1}^*, \dots, \tilde{C}_{2,\nu^*}^*, \tilde{C}_3^*)$ be an encryption of m_β (for some $\beta \in \{0, 1\}$) for the identity \widetilde{ID}^* . \mathcal{B} outputs the ciphertexts (C^0, \dots, C^n) that are computed as follows. For all i = 0 to n, \mathcal{B}_2 sets:

$$C_{1}^{i} = \tilde{C}_{1}^{*}, \ C_{3}^{i} = \tilde{C}_{3}^{*}$$

$$C_{2,j}^{i} = \begin{cases} \tilde{C}_{2,\pi(j)}^{*} & \text{if } i \in Z(\Delta) \\ (\tilde{C}_{1}^{*})^{\alpha_{0,j} + \sum_{l=1}^{\lambda} \alpha_{j,l} \cdot ID_{j,l}^{i}} & \text{if } i \notin Z(\Delta) \end{cases}, \ \forall j = 1, \dots, \ell^{*}$$

Finalize(β'): Let β' be the bit received by \mathcal{A} . \mathcal{B} concludes its simulation by executing **Finalize**(β').

Since \mathcal{B} can perfectly simulate the game IND-sCR-CPA to the adversary \mathcal{A} , then we have:

$$\mathbf{Adv}_{Wat-HIBE}^{IND-sID-CPA}(\mathcal{B}) = \mathbf{Adv}_{Wat-HIBE}^{\mathrm{IND-sCR-CPA}}(\mathcal{A})$$

which concludes the proof.