An extended abstract of this paper appeared in Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *33rd International Colloquium on Automata, Languages and Programming – ICALP 2006*, volume 4052 of *Lecture Notes in Computer Science*, Springer-Verlag, 2006 [ACD+06]. This is the full version.

Identity-Based Encryption Gone Wild

Michel Abdalla[1], Dario Catalano[1], Alexander W. Dent[2],
John Malone-Lee[3], Gregory Neven[1,4], Nigel P. Smart[3].

December 3, 2006

[1] Département d'Informatique, Ecole Normale Supérieure,
45 rue d'Ulm, 75230 Paris Cedex 05, France.
Email: {Michel.Abdalla,Dario.Catalano,Gregory.Neven}@ens.fr
[2] Information Security Group,
Royal Holloway, University of London,
Egham, Surrey, TW20 0EX, United Kingdom.
Email: a.dent@rhul.ac.uk
[3] Department of Computer Science, University of Bristol,
Woodland Road, Bristol, BS8 1UB, United Kingdom.
Email: {malone,nigel}@cs.bris.ac.uk
[4] Department of Electrical Engineering, Katholieke Universiteit Leuven,
Kasteelpark Arenberg 10, B-3001 Heverlee, Belgium.
Email: Gregory.Neven@esat.kuleuven.be

Abstract

In this paper we introduce a new primitive called identity-based encryption with wildcards, or WIBE for short. It allows to encrypt messages to a whole range of users simultaneously whose identities match a certain pattern. This pattern is defined through a sequence of fixed strings and wildcards, where any string can take the place of a wildcard in a matching identity. Our primitive can be applied to provide an intuitive way to send encrypted email to groups of users in a corporate hierarchy. We propose a full security notion and give efficient implementations meeting this notion under different pairing-related assumptions, both in the random oracle model and in the standard model.

Keywords: Identity-based encryption, provable security.

Contents

1 Introduction

The concept of identity based cryptography was introduced by Shamir as early as in 1984 [Sha85]. However, it took nearly twenty years for an efficient identity based encryption (IBE) scheme to be proposed. In 2000 and 2001 respectively Sakai, Ohgishi and Kasahara [SOK00] and Boneh and Franklin [BF03] proposed IBE schemes based on elliptic curve pairings. Also, in 2001 Cocks proposed a system based on the quadratic residuosity problem [Coc01].

One of the main application areas proposed for IBE is that of email encryption. In this scenario, given an email address, one can encrypt a message to the owner of the email address without needing to obtain an authentic copy of the owner's public key first. In order to decrypt the email the recipient must authenticate itself to a trusted authority who generates a private key corresponding to the email address used to encrypt the message.

Identity-based encryption with wildcards. Our work is motivated by the fact that many email addresses correspond to groups of users rather than single individuals. Consider the scenario where there is some kind of organisational hierarchy. Take as an example an organisation called ECRYPT which is divided into virtual labs, say AZTEC and STVL. In addition, these virtual labs are further subdivided into working groups WG1, WG2 and WG3. Finally, each working group may consist of many individual members. There are several extensions of the IBE primitive to such a hierarchical setting (HIBE) [HL02, GS02]. The idea is that each level can issue keys to users on the level below. For example the owner of the ECRYPT key can issue decryption keys for ECRYPT.AZTEC and ECRYPT.STVL.

Suppose that we wish to send an email to all the members of the AZTEC.WG1 working group, which includes personal addresses ECRYPT.AZTEC.WG1.Nigel, ECRYPT.AZTEC.WG1.Dario and ECRYPT.AZTEC.WG1.John. Given a standard HIBE one would have to encrypt the message to each user individually. To address this limitation we introduce the concept of *identity based encryption with wildcards* (WIBE). The way in which decryption keys are issued is exactly as in a standard HIBE scheme; what differs is encryption. Our primitive allows the encrypter to replace any component of the recipient identity with a *wildcard* so that any identity matching the *pattern* can decrypt. Denoting wildcards by `*`, in the example above the encrypter would use the identity ECRYPT.AZTEC.WG1.`*` to encrypt to all members of the AZTEC.WG1 group.

It is often suggested that identity strings should be appended with the date so as to add timeliness to the message, and so try to mitigate the problems associated with key revocation. Using our technique we can now encrypt to a group of users, with a particular date, by encrypting to an identity of the form ECRYPT.AZTEC.WG1.`*`.22Oct2006 for example. Thus any individual in ECRYPT.AZTEC.WG1 with a decryption key for 22nd October 2006 will be able to decrypt.

As another example, take a hierarchy of email addresses at academic institutions of the form `name@department.university.edu`, i.e., the email address of John Smith working at the computer science department of Some State University would be `johnsmith@cs.ssu.edu`. Using our primitive, one can send encrypted email to everyone in the computer science department at SSU by encrypting to identity `*@cs.ssu.edu`, to everyone at SSU by encrypting to `*@*.ssu.edu`, to all computer scientists at any institution by encrypting to `*@cs.*.edu`, or to all system administrators in the university by encrypting to `sysadmin@*.ssu.edu`.

Our contributions. In this paper, we introduce the primitive of identity-based encryption with wildcards, define appropriate security notions under chosen-plaintext and chosen-ciphertext attack, and present the first instantiations of this primitive. An overview of our schemes is given in Figure 1.

We first show a generic construction from any HIBE scheme which has the disadvantage that the size of the secret key of a user on level ℓ in the hierarchy is exponential in ℓ. The scheme clarifies the relationship between both primitives and motivates our search for direct schemes with efficiency

Scheme	$\|mpk\|$	$\|d\|$	$\|C\|$	Dec	Assumption	RO
Generic	$\|mpk_{\mathcal{HIBE}}\|$	$2^L \cdot \|d_{\mathcal{HIBE}}\|$	$\|C_{\mathcal{HIBE}}\|$	$\mathsf{Dec}_{\mathcal{HIBE}}$	$\mathcal{HIBE}$ is IND-ID-CPA	No
$\mathcal{Wa\text{-}WIBE}$	$(n+1)L+3$	$L+1$	$(n+1)L+2$	$L+1$	BDDH	No
$\mathcal{BB\text{-}WIBE}$	$2L+3$	$L+1$	$2L+2$	$L+1$	BDDH	Yes
$\mathcal{BBG\text{-}WIBE}$	$L+4$	$L+2$	$L+3$	2	L-BDHI	Yes

Figure 1: Efficiency and security comparison between the generic scheme of Section 4, the $\mathcal{Wa\text{-}WIBE}$ scheme of Section 5.2, and the $\mathcal{BB\text{-}WIBE}$ and $\mathcal{BBG\text{-}WIBE}$ schemes presented in Section 6.1 and Section 6.2. The schemes are compared in terms of master public key size ($|mpk|$), user secret key size ($|d|$), ciphertext size ($|C|$), decryption time (Dec), the security assumption under which the scheme is proved secure, and whether this proof is in the random oracle model or not. (The generic construction does not introduce any random oracles, but if the security proof of the HIBE scheme is in the random oracle model, then the WIBE obviously inherits this property.) Values refer to the underlying HIBE scheme for the generic scheme, and to the number of group elements ($|mpk|$, $|d|$, $|C|$) or pairing computations (Dec) for the other schemes. L is the maximal hierarchy depth and n is the bit length of an identity string. Figures are worst-case values, usually occurring for identities at level L with all-wildcard ciphertexts. L-BDHI refers to the decisional bilinear Diffie-Hellman inversion assumption [MSK02, BB04].

polynomial in all parameters.

The $\mathcal{Wa\text{-}WIBE}$ scheme is based on Waters' HIBE scheme [Wat05] and provably secure in the standard (i.e., non-random-oracle [BR93]) model under the bilinear decisional Diffie-Hellman (BDDH) assumption. Its efficiency is polynomial in all parameters, but has the disadvantage that each wildcard adds $n+1$ group elements to the ciphertext. In practice, one would typically use the output of a collision-resistant hash function as identity strings, so that $n \approx 160$. The resulting ciphertexts may be prohibitively long for many applications.

Our second direct construction, the $\mathcal{BB\text{-}WIBE}$ scheme, is based on the Boneh-Boyen HIBE scheme [BB04], and adds only two group elements to the ciphertext for each wildcard in the recipient pattern. It is provably secure under the (weaker) selective-identity security notion under the BDDH assumption. We extend an observation of [BB04, BBG05] to the case of WIBE schemes and show how to achieve full security in the random oracle model.

Lastly, the $\mathcal{BBG\text{-}WIBE}$ scheme that we derive from the Boneh-Boyen-Goh [BBG05] HIBE scheme offers more efficient decryption (two pairings, versus $L+1$ for the other schemes) and even shorter ciphertexts if few wildcards are used. (The ciphertext size is not constant, however, but depends linearly on the number of wildcards in the recipient pattern.) The scheme is provably selective-identity secure under the decisional L-bilinear Diffie-Hellman inversion (L-BDHI) assumption [BBG05], which is a stronger assumption than BDDH.

We note that all of our fully (as opposed to selective-identity) secure constructions lose a factor exponential in L in the reduction to the underlying assumption. This limits the secure use of our schemes to very small hierarchy depths. This (quite severe) restriction is not so surprising however, viewing that WIBE schemes are in fact a generalization of HIBE schemes, and that the same restriction arises in all currently-known HIBE constructions. We therefore leave the construction of a truly polynomial (in terms of efficiency and security) WIBE scheme as an open problem.

Finally, we show how to achieve chosen ciphertext security in Section 7. We adapt the technique of Canetti, Halevi and Katz [CHK04] and show that an L-level CCA-secure WIBE can be built from a $(2L+2)$-level CPA-secure WIBE and a strongly unforgeable one-time signature scheme.

2 Basic Definitions

In this section we introduce some notation, computational problems and basic primitives that we will use throughout the rest of the paper. Let $\mathbb{N} = \{0, 1, \ldots\}$ be the set of natural numbers. Let ε be the empty string. If $n \in \mathbb{N}$, then $\{0,1\}^n$ denotes the set of n-bit strings, and $\{0,1\}^*$ is the set of all bit strings. More generally, if S is a set, then S^n is the set of n-tuples of elements of S, $S^{\leq n}$ is the set of tuples of length at most n. If S is finite, then $x \stackrel{\$}{\leftarrow} S$ denotes the assignment to x of an element chosen uniformly at random from S. If $\mathcal{A}$ is an algorithm, then $y \leftarrow \mathcal{A}(x)$ denotes the assignment to y of the output of $\mathcal{A}$ on input x, and if $\mathcal{A}$ is randomised, then $y \stackrel{\$}{\leftarrow} \mathcal{A}(x)$ denotes that the output of an execution of $\mathcal{A}(x)$ with fresh coins is assigned to y.

Bilinear maps and related assumptions. Let $\mathbb{G}, \mathbb{G}_{\mathrm{T}}$ be multiplicative groups of prime order p with an admissible map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_{\mathrm{T}}$. By admissible we mean that the map is bilinear, non-degenerate and efficiently computable. Bilinearity means that for all $a, b \in \mathbb{Z}_p$ and all $g \in \mathbb{G}$ we have $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$. By non-degenerate we mean that $\hat{e}(g, g) = 1$ if and only if $g = 1$.

In such a setting we can define a number of computational problems. The first we shall be interested in is called the bilinear decisional Diffie-Hellman (BDDH) problem: given a tuple $(g, A = g^a, B = g^b, C = g^c, T)$, the problem is to decide whether $T = \hat{e}(g, g)^{abc}$ or whether it is a random element of $\mathbb{G}_{\mathrm{T}}$. More formally, we define the following game between an adversary $\mathcal{A}$ and a challenger. The challenger first chooses a random generator $g \stackrel{\$}{\leftarrow} \mathbb{G}^*$, random integers $a, b, c \stackrel{\$}{\leftarrow} \mathbb{Z}_p$, a random element $T \stackrel{\$}{\leftarrow} \mathbb{G}_{\mathrm{T}}$ and a random bit β. If $\beta = 1$ it feeds $\mathcal{A}$ as input the tuple $(g, g^a, g^b, g^c, \hat{e}(g, g)^{abc})$, if $\beta = 0$ it feeds it (g, g^a, g^b, g^c, T). The adversary $\mathcal{A}$ must then output its guess β' for β. The adversary has advantage ϵ in solving the BDDH problem if

$$\left| \Pr\left[\mathcal{A}\big(g, g^a, g^b, g^c, \hat{e}(g, g)^{abc}\big) = 1\right] - \Pr\left[\mathcal{A}\big(g, g^a, g^b, g^c, T\big) = 1\right] \right| \geq 2\epsilon,$$

where the probabilities are over the choice of g, a, b, c, T and over the random coins of $\mathcal{A}$.

Definition 2.1 The (t, ϵ) BDDH assumption holds if no t-time adversary $\mathcal{A}$ has at least ϵ advantage in the above game.

We note that throughout this paper we will assume that the time t of an adversary includes its code size, in order to exclude trivial "lookup" adversaries.

A second problem we will use in our constructions is the ℓ-bilinear Diffie-Hellman Inversion (ℓ-BDHI) problem [MSK02, BB04]. The problem is to, compute $\hat{e}(g, g)^{1/\alpha}$ for random $g \stackrel{\$}{\leftarrow} \mathbb{G}^*$ and $\alpha \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ given $g, g^\alpha, \ldots, g^{(\alpha^\ell)}$. The decisional variant of this problem is to distinguish $\hat{e}(g, g)^{1/\alpha}$ from a random element of $\mathbb{G}_{\mathrm{T}}$. We say that adversary $\mathcal{A}$ has advantage ϵ in solving the decisional ℓ-BDHI problem if

$$\left| \Pr\left[\mathcal{A}\big(g, g^\alpha, \ldots, g^{(\alpha^\ell)}, \hat{e}(g, g)^{1/\alpha}\big) = 1\right] - \Pr\left[\mathcal{A}\big(g, g^\alpha, \ldots, g^{(\alpha^\ell)}, T\big) = 1\right] \right| \geq 2\epsilon ,$$

where the probability is over the random choice of $g \stackrel{\$}{\leftarrow} \mathbb{G}^*$, $\alpha \stackrel{\$}{\leftarrow} \mathbb{Z}_p$, $T \stackrel{\$}{\leftarrow} \mathbb{G}_{\mathrm{T}}$ and over the coins of $\mathcal{A}$.

Definition 2.2 The (t, ϵ) decisional ℓ-BDHI assumption holds if no t-time adversary $\mathcal{A}$ has at least ϵ advantage in the above game.

Identity-based encryption schemes. An identity-based encryption (IBE) scheme is a tuple of algorithms $\mathcal{IBE} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ providing the following functionality. The trusted authority runs Setup to generate a master key pair (mpk, msk). It publishes the master public key mpk and keeps the master secret key msk private. When a user with identity ID wishes to become part of the system, the trusted authority generates a user decryption key $d_{ID} \stackrel{\$}{\leftarrow} \mathsf{KeyDer}(msk, ID)$, and sends this key over a secure and authenticated channel to the user. To send an encrypted message $\mathfrak{m}$ to the user with identity ID, the sender computes the ciphertext $C \stackrel{\$}{\leftarrow} \mathsf{Enc}(mpk, ID, \mathfrak{m})$, which can be decrypted by the user as $\mathfrak{m} \leftarrow \mathsf{Dec}(d_{ID}, C)$. We refer to [BF03] for details on the security definitions for IBE schemes.

Hierarchical IBE schemes. In a hierarchical IBE (HIBE) scheme, users are organised in a tree of depth L, with the root being the master trusted authority. The identity of a user at level $0 \leq \ell \leq L$ in the tree is given by a vector $ID = (ID_1, \ldots, ID_\ell) \in (\{0,1\}^*)^\ell$. A HIBE scheme is a tuple of algorithms $\mathcal{HIBE} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ providing the same functionality as in an IBE scheme, except that a user $ID = (ID_1, \ldots, ID_\ell)$ at level ℓ can use its own secret key d_{ID} to generate a secret key for any of its children $ID' = (ID_1, \ldots, ID_\ell, ID_{\ell+1})$ via $d_{ID'} \stackrel{\$}{\leftarrow} \mathsf{KeyDer}(d_{ID}, ID_{\ell+1})$. Note that by iteratively applying the KeyDer algorithm, user ID can derive secret keys for any of its descendants $ID' = (ID_1, \ldots, ID_{\ell+\delta})$, $\delta \geq 0$. We will occasionally use the overloaded notation $d_{ID'} \stackrel{\$}{\leftarrow} \mathsf{KeyDer}(d_{ID}, (ID_{\ell+1}, \ldots, ID_{\ell+\delta}))$ to denote this process. The secret key of the root identity at level 0 is $d_\varepsilon = msk$. Encryption and decryption are the same as for IBE, but with vectors of bit strings as identities instead of ordinary bit strings. For $1 \leq i \leq \ell$ and $I \subseteq \{1, \ldots, \ell\}$, we will occasionally use the notations $ID|_{\leq i}$ to denote the vector $(ID_1, \ldots, ID_i)$, $ID|_{> i}$ to denote $(ID_{i+1}, \ldots, ID_\ell)$, and $ID|_I$ to denote $(ID_{i_1}, \ldots, ID_{i_{|I|}})$ where $i_1, \ldots, i_{|I|}$ are the elements of a set $I \subseteq \mathbb{N}$ in increasing order. Also, if $S \subset \mathbb{N}$, then we define $S|_{\leq i} = \{j \in S : j \leq i\}$ and $S|_{> i} = \{j \in S : j > i\}$.

The security of a HIBE scheme is defined through the following game. In a first phase, the adversary is given as input the master public key mpk of a freshly generated key pair $(mpk, msk) \stackrel{\$}{\leftarrow} \mathsf{Setup}$ as input. In a chosen-plaintext attack (IND-ID-CPA), the adversary is given access to a key derivation oracle that on input of an identity $ID = (ID_1, \ldots, ID_\ell)$, returns the secret key $d_{ID} \stackrel{\$}{\leftarrow} \mathsf{KeyDer}(msk, ID)$ corresponding to identity ID. In a chosen-ciphertext attack (IND-ID-CCA), the adversary is additionally given access to a decryption oracle that for a given identity $ID = (ID_1, \ldots, ID_\ell)$ and a given ciphertext C returns the decryption $\mathfrak{m} \leftarrow \mathsf{Dec}(\mathsf{KeyDer}(msk, ID), C)$.

At the end of the first phase, the adversary outputs two equal-length challenge messages $\mathfrak{m}_0^*$, $\mathfrak{m}_1^* \in \{0,1\}^*$ and a challenge identity $ID^* = (ID_1^*, \ldots, ID_{\ell^*}^*)$, where $0 \leq \ell^* \leq L$. The game chooses a random bit $b \stackrel{\$}{\leftarrow} \{0,1\}^*$, generates a challenge ciphertext $C^* \stackrel{\$}{\leftarrow} \mathsf{Enc}(mpk, ID^*, \mathfrak{m}_b^*)$ and gives C^* as input to the adversary for the second phase, during which it gets access to the same oracles as during the first phase. The adversary wins the game if it outputs a bit $b' = b$ without ever having queried the key derivation oracle on any ancestor identity $ID = (ID_1^*, \ldots, ID_\ell^*)$ of ID^*, $\ell \leq \ell^*$, and, additionally, in the IND-ID-CCA case, without ever having queried (ID^*, C^*) to the decryption oracle.

Definition 2.3 A HIBE scheme is $(t, q_{\mathrm{K}}, \epsilon)$ IND-ID-CPA-secure if all t-time adversaries making at most q_{K} queries to the key derivation oracle have at most advantage ϵ in winning the IND-ID-CPA game described above. It is said to be $(t, q_{\mathrm{K}}, q_{\mathrm{D}}, \epsilon)$ IND-ID-CCA-secure if all such adversaries that additionally make at most q_{D} queries to the decryption oracle have advantage at most ϵ in winning the IND-ID-CCA game described above.

In a *selective-identity* (sID) attack [BB04], the adversary has to output the challenge identity ID^* at the very beginning of the game, before even seeing the master public key. The definitions for IND-sID-CPA and IND-sID-CCA security are otherwise identical to those above. In the random oracle model [BR94], all algorithms, as well as the adversary, have access to a random oracle mapping arbitrary bit strings onto a range that possibly depends on the master public key. All above security definitions then take an extra parameter q_{H} denoting the adversary's maximum number of queries to the random oracle.

3 Identity-Based Encryption with Wildcards

Syntax. Identity-based encryption with wildcards (WIBE) schemes are essentially a generalisation of HIBE schemes where at the time of encryption, the sender can decide to make the ciphertext decryptable by a whole range of users whose identities match a certain pattern. Such a pattern is described by a vector $P = (P_1, \ldots, P_\ell) \in (\{0,1\}^* \cup \{*\})^\ell$, where $*$ is a special wildcard symbol. We say that identity $ID = (ID_1, \ldots, ID_{\ell'})$ *matches* P, denoted $ID \in_* P$, 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 is reasonable for our purposes because any ancestor can derive the secret key of a matching descendant identity anyway.

More formally, a WIBE scheme is a tuple of algorithms $\mathcal{WIBE} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ providing the following functionality. The Setup and KeyDer algorithms behave exactly as those of a HIBE scheme. To create a ciphertext of message $\mathfrak{m} \in \{0,1\}^*$ intended for all identities matching pattern P, the sender computes $C \xleftarrow{\$} \mathsf{Enc}(mpk, P, \mathfrak{m})$. Any of the intended recipients $ID \in_* P$ can decrypt the ciphertext using its own decryption key as $\mathfrak{m} \leftarrow \mathsf{Dec}(d_{ID}, C)$. Correctness requires that for all key pairs (mpk, msk) output by Setup, all messages $\mathfrak{m} \in \{0,1\}^*$, all $0 \leq \ell \leq L$, all patterns $P \in (\{0,1\}^* \cup \{*\})^\ell$, and all identities $ID \in_* P$, $\mathsf{Dec}(\ \mathsf{KeyDer}(msk, ID)\ ,\ \mathsf{Enc}(mpk, P, \mathfrak{m})\ ) = \mathfrak{m}$ with probability one.

Security. We define the security of WIBE schemes analogously to that of HIBE schemes, but with the adversary choosing a challenge pattern instead of an identity to which the challenge ciphertext will be encrypted. To exclude trivial attacks, the adversary is not able to query the key derivation oracle on any identity that matches the challenge pattern, nor is it able to query the decryption oracle on the challenge ciphertext in combination with any identity matching the challenge pattern.

More formally, security is defined through the following game with an adversary. In the first phase, the adversary is run on input the master public key of a freshly generated key pair $(mpk, msk) \xleftarrow{\$} \mathsf{Setup}$. In a chosen-plaintext attack (IND-WID-CPA), the adversary is given access to a key derivation oracle that on input $ID = (ID_1, \ldots, ID_\ell)$ returns $d_{ID} \xleftarrow{\$} \mathsf{KeyDer}(msk, ID)$. In a chosen-ciphertext attack (IND-WID-CCA), the adversary additionally has access to a decryption oracle that on input a ciphertext C and an identity $ID = (ID_1, \ldots, ID_\ell)$ returns $\mathfrak{m} \leftarrow \mathsf{Dec}(\mathsf{KeyDer}(msk, ID), C)$.

At the end of the first phase, the adversary outputs two equal-length challenge messages $\mathfrak{m}_0^*, \mathfrak{m}_1^*$ and a challenge pattern $P^* = (P_1^*, \ldots, P_{\ell^*}^*)$ where $0 \leq \ell^* \leq L$. The adversary is given a challenge ciphertext $C^* \xleftarrow{\$} \mathsf{Enc}(mpk, P^*, \mathfrak{m}_b^*)$ for a randomly chosen bit β, and is given access to the same oracles as during the first phase of the attack. The second phase ends when the adversary outputs a bit β'. The adversary is said to win the IND-WID-CPA game if $\beta' = \beta$ and if it never queried the key derivation oracle for the keys of any identity that matches the target pattern (i.e., any ID such that $ID \in_* P^*$). Also, in a chosen-ciphertext attack (IND-WID-CCA), the adversary cannot query the decryption oracle on C^* in combination with any identity $ID \in_* P^*$ matching the challenge pattern.

Definition 3.1 A WIBE scheme is $(t, q_{\mathrm{K}}, \epsilon)$ IND-WID-CPA-secure if all t-time adversaries making at most q_{K} queries to the key derivation oracle have at most advantage ϵ in winning the IND-WID-CPA game described above. It is said to be $(t, q_{\mathrm{K}}, q_{\mathrm{D}}, \epsilon)$ IND-WID-CCA-secure if all such adversaries that additionally make at most q_{D} queries to the decryption oracle have advantage at most ϵ in winning the IND-WID-CCA game described above.

As for the case of HIBEs, we also define a weaker selective-identity (sWID) security notion, in which the adversary commits to the challenge pattern at the beginning of the game, before the master public key is made available. The notions of IND-sWID-CPA and IND-sWID-CCA security are defined analogously to the above. In the random oracle model, the additional parameter q_{H} denotes the adversary's maximum number of queries to the random oracle, or the total number of queries to all random oracles when it has access to multiple ones.

4 A Generic Construction

We first point out that a generic construction of a WIBE scheme exists based on any HIBE scheme, but with a secret key size that is exponential in the depth of the hierarchy tree. Let "$*$" denote a dedicated bitstring that cannot occur as a user identity. Then the secret key of a user with identity $(ID_1, \ldots, ID_\ell)$ in the WIBE scheme contains the HIBE secret keys of all patterns matching this identity. For example, the secret key of identity (ID_1, ID_2) contains four HIBE secret keys, namely those corresponding to identities $(ID_1, ID_2), (\text{“}*\text{”}, ID_2), (ID_1, \text{“}*\text{”}), (\text{“}*\text{”}, \text{“}*\text{”})$. In general, the secret key of $(ID_1, \ldots, ID_\ell)$ contains the HIBE secret keys of all 2^ℓ identities $(ID'_1, \ldots, ID'_\ell)$ such that $ID'_i = ID_i$ or $ID'_i = \text{“}*\text{”}$ for all $i = 1, \ldots, \ell$. To encrypt to a pattern $(P_1, \ldots, P_\ell)$, one uses the HIBE scheme to encrypt to the identity obtained by replacing each wildcard in the pattern with the "$*$" string, i.e. the identity $(ID_1, \ldots, ID_\ell)$ where $ID_i = \text{“}*\text{”}$ if $P_i = *$ and $ID_i = P_i$ otherwise. Decryption is done by selecting the appropriate secret key from the list and using the decryption algorithm of the HIBE scheme.

The efficiency of the WIBE scheme thus obtained is roughly the same as that of the underlying HIBE scheme, but with the major disadvantage that the size of the secret key is 2^ℓ times that of a secret key in the underlying HIBE scheme. This is highly undesirable for many applications, especially since the secret key may very well be kept on an expensive, secure storage device. Moreover, from a theoretical point of view, it is interesting to investigate whether WIBE schemes exist with overhead polynomial in all parameters. We answer this question in the affirmative here by presenting direct schemes with secret key size linear in ℓ. Unfortunately, for all of our schemes, this reduction in key size comes at the cost of linear-size ciphertexts, while the generic scheme can achieve contant-size ciphertexts when underlain by a HIBE with constant ciphertext size, e.g. that of [BBG05].

Another related primitive is fuzzy identity-based encryption (FIBE) [SW05], which allows a ciphertext encrypted to identity ID to be decrypted by any identity ID' that is "close" to ID according to some metric. In the schemes of [SW05], an identity is a subset containing n elements from a finite universe. Two identities ID and ID' are considered "close" if $|ID \cap ID'| \geq d$ for some parameter d. Such a FIBE scheme could be used to construct a limited WIBE scheme (without hierarchical key derivation) by letting identity $(ID_1, \ldots, ID_n)$ correspond to the set $\{1\|ID_1, \ldots, \ell\|ID_n\}$. To encrypt to a pattern $P = (P_1, \ldots, P_n)$ containing $n - d$ wildcards, one uses the FIBE scheme to encrypt to the set $\{P'_1, \ldots, P'_n\}$ where $P'_i = i\|P_i$ if $P_i \neq *$ and $P'_i = i\|\text{“}*\text{”}$ if $P_i = *$. The schemes of [SW05] require n, d to be fixed beforehand. Variable identity lengths ℓ and number of wildcards w can be accomodated for by setting $n = 2L$, $d = L$ and by letting the set corresponding to identity $(ID_1, \ldots, ID_\ell)$ be $\{1\|ID_1, \ldots, \ell\|ID_\ell,\ (\ell+1)\|\varepsilon, \ldots, L\|\varepsilon,\ 1\|\text{“}*\text{”}, \ldots, L\|\text{“}*\text{”}\}$. One can then encrypt to pattern $(P_1, \ldots, P_\ell)$ by encrypting to the set $\{1\|P'_1, \ldots, \ell\|P'_\ell,\ (\ell+1)\|\varepsilon, \ldots, 2L\|\varepsilon\}$, where the P'_i are defined as above.

5 A Construction from Waters' HIBE Scheme

5.1 Waters' HIBE Scheme

Waters [Wat05] argued that his IBE scheme can easily be modified into a L-level HIBE scheme as per [BB04]. Here we explicitly present this construction, that we refer to as the $\mathcal{W}a$-$\mathcal{HIBE}$ scheme, as it will be useful in the understanding of our first construction of a WIBE scheme.

Setup. The trusted authority chooses random generators $g_1, g_2, u_{1,0}, \ldots, u_{L,n} \stackrel{\$}{\leftarrow} \mathbb{G}^*$ and a random value $\alpha \stackrel{\$}{\leftarrow} \mathbb{Z}_p$, where L is the maximum hierarchy depth and n is the length of an identity string. Next, it computes $h_1 \leftarrow g_1^\alpha$ and $h_2 \leftarrow g_2^\alpha$. The master public key is $mpk = (g_1, g_2, h_1, u_{1,0}, \ldots, u_{L,n})$, the corresponding master secret key is $msk = h_2$.

Key Derivation. A user's identity is given by a vector $ID = (ID_1, \ldots, ID_\ell)$ where each ID_i is a n-bit string, applying a collision-resistant hash function if necessary. When we write "$j \in ID_i$", we mean that the variable j iterates over all bit positions $1 \le j \le n$ where the j-th bit of ID_i is one. Using this notation, for $i = 1, \ldots, L$, we define the function

$$F_i(ID_i) = u_{i,0} \prod_{j \in ID} u_{i,j}$$

where the $u_{i,j}$ are the elements in the master public key. To compute the decryption key for identity ID from the master secret key, first random values $r_1, \ldots, r_\ell \xleftarrow{\$} \mathbb{Z}_p$ are chosen, then the private key d_{ID} is constructed as

$$(a_0, a_1, \ldots, a_\ell) = \left(h_2 \prod_{i=1}^{\ell} F_i(ID_i)^{r_i} \;,\; g_1^{r_1}, \ldots, g_1^{r_\ell} \right) .$$

A secret key for identity $ID = (ID_1, \ldots, ID_\ell)$ can be computed by its parent with identity $(ID_1, \ldots, ID_{\ell-1})$ as follows. Let $(a_0, a_1, \ldots, a_{\ell-1})$ be the parent's secret key. It chooses $r_\ell \xleftarrow{\$} \mathbb{Z}_p$ and outputs

$$d_{ID} = (a_0 \cdot F_i(ID_i)^{r_\ell} \;,\; a_1, \ldots, a_{\ell-1} \;,\; g_1^{r_\ell}) .$$

Encryption. To encrypt a message $\mathfrak{m} \in \mathbb{G}_\mathrm{T}$ for identity $ID = (ID_1, \ldots, ID_\ell)$, the sender chooses $t \xleftarrow{\$} \mathbb{Z}_p$ and computes the ciphertext $C = (C_1, C_2, C_3)$ as

$$C_1 \leftarrow g_1^t \;,\; C_2 \leftarrow \left(C_{2,i} = F_i(ID_i)^t\right)_{i=1,\ldots,\ell} \;,\; C_3 \leftarrow \mathfrak{m} \cdot \hat{e}(h_1, g_2)^t \; .$$

Decryption. If the receiver is the root authority (i.e., the empty identity $ID = \varepsilon$) holding the master key $msk = h_2$, then he can recover the message by computing $\mathfrak{m} \leftarrow C_3/\hat{e}(C_1, h_2)$. Any other receiver with identity $ID = (ID_1, \ldots, ID_\ell)$ and decryption key $d_{ID} = (a_0, a_1, \ldots, a_\ell)$ decrypts a ciphertext $C = (C_1, C_2, C_3)$ as follows.

$$\begin{aligned}
C_3 \cdot \frac{\prod_{i=1}^{\ell} \hat{e}\left(a_i, C_{2,i}\right)}{\hat{e}\left(C_1, a_0\right)} &= \mathfrak{m} \cdot \hat{e}(h_1, g_2)^t \cdot \frac{\prod_{i=1}^{\ell} \hat{e}\left(g_1^{r_i}, F_i(ID_i)^t\right)}{\hat{e}\left(g_1^t, h_2 \prod_{i=1}^{\ell} F_i(ID_i)^{r_i}\right)} \\
&= \mathfrak{m} \cdot \hat{e}(h_1, g_2)^t \cdot \frac{\prod_{i=1}^{\ell} \hat{e}\left(g_1^{r_i}, F_i(ID_i)^t\right)}{\hat{e}\left(g_1^t, h_2\right) \cdot \hat{e}\left(g_1^t, \prod_{i=1}^{\ell} F_i(ID_i)^{r_i}\right)} \\
&= \mathfrak{m} \cdot \frac{\hat{e}(g_1^\alpha, g_2)^t}{\hat{e}\left(g_1^t, g_2^\alpha\right)} \cdot \frac{\prod_{i=1}^{\ell} \hat{e}\left(g_1^{r_i}, F_i(ID_i)^t\right)}{\prod_{i=1}^{\ell} \hat{e}\left(g_1^t, F_i(ID_i)^{r_i}\right)} \;=\; \mathfrak{m}
\end{aligned}$$

Waters [Wat05] informally states that the above HIBE scheme is IND-ID-CPA secure under the BDDH assumption, in the sense that if there exists a $(t, q_\mathrm{K}, \epsilon)$-adversary against the HIBE, then there exists an algorithm solving the BDDH problem with advantage $\epsilon' = O((nq_\mathrm{K})^L \epsilon)$.

5.2 The *Wa-WIBE* Scheme

We first introduce some additional notation. If $P = (P_1, \ldots, P_\ell)$ is a pattern, then let $|P| = \ell$ be the length of P, let $\mathrm{W}(P)$ be the set containing all wildcard indices in P, i.e. the indices $1 \le i \le \ell$ such that $P_i = *$, and let $\overline{\mathrm{W}}(P)$ be the complementary set containing all non-wildcard indices. Clearly $\mathrm{W}(P) \cap \overline{\mathrm{W}}(P) = \emptyset$ and $\mathrm{W}(P) \cup \overline{\mathrm{W}}(P) = \{1, \ldots, \ell\}$. We also extend the notations $P|_{\le i}$, $P|_{> i}$ and $P|_I$ that we introduced for identity vectors to patterns in the natural way.

Intuitively, we adapt the $\mathcal{W}a\text{-}\mathcal{HIBE}$ scheme to support wildcards by observing that the ciphertext components $C_{2,i}$ are actually products of $u_{i,0}^t$ and those factors $u_{i,j}^t$ for which the j-th bith of ID_i is one. If we include these factors separately in the ciphertext, instead of their product, then we can postpone the computation of the product to decryption time and let each recipient combine the factors corresponding to his own identity.

Of course, one still needs to show that giving away these factors in the ciphertext does not affect security. We first describe our construction in more detail, and subsequently show in Theorem 5.1 that its security is implied by that of the $\mathcal{W}a\text{-}\mathcal{HIBE}$ scheme. We build a WIBE scheme $\mathcal{W}a\text{-}\mathcal{WIBE}$ from the $\mathcal{W}a\text{-}\mathcal{HIBE}$ scheme with Setup and KeyDer algorithms identical to those of the $\mathcal{W}a\text{-}\mathcal{HIBE}$ scheme, and with encryption and decryption algorithms that work as follows.

Encryption. To encrypt a message $\mathfrak{m} \in \mathbb{G}_T$ to all identities matching pattern $P = (P_1, \ldots, P_\ell)$, the sender chooses $t \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ and outputs the ciphertext $C = (P, C_1, C_2, C_3, C_4)$, where

$$C_1 \leftarrow g_1^t \qquad C_2 \leftarrow \left(C_{2,i} = F_i(P_i)^t\right)_{i \in \overline{\mathrm{W}}(P)}$$
$$C_3 \leftarrow \mathfrak{m} \cdot \hat{e}(h_1, g_2)^t \qquad C_4 \leftarrow \left(C_{4,i,j} = u_{i,j}^t\right)_{i \in \mathrm{W}(P),\, j=0,\ldots,n}$$

Decryption. If the receiver is the root authority (i.e., the empty identity $ID = \varepsilon$) holding the master key $msk = h_2$, then it can recover the message by computing $C_3/\hat{e}(C_1, h_2)$. Any other receiver with identity $ID = (ID_1, \ldots, ID_\ell)$ matching the pattern P to which the ciphertext was created (i.e., $ID \in_* P$) can decrypt the ciphertext $C = (P, C_1, C_2, C_3, C_4)$ by computing $C_2' = \left(C_{2,i}'\right)_{i=1,\ldots,\ell}$ as

$$C_{2,i}' = F_i(ID_i)^t \leftarrow \begin{cases} C_{2,i} & \text{if } i \in \overline{\mathrm{W}}(P) \\ C_{4,i,0} \cdot \prod_{j \in ID_i} C_{4,i,j} & \text{if } i \in \mathrm{W}(P)|_{\leq \ell} \end{cases}$$

and by using his secret key to decrypt the ciphertext $C' = (C_1, C_2', C_3)$ via the Dec algorithm of the $\mathcal{W}a\text{-}\mathcal{HIBE}$ scheme.

The master public key of the $\mathcal{W}a\text{-}\mathcal{WIBE}$ scheme contains $(n+1)L + 3$ group elements. Encrypting to a pattern of length ℓ containing w wildcards comes at the cost of $\ell + nw + 2$ exponentiations and $\ell + nw + 2$ group elements in the ciphertext; in the worst case of $\ell = w = L$ this means $(n+1)L + 2$ exponentiations and group elements. (The pairing $\hat{e}(h_1, g_2)$ can be precomputed.) Decryption requires the computation of $\ell + 1$ pairings.

In terms of efficiency, the $\mathcal{W}a\text{-}\mathcal{WIBE}$ scheme performs well enough to be considered for use in practice, but definitely leaves room for improvement. The main problem is the dependency of the scheme on n, the bit length of identity strings. In practice, one would typically use the output of a collision-resistant hash function as identity strings, so that $n = 160$ for a reasonable level of security. We note that the techniques of [CS06, Nac05] could be applied to trade a factor d of efficiency against losing a factor 2^{Ld} in the tightness of the reduction.

We now prove the security of the $\mathcal{W}a\text{-}\mathcal{WIBE}$ scheme. To make the proof more modular, and to avoid repeating the work of [Wat05], we do this by reducing to the security of the $\mathcal{W}a\text{-}\mathcal{HIBE}$ scheme, rather than to the BDDH problem directly.

Theorem 5.1 If the $\mathcal{W}a\text{-}\mathcal{HIBE}$ of depth L is $(t, q_{\mathrm{K}}, \epsilon)$ IND-ID-CPA-secure, then the $\mathcal{W}a\text{-}\mathcal{WIBE}$ scheme of depth L is $(t', q_{\mathrm{K}}', \epsilon')$ IND-WID-CPA-secure for all

$$t' \;\leq\; t - Ln(1 + q_{\mathrm{K}}) \cdot t_{\exp} \;, \quad q_{\mathrm{K}}' \;\leq\; q_{\mathrm{K}} \;, \quad \text{and} \quad \epsilon' \;\geq\; \epsilon/2^L \;,$$

and $t_{\exp}$ is the time it takes to perform an exponentiation in $\mathbb{G}$.

Proof: The proof of Theorem 5.1 is by contradiction. That is, we first assume that there exists an adversary $\mathcal{A}$ that breaks the IND-WID-CPA-security of the *Wa-WIBE* scheme and then we show how to efficiently build another adversary $\mathcal{B}$ which uses $\mathcal{A}$ to break the security of the *Wa-HIBE* scheme.

Let $mpk_{\mathrm{H}} = (g_1, g_2, h_1, u_{1,0}, \ldots, u_{L,n})$ be the master public key of the *Wa-HIBE* scheme that adversary $\mathcal{B}$ receives as input for its first phase. The idea of the proof is that $\mathcal{B}$ will guess upfront where in the challenge pattern P^* the wildcards are going to be, and "project" the non-wildcard levels of the identity tree of the WIBE scheme onto the first levels of the HIBE scheme. In particular, $\mathcal{B}$ will reuse values $u_{i,j}$ from mpk_{H} for the non-wildcard levels, and will embed new values $u'_{i,j}$ values of which $\mathcal{B}$ knows the discrete logarithms for wildcard levels.

First, $\mathcal{B}$ guesses a random vector $\hat{P} = (\hat{P}_1, \ldots, \hat{P}_L) \stackrel{\$}{\leftarrow} \{\varepsilon, *\}^L$. Define the projection function $\pi : \{1, \ldots, L\} \to \{0, \ldots, L\}$ such that

$$\pi(i) \;=\; \begin{cases} 0 & \text{if } i \in \mathrm{W}(\hat{P}) \\ i - \left|\mathrm{W}(\hat{P})|_{\leq i}\right| & \text{otherwise} \end{cases}$$

Intuitively, $\mathcal{B}$ will "project" identities at level i of the WIBE scheme onto level $\pi(i)$ of the HIBE scheme whenever $\pi(i) \neq 0$. Next, the adversary $\mathcal{B}$ runs adversary $\mathcal{A}$ providing it as input for its first phase a public-key $mpk_{\mathrm{W}} = (g_1, g_2, h_1, u'_{1,0}, \ldots, u'_{L,n})$, where for all $1 \leq i \leq L$ and $0 \leq j \leq n$, the elements $u'_{i,j}$ are generated as $u'_{i,j} \leftarrow g_1^{\alpha_{i,j}}$ where $\alpha_{i,j} \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ if $i \in \mathrm{W}(\hat{P})$, and $u'_{i,j} \leftarrow u_{\pi(i),j}$ otherwise. Define functions $F'_i(ID'_i) = u'_{i,0} \prod_{j \in ID'_i} u'_{i,j}$. Notice that $mpk_{\mathcal{A}}$ is distributed exactly as it would be if produced by the setup algorithm described in Section 5.2.

During the first phase, $\mathcal{B}$ has to answer all the key derivation queries $ID' = (ID'_1, \ldots, ID'_\ell)$ that $\mathcal{A}$ is allowed to ask. For that, $\mathcal{B}$ first computes the corresponding identity on the HIBE tree $ID = ID'|_{\overline{\mathrm{W}}(\hat{P})}$, which is the identity obtained by removing from ID' all components at levels where $\hat{P}$ contains a wildcard. That is, the identity ID is obtained from ID' by projecting the component at level i of the WIBE onto level $\pi(i)$ of the HIBE if $\pi(i) \neq 0$. $\mathcal{B}$ then queries its own key derivation oracle for the *Wa-HIBE* scheme on input ID to get the key $d = (a_0, \ldots, a_{\pi(\ell)})$. From this, it computes the key $d' = (a'_0, \ldots, a'_\ell)$ as

$$a'_0 \;\leftarrow\; a_0 \cdot \prod_{i \in \mathrm{W}(\hat{P})} F'_i(ID'_i)^{r_i}$$

$$a'_i \;\leftarrow\; \begin{cases} g_1^{r_i} & \text{if } i \in \mathrm{W}(\hat{P}) \\ a_{\pi(i)} & \text{if } i \in \overline{\mathrm{W}}(\hat{P}) \end{cases}$$

where $r_i \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ for all $i \in \mathrm{W}(\hat{P})$. At the end of its first phase, $\mathcal{A}$ outputs the challenge pattern $P^* = (P^*_1, \ldots, P^*_{\ell^*})$ and challenge messages $\mathfrak{m}^*_0, \mathfrak{m}^*_1$. If $\mathrm{W}(P^*) \neq \mathrm{W}(\hat{P})$ then $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ outputs the corresponding HIBE identity $ID^* = P^*|_{\overline{\mathrm{W}}(P^*)}$ together with challenge messages $\mathfrak{m}^*_0, \mathfrak{m}^*_1$. Let $C^* = (C^*_1, C^*_2, C^*_3)$ be the challenge ciphertext that $\mathcal{B}$ receives in return from its challenger, meaning that C^* is an encryption of $\mathfrak{m}^*_b$ with respect to the identity ID^*, where b is the secret bit chosen at random by the challenger. $\mathcal{B}$ sets $C'^*_1 \leftarrow C^*_1$, $C'^*_2 \leftarrow C^*_2$, $C'^*_3 \leftarrow C^*_3$ and $C'^*_4 \leftarrow (C_1^{*\alpha_{i,j}})_{i \in \mathrm{W}(P^*),\, j=0,\ldots,n}$ and sends to $\mathcal{A}$ the ciphertext $C'^* = (P^*, C'^*_1, C'^*_2, C'^*_3, C'^*_4)$ as the input for its second phase. During the second phase, $\mathcal{A}$ is then allowed to issue more key derivation queries, which are answered by $\mathcal{B}$ exactly as in the first phase. When $\mathcal{A}$ outputs a bit b', $\mathcal{B}$ outputs b' and stops.

In order to analyse the success probability of $\mathcal{B}$, we first need to show that the simulation it provides to $\mathcal{A}$ is correct. The secret key $d' = (a'_0, \ldots, a'_\ell)$ returned for identity $(ID'_1, \ldots, ID'_\ell)$ can be seen to be correctly distributed since if $a'_i = g_1^{r_i}$ for $1 \le i \le \ell$ then

$$\begin{aligned}
a'_0 &= a_0 \cdot \prod_{i \in \mathrm{W}(\hat{P})} F'_i(ID'_i)^{r_i} \\
&= h_2 \cdot \prod_{i \in \overline{\mathrm{W}}(\hat{P})} F_{\pi(i)}(ID'_i)^{r_i} \cdot \prod_{i \in \mathrm{W}(\hat{P})} F'_i(ID'_i)^{r_i} \\
&= h_2 \cdot \prod_{i \in \overline{\mathrm{W}}(\hat{P})} \left(u_{\pi(i),0} \prod_{j \in ID'_i} u_{\pi(i),j} \right)^{r_i} \cdot \prod_{i \in \mathrm{W}(\hat{P})} F'_i(ID'_i)^{r_i} \\
&= h_2 \cdot \prod_{i \in \overline{\mathrm{W}}(\hat{P})} \left(u'_{i,0} \prod_{j \in ID'_i} u'_{i,j} \right)^{r_i} \cdot \prod_{i \in \mathrm{W}(\hat{P})} F'_i(ID'_i)^{r_i} \\
&= h_2 \cdot \prod_{i=1}^{\ell} F'_i(ID'_i)^{r_i}
\end{aligned}$$

Moreover, the challenge ciphertext $C'^* = (P^*, C'^*_1, C'^*_2, C'^*_3, C'^*_4)$ sent to $\mathcal{A}$ can be seen to be correctly formed when $\mathrm{W}(P^*) = \mathrm{W}(\hat{P})$ as follows. Consider the ciphertext $C^* = (C^*_1, C^*_2, C^*_3)$ that $\mathcal{B}$ receives back from the challenger after outputting $(ID^*, \mathfrak{m}^*_0, \mathfrak{m}^*_1)$ where $ID^* = P^*|_{\overline{\mathrm{W}}(P^*)}$. We know that, for unknown values $t \in \mathbb{Z}_p$ and $b \in \{0, 1\}$, $C^*_1 = g^t$, $C^*_3 = \mathfrak{m}^*_b \cdot \hat{e}(h_1, g_2)^t$ and

$$C^*_2 \;=\; \left(C^*_{2,i} = F_i(ID^*_i)^t\right)_{i=1,\ldots,\pi(\ell^*)} \;=\; \left(C'^*_{2,i} = F'_i(P^*_i)^t\right)_{i \in \overline{\mathrm{W}}(P^*)} .$$

Since $\mathcal{B}$ sets $C'^*_1 = C^*_1$, $C'^*_2 = C^*_2$ and $C'^*_3 = C^*_3$, it follows that C'^*_1, C'^*_2 and C'^*_3 are of the correct form. To show that C^*_4 is correctly formed, notice that $u'_{i,j} = g_1^{\alpha_{i,j}}$ for indices $i \in \mathrm{W}(P^*)$ and $j = 0, \ldots, n$. Thus, $C'^*_{4,i,j} = (C^*_1)^{\alpha_{i,j}} = {g_1}^{t\,\alpha_{i,j}} = (g_1^{\alpha_{i,j}})^t = {u'_{i,j}}^t$ as required.

We also need to argue that $\mathcal{B}$ does not query its key derivation oracle on any identities that are considered illegal in the IND-ID-CPA game when its guess for $\mathrm{W}(P^*)$ is correct. Illegal identities are the challenge identity $ID^* = P^*|_{\overline{\mathrm{W}}(P^*)}$ or any ancestors of it, i.e. any $ID^*|_{\le \ell}$ for $\ell \le \ell^*$. Adversary $\mathcal{B}$ only makes such queries when $\mathcal{A}$ queries its key derivation oracle on an identity $ID' = (ID'_1, \ldots, ID'_{\ell'})$ such that $\ell' \le \ell^*$ and $ID'_i = P^*_i$ for all $i \in \overline{\mathrm{W}}(P^*)|_{\le \ell'}$. By our matching definition, this would mean that $ID' \in_* P^*$, which is illegal in the IND-WID-CPA game as well. Note that, whenever $\ell' > \ell^*$, we always have that $|ID| > |ID^*|$ since $\mathrm{W}(\hat{P})|_{> \ell^*} = \emptyset$.

To conclude the proof, we notice that the success probability of $\mathcal{B}$ is at least that of $\mathcal{A}$ when its guess for $\mathrm{W}(P^*)$ is correct. Let ϵ be the probability that $\mathcal{A}$ wins the IND-WID-CPA game. Thus, it follows that the overall success probability of $\mathcal{B}$ winning the IND-ID-CPA game is at least $\epsilon' \ge \epsilon / 2^L$. ▮

Note that the proof above loses a factor of 2^L in the security reduction. This limits the secure use of the scheme in practice to very small (logarithmic) hierarchy depths, but this was already the case for the *Wa-HIBE* scheme as well which loses a factor $(nq_{\mathrm{K}})^L$ in its reduction to the BDDH problem. In addition, we only lose an additional factor of L^2 when allowing only patterns with a single sequence of consecutive wildcards, for example $(ID_1, *, *, *, ID_5)$ or $(ID_1, *, *)$. In the selective-identity notion, there is no need to guess the challenge pattern, so we do not lose any tightness with respect to the *Wa-HIBE* scheme.

6 More Efficient Constructions in the Random Oracle Model

In this section, we present two alternative schemes based on the Boneh-Boyen [BB04] and Boneh-Boyen-Goh [BBG05] HIBE schemes that perform better in terms of efficiency. In particular, unlike the $\mathcal{Wa}$-$\mathcal{WIBE}$ scheme, their efficiency is independent of the bit length n of identity strings: adding a wildcard to the recipient pattern only requires two extra exponentiations and two extra group elements in the ciphertext, as opposed to $n+1$ of these in the $\mathcal{Wa}$-$\mathcal{WIBE}$ scheme. Just like their underlying HIBE schemes however, they can be proved secure only in the weaker selective-identity setting. As observed for the case of IBE and HIBE schemes by Boneh, Boyen and Goh [BB04, BBG05], these schemes can be made fully secure in the random oracle model (but losing a factor exponential in L in tightness) by applying a hash function to the identity strings. We first present the two alternative WIBE schemes, and for completeness prove the generic transformation from selective-identity to full security for the case of WIBE schemes in Section 6.3.

6.1 A Construction from Boneh-Boyen's HIBE Scheme

Our first construction in the random oracle model is based on a slight variant of the Boneh-Boyen HIBE scheme [BB04] that we refer to as the $\mathcal{BB}$-$\mathcal{HIBE}$ scheme. It is presented in detail and proved secure in Appendix A. The $\mathcal{BB}$-$\mathcal{WIBE}$ scheme that we derive from it works as follows.

Setup. The trusted authority chooses random generators g_1, g_2 from $\mathbb{G}^*$, a random $\alpha \in \mathbb{Z}_p$ and sets $h_1 \leftarrow g_1^\alpha$. Next, it picks random elements $u_{1,0}, \ldots, u_{L,0}, u_{1,1}, \ldots, u_{L,1}$ from $\mathbb{G}^*$ and sets $h_2 \leftarrow g_2^\alpha$. The master public key is $mpk = (g_1, h_1, g_2, u_{1,0}, \ldots, u_{L,0}, u_{1,1}, \ldots, u_{L,1})$. The corresponding master secret key is $msk = h_2$.

Key Derivation. A user's identity is given by a vector $ID = (ID_1, \ldots, ID_\ell)$ where each ID_i is an element in $\mathbb{Z}_p$. To compute the decryption key for identity ID from the master secret key, first one chooses random values $r_i \xleftarrow{\$} \mathbb{Z}_p$ for $i = 1, \ldots, \ell$, then the private key d_{ID} is constructed as

$$(a_0, a_1, \ldots, a_\ell) \;=\; \left(h_2 \prod_{i=1}^{\ell} \left(u_{i,0} \cdot u_{i,1}^{ID_i} \right)^{r_i} \;,\; g_1^{r_1} \;,\; \ldots \;,\; g_1^{r_\ell} \right) .$$

Notice that, as required, the secret key for identity $ID = (ID_1, \ldots, ID_\ell)$ can be computed from the secret key $(a_0, a_1, \ldots, a_{\ell-1})$ of the parent $(ID_1, \ldots, ID_{\ell-1})$ by choosing a random $r_\ell \xleftarrow{\$} \mathbb{Z}_p$ and outputting

$$d_{ID} \;=\; \left(a_0 \cdot \left(u_{\ell,0} \cdot u_{\ell,1}^{ID_\ell} \right)^{r_\ell} \;,\; a_1 \;,\; \ldots \;,\; a_{\ell-1} \;,\; g_1^{r_\ell} \right)$$

Encryption. To create a ciphertext of message $\mathfrak{m} \in \mathbb{G}_T$ intended for all identities matching pattern $P = (P_1, \ldots, P_\ell)$, where $1 \le \ell \le L$, the sender chooses $t \xleftarrow{\$} \mathbb{Z}_p$ and outputs the ciphertext $C = (P, C_1, C_2, C_3)$, where

$$\begin{array}{llll} C_1 \;\leftarrow\; g_1^t & & C_2 \;\leftarrow\; \left(C_{2,i} = (u_{i,0} \cdot u_{i,1}^{ID_i})^t \right)_{i \in \overline{\mathrm{W}}(P)} \\ C_3 \;\leftarrow\; \mathfrak{m} \cdot \hat{e}(h_1, g_2)^t & & C_4 \;\leftarrow\; \left(C_{4,i,j} = u_{i,j}^t \right)_{i \in \mathrm{W}(P),\, j=0,1} \end{array}$$

Decryption. If the receiver is the root authority (i.e., the empty identity $ID = \varepsilon$) holding the master key $msk = h_2$, then he can recover the message by computing $C_3 / \hat{e}(C_1, h_2)$. Any other receiver with identity $ID = (ID_1, \ldots, ID_\ell)$ matching the pattern P to which the ciphertext was created (i.e., $ID \in_* P$) can decrypt the ciphertext $C = (P, C_1, C_2, C_3, C_4)$ as follows. Let $d_{ID} = (a_0,$

$a_1, \ldots, a_\ell)$ be the decryption key for the receiver with identity ID. He recovers the message by computing

$$C'_{2,i} \;\leftarrow\; \begin{cases} C_{2,i} & \text{if } i \in \overline{\mathrm{W}}(P) \\ C_{4,i,0} \cdot C_{4,i,1}^{ID_i} & \text{if } i \in \mathrm{W}(P)|_{\leq \ell} \end{cases}$$

$$\mathfrak{m} \;\leftarrow\; C_3 \cdot \frac{\prod_{i=1}^{\ell} \hat{e}(a_i,\ C'_{2,i})}{\hat{e}(C_1,\ a_0)} \ .$$

In terms of efficiency, the *BB-WIBE* scheme easily outperforms the *Wa-WIBE* scheme: the master public key contains $2L + 3$ group elements. Encryption to a recipient pattern of length ℓ and w wildcards involves $\ell + w + 2$ (multi-)exponentiations and produces ciphertexts containing $\ell + w + 2$ group elements, or $2L + 2$ of each of these in the worst case that $\ell = w = L$. Decryption requires the computation of $\ell + 1$ pairings, just like the *Wa-WIBE* scheme.

The *BB-WIBE* scheme can actually be seen as a close relative to the *Wa-WIBE* scheme, with the functions $F_i(ID_i)$ being defined as

$$F_i(ID_i) \;=\; u_{i,0} \cdot u_{i,1}^{ID_i} \ .$$

Its security properties are different though: the *BB-WIBE* scheme can be proved secure in the selective-identity model only. We reduce its security to that of the *BB-HIBE* scheme, which on its turn is proved IND-sID-CPA-secure under the BDDH assumption in Appendix A. The proof of the theorem below is very analagous to that of Theorem 5.1, and hence omitted. One important difference with Theorem 5.1 is that the reduction from the *BB-HIBE* scheme is tight: because we prove security in the selective-identity model, we do not lose a factor 2^L due to having to guess the challenge pattern upfront.

Theorem 6.1 If the *BB-HIBE* scheme with hierarchy depth L is $(t, q_{\mathrm{K}}, \epsilon)$ IND-sID-CPA-secure, then the *BB-WIBE* scheme of depth L is $(t', q'_{\mathrm{K}}, \epsilon')$ IND-sWID-CPA-secure for all

$$t' \;\leq\; t - 2L(1 + q_{\mathrm{K}}) \cdot t_{\mathrm{exp}} \ , \quad q'_{\mathrm{K}} \;\leq\; q_{\mathrm{K}} \quad \text{and } \epsilon' \;\geq\; \epsilon \ ,$$

where t_{exp} is the time required to compute an exponentiation in $\mathbb{G}$.

6.2 A Construction from Boneh-Boyen-Goh's HIBE Scheme

In this section we describe a WIBE scheme with shorter ciphertexts, especially when the recipient pattern contains few wildcards. When encrypting to a pattern of length ℓ with w wildcards, a ciphertext of the *BB-WIBE* scheme contains $\ell + w + 2$ group elements. The HIBE scheme of Boneh-Boyen-Goh [BBG05] offers constant-size ciphertexts (i.e, independent of the level ℓ of the recipient identity) at the cost of being secure only under the stronger decisional L-BDHI assumption. Based on this scheme, we build the *BBG-WIBE* scheme that offers ciphertexts of length $w + 3$ group elements and is secure under the same decisional L-BDHI assumption. The scheme is the following:

Setup. The trusted authority chooses random generators $g_1, g_2, u_0, \ldots, u_L$ from $\mathbb{G}^*$, a random $\alpha \in \mathbb{Z}_p$ and sets $h_1 \leftarrow g^\alpha$ and $h_2 \leftarrow g_2^\alpha$. The master public key is $mpk = (g_1, g_2, h_1, u_0, \ldots, u_L)$. The corresponding master secret key is $msk = h_2$.

Key Derivation. The scheme assumes that a user's identity is given by a vector $ID = (ID_1, \ldots, ID_\ell)$ of elements in $\mathbb{Z}_p^*$.[1]. To compute the decryption key for identity ID from the master secret

[1]This can be easily generalised to the case on which the identities are vectors of n bit strings by first hashing each component $ID_i \in \mathbb{Z}_p^*$ using a collision resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_p^*$

key, first a random $r \xleftarrow{\$} \mathbb{Z}_p$ is chosen, then the private key is constructed as

$$d_{ID} \;=\; (a_0, a_{\ell+1}, \ldots, a_L, a_{L+1}) \;=\; \left(h_2 \left(u_0 \prod_{i=1}^{\ell} u_i^{ID_i} \right)^r \;,\; u_{\ell+1}^r \;,\; \ldots \;,\; u_L^r \;,\; g_1^r \right)$$

The secret key for identity $ID = (ID_1, \ldots, ID_\ell)$ can be computed from the secret key $(a_0, a_\ell, \ldots, a_{L+1})$ of its parent $(ID_1, \ldots, ID_{\ell-1})$ by choosing a random $r' \xleftarrow{\$} \mathbb{Z}_p$ and outputting

$$d_{ID} = \left(a_0 \cdot a_\ell^{ID_\ell} \cdot \left(u_0 \prod_{i=1}^{\ell} u_i^{ID_i} \right)^{r'} \;,\; a_{\ell+1} \cdot u_{\ell+1}^{r'} \;,\; \ldots \;,\; a_L \cdot u_L^{r'} \;,\; a_{L+1} \cdot g_1^{r'} \right)$$

Encryption. To create a ciphertext of message $\mathfrak{m} \in \mathbb{G}_T$ intended for all identities matching pattern $P = (P_1, \ldots, P_\ell)$, where $\ell \leq L$, the sender chooses $t \xleftarrow{\$} \mathbb{Z}_p$ and outputs the ciphertext $C = (P, C_1, C_2, C_3, C_4)$, where

$$\begin{array}{ll} C_1 \leftarrow g_1^t & C_2 \leftarrow \left(u_0 \prod_{i \in \overline{\mathrm{W}}(P)} u_i^{P_i} \right)^t \\ C_3 \leftarrow \mathfrak{m} \cdot \hat{e}(h_1, g_2)^t & C_4 \leftarrow \left(C_{4,i} = u_i^t \right)_{i \in \mathrm{W}(P)} . \end{array}$$

Decryption. If the receiver is the root authority (i.e., the empty identity $ID = \varepsilon$) holding the master key $msk = h_2$, then he can recover the message by computing $C_3 / \hat{e}(C_1, h_2)$. Any other receiver with identity $ID = (ID_1, \ldots, ID_\ell)$ matching the pattern P to which the ciphertext was created (i.e., $ID \in_* P$) can decrypt the ciphertext $C = (P, C_1, C_2, C_3, C_4)$ as follows. Let $d_{ID} = (a_0, \ldots, a_{L+1})$ be the decryption key for the receiver with identity ID. He recovers the message by computing

$$\begin{aligned} C_2' \;&\leftarrow\; C_2 \cdot \prod_{i \in \mathrm{W}(P)|_{\leq \ell}} C_{4,i}^{P_i} \\ \mathfrak{m} \;&\leftarrow\; C_3 \cdot \frac{\hat{e}(C_2', a_{L+1})}{\hat{e}(C_1, a_0)} \end{aligned}$$

The fact that decryption works can be seen as follows. Since $ID \in_* P$, we have that $P_i = ID_i$ for all $i \in \overline{\mathrm{W}}(P)$. We then have that:

$$\begin{aligned} \frac{\hat{e}(C_2', a_{L+1})}{\hat{e}(C_1, a_0)} \;&=\; \frac{\hat{e}\left(C_2 \cdot \prod_{i \in \mathrm{W}(P)|_{\leq \ell}} C_{4,i}^{ID_i} \;,\; g_1^r \right)}{\hat{e}\left(g_1^t \;,\; h_2 (u_0 \prod_{i=1}^{\ell} u_i^{ID_i})^r \right)} \\ &=\; \frac{\hat{e}\left((u_0 \prod_{i \in \overline{\mathrm{W}}(P)} u_i^{P_i})^t \cdot \prod_{i \in \mathrm{W}(P)|_{\leq \ell}} (u_i^t)^{ID_i} \;,\; g_1^r \right)}{\hat{e}\left(g_1^t \;,\; h_2 \right) \cdot \hat{e}\left(g_1^t \;,\; (u_0 \prod_{i=1}^{\ell} u_i^{ID_i})^r \right)} \\ &=\; \frac{\hat{e}\left((u_0 \prod_{i=1}^{\ell} u_i^{ID_i})^t \;,\; g_1^r \right)}{\hat{e}\left(g_1^t \;,\; h_2 \right) \cdot \hat{e}\left(g_1^t \;,\; (u_0 \prod_{i=1}^{\ell} u_i^{ID_i})^r \right)} \\ &=\; \frac{1}{\hat{e}\left(h_1, g_2 \right)^t} \;. \end{aligned}$$

The $\mathcal{BBG}$-$\mathcal{WIBE}$ scheme is significantly more efficient than the $\mathcal{W}a$-$\mathcal{WIBE}$ and $\mathcal{BB}$-$\mathcal{WIBE}$ schemes in terms of decryption, and also offers more efficient encryption and shorter ciphertexts when the recipient pattern contains few wildcards. More precisely, the master public key contains $L+4$ group elements. Encryption to a recipient pattern of length ℓ with w wildcards involves $w+3$ (multi-)exponentiations and $w+3$ group elements in the ciphertext, or $L+3$ of these in the worst case that $\ell = w = L$. Decryption requires the computation of two pairings, as opposed to $\ell+1$ of these for the $\mathcal{W}a$-$\mathcal{WIBE}$ and $\mathcal{BB}$-$\mathcal{WIBE}$ schemes. We prove the security of the $\mathcal{BBG}$-$\mathcal{WIBE}$ scheme in the selective-identity model by reducing to the security of the $\mathcal{BBG}$-$\mathcal{HIBE}$ scheme that is recalled in Appendix B, rather than to the underlying decisional L-BDHI assumption directly.

Theorem 6.2 If the $\mathcal{BBG}$-$\mathcal{HIBE}$ scheme is $(t, q_{\mathrm{K}}, \epsilon)$ IND-sID-CPA-secure, then the $\mathcal{BBG}$-$\mathcal{WIBE}$ scheme presented above is $(t', q'_{\mathrm{K}}, \epsilon')$ IND-sWID-CPA-secure for all

$$t' \leq t - L(1+2q_{\mathrm{K}}) \cdot t_{\mathrm{exp}} \ , \quad q'_{\mathrm{K}} \leq q_{\mathrm{K}} \ , \quad \text{and } \epsilon' \geq \epsilon \ ,$$

where t_{exp} is the time it takes to perform an exponentiation in $\mathbb{G}$.

Proof: The proof of Theorem 6.2 is almost identical to the proof given for Theorem 5.1. We present it here for completeness. As before we assume that there exist an adversary $\mathcal{A}$ that breaks the IND-sWID-CPA-security of the $\mathcal{BBG}$-$\mathcal{WIBE}$ scheme and then we show how to efficiently build another adversary $\mathcal{B}$ that, using $\mathcal{A}$ as a black box, manages to break the IND-sID-CPAsecurity of the $\mathcal{BBG}$-$\mathcal{HIBE}$ scheme.

Algorithm $\mathcal{B}$ begins by running $\mathcal{A}$ to obtain a challenge pattern $P^* = (P^*_1, \ldots, P^*_{\ell^*})$. Define a projection function $\pi : \{1, \ldots, L\} \to \{0, \ldots, L\}$ where

$$\begin{aligned} \pi(i) &= 0 && \text{if } i \in \mathrm{W}(P^*) \\ &= i - |\mathrm{W}(P^*)|_{\leq i}| && \text{otherwise .} \end{aligned}$$

The projection function is such that identities at level $i \in \overline{\mathrm{W}}(P^*)$ in the WIBE tree will be mapped onto level $\pi(i)$ in the HIBE tree. $\mathcal{B}$ outputs $ID^* = P^*|_{i \in \overline{\mathrm{W}}(P^*)}$ as its own challenge identity and gets a master public key $mpk_{\mathrm{H}} = (g_1, g_2, h_1, u_0, \ldots, u_L)$ for the $\mathcal{BBG}$-$\mathcal{HIBE}$ scheme in return. It runs adversary $\mathcal{A}$ on master public key $mpk_{\mathrm{W}} = (g_1, g_2, h_1, u_0, u'_1, \ldots, u'_L)$, where for all $1 \leq i \leq L$, the values u'_i are generated as:

$$\begin{aligned} u'_i &\leftarrow g_1^{\alpha_i} && \text{if } i \in \mathrm{W}(\hat{P}), \text{ where } \alpha_i \xleftarrow{\$} \mathbb{Z}_p \\ &\leftarrow u_{\pi(i)} && \text{otherwise.} \end{aligned}$$

Notice that mpk_{W} is distributed exactly as it would be if produced by the real Setup algorithm of the $\mathcal{BBG}$-$\mathcal{WIBE}$ scheme.

During the first phase, $\mathcal{B}$ has to answer all the key derivation queries $ID = (ID_1, \ldots, ID_\ell)$ that $\mathcal{A}$ is allowed to ask. For that, $\mathcal{B}$ queries its own key derivation oracle on identity $ID' = ID|_{i \in \overline{\mathrm{W}}(P^*)|_{\leq \ell}}$ to get the key $d'_{ID'} = (a'_0, a'_{\ell'+1} \ldots, a'_{L+1})$ where $\ell' = |ID'|$. Next, $\mathcal{B}$ computes the key $d_{ID} = (a_0, a_{\ell+1}, \ldots, a_{L+1})$ as

$$\begin{aligned} a_0 &\leftarrow a'_0 \cdot \prod_{i \in \mathrm{W}(P^*)|_{\leq \ell}} (a'_{L+1})^{\alpha_i \cdot ID_i} \\ a_i &\leftarrow \begin{cases} (a'_{L+1})^{\alpha_i} & \text{if } i \in \mathrm{W}(P^*)|_{> \ell} \\ a'_{\pi(i)} & \text{otherwise} \end{cases} \qquad \text{for } i = \ell+1, \ldots, L \\ a'_{L+1} &\leftarrow a_{L+1} \ . \end{aligned}$$

When at the end of its first phase $\mathcal{A}$ outputs challenge messages $\mathfrak{m}_0^*, \mathfrak{m}_1^*$, $\mathcal{B}$ also ends its first phase with the same messages. Let $C'^* = (C_1'^*, C_2'^*, C_3'^*)$ be the challenge ciphertext that $\mathcal{B}$ receives in return from its challenger, meaning that C'^* is an encryption of $\mathfrak{m}_b^*$ with respect to the identity ID^*, where b is the secret bit chosen at random by the challenger. $\mathcal{B}$ sets $C_1^* \leftarrow C_1'^*$, $C_2^* \leftarrow C_2'^*$, $C_3^* \leftarrow C_3'^*$ and $C_4^* \leftarrow \left(C_{4,i}^* = (C_1^*)^{\alpha_i}\right)_{i \in \mathrm{W}(P^*)}$, and feeds the ciphertext $C^* = (P^*, C_1^*, C_2^*, C_3^*, C_4^*)$ to $\mathcal{A}$ as the input for its second phase. During the second phase, $\mathcal{A}$ is then allowed to issue more key derivation queries, which are answered by $\mathcal{B}$ exactly as in the first phase. When $\mathcal{A}$ outputs a bit b', $\mathcal{B}$ outputs the same bit b' and stops.

By arguments similar to those given in the proof of Theorem 5.1, one can see that $\mathcal{B}$ provides a perfectly simulated environment for $\mathcal{A}$ and that $\mathcal{B}$ does not query for the key of the challenge identity ID^* or any of its parents. Hence, $\mathcal{B}$ wins the game whenever $\mathcal{A}$ does. The running time of $\mathcal{B}$ is that of $\mathcal{A}$ plus the time needed for at most $2L$ exponentiations for each key derivation query and at most L exponentiations to compute the challenge ciphertext. ▮

6.3 From Selective-Identity to Full Security

As observed by Boneh-Boyen [BB04] for the case of IBE schemes and by Boneh-Boyen-Goh [BBG05] for the case of HIBE schemes, any HIBE scheme $\mathcal{HIBE}$ that is selective-identity secure can be transformed into a HIBE scheme $\mathcal{HIBE}'$ that is IND-sID-CPA-secure in the random oracle model. The transformation only works for small hierarchy depths though, since the proof loses a factor $O(q_{\mathrm{H}}^L)$ in reduction tightness. We show here that the same transformation works for the case of WIBE schemes at the cost of a factor $(q_{\mathrm{H}} + 1)^L$ in reduction tightness.

Let $\mathcal{WIBE}$ be a WIBE scheme with maximum hierarchy depth L. The idea of the transformation is to replace every pattern (or identity) $P = (P_1, \ldots, P_\ell)$ at key derivation or encryption with the pattern $P' = (P_1', \ldots, P_\ell')$ where

$$P_i' \;\leftarrow\; \begin{cases} \mathsf{H}_i(P_i) & \text{if } P_i \neq * \\ * & \text{otherwise,} \end{cases}$$

where H_i, $1 \le i \le L$ are independent random oracles mapping arbitrary bit strings into an appropriate range $\mathcal{ID}$ corresponding to the identity space of $\mathcal{WIBE}$. (These L independent random oracles are easily constructed from a single random oracle $\mathsf{H}(\cdot)$, e.g. by setting $\mathsf{H}_i(\cdot) = \mathsf{H}(i\|\cdot)$.) We refer to the scheme thus obtained as $\mathcal{WIBE}_{\mathsf{H}}$ and prove the following statement about its security.

Theorem 6.3 If $\mathcal{WIBE}$ is a $(t, q_{\mathrm{K}}, \epsilon)$ IND-sWID-CPA-secure WIBE scheme of depth L with identity space $\mathcal{ID}$, then the $\mathcal{WIBE}_{\mathsf{H}}$ scheme described above is $(t', q_{\mathrm{K}}', q_{\mathrm{H}}', \epsilon')$ IND-WID-CPA-secure in the random oracle model for all

$$t' \;\le\; t\;, \quad q_{\mathrm{K}}' \;\le\; q_{\mathrm{K}} \quad \text{and } \epsilon' \;\ge\; (L+1)(q_{\mathrm{H}}'+1)^L \cdot \epsilon + \frac{{q_{\mathrm{H}}'}^2}{|\mathcal{ID}|}\;.$$

Proof: Assume there is an adversary $\mathcal{A}$ breaking the full security of the $\mathcal{WIBE}_{\mathsf{H}}$ scheme, we present an adversary $\mathcal{B}$ that uses $\mathcal{A}$ as a black box and breaks the selective-ID security of the underlying $\mathcal{WIBE}$ scheme.

In a preliminary phase, $\mathcal{B}$ guesses $\hat{\ell} \xleftarrow{\$} \{0, \ldots, L\}$ and $\hat{ctr}_i \xleftarrow{\$} \{0, \ldots, q_{\mathrm{H}}'\}$ for all $1 \le i \le \hat{\ell}$. It then chooses a pattern $\hat{P}' = (\hat{P}_1', \ldots, \hat{P}_{\hat{\ell}}')$ by setting $\hat{P}_i' \leftarrow *$ if $\hat{ctr}_i = 0$ and choosing $\hat{P}_i' \xleftarrow{\$} \mathcal{ID}$ otherwise. $\mathcal{B}$ outputs $\hat{P}'$ as its challenge pattern, and gets public key mpk in return. It runs $\mathcal{A}$ on the same public key mpk and responds to its oracle queries as follows:

- $\mathsf{H}_i(ID_i)$: $\mathcal{B}$ keeps initially empty associative arrays $T_i[\cdot]$ and counters ctr_i that are initialized to zero for $1 \leq i \leq L$. If $T_i[ID_i]$ is undefined, $\mathcal{B}$ increases ctr_i. If $ctr_i = \hat{ctr}_i$, it sets $\hat{P}_i \leftarrow ID_i$ and $T_i[ID_i] \leftarrow \hat{P}'_i$; otherwise, it chooses $T_i[ID_i] \stackrel{\$}{\leftarrow} \mathcal{ID}$. $\mathcal{B}$ returns $T_i[ID_i]$ as the random oracle response to $\mathcal{A}$.

- $\mathsf{KeyDer}(ID = (ID_1, \ldots, ID_\ell))$: $\mathcal{B}$ simulates additional random oracle queries $ID'_i \leftarrow \mathsf{H}_i(ID_i)$ for $1 \leq i \leq \ell$ and lets $ID' = (ID'_1, \ldots, ID'_\ell)$. If $ID' \in_* \hat{P}$ then $\mathcal{B}$ aborts. If $\mathsf{H}_i(ID_i) = \hat{P}'_i$ while $ID_i \neq \hat{P}_i$ for some $1 \leq i \leq \ell$, then $\mathcal{B}$ also aborts. Otherwise, it queries ID' from its own key derivation oracle and forwards the resulting key to $\mathcal{A}$.

Eventually, $\mathcal{A}$ outputs its challenge pattern $P^* = (P^*_1, \ldots, P^*_{\ell^*})$ and challenge messages $\mathfrak{m}^*_0, \mathfrak{m}^*_1$. $\mathcal{B}$ performs additional random oracle queries $\mathsf{H}_i(P^*_i)$ for all $i \in \overline{\mathrm{W}}(P^*)$. If $\ell^* \neq \hat{\ell}$, $\mathrm{W}(P^*) \neq \mathrm{W}(\hat{P})$ or $\mathsf{H}(P^*_i) \neq \hat{P}_i$ for some $i \in \overline{\mathrm{W}}(P^*)$, then $\mathcal{B}$ aborts. Otherwise, it submits $\mathfrak{m}^*_0, \mathfrak{m}^*_1$ as its own challenge messages and forwards the challenge ciphertext C^* that it gets in return to $\mathcal{A}$. During the second phase, $\mathcal{B}$ responds to $\mathcal{A}$'s oracle queries exactly as during the first phase. When $\mathcal{A}$ outputs a bit b', $\mathcal{B}$ outputs the same bit b'.

It is easy to see that $\mathcal{B}$'s simulation of $\mathcal{A}$'s environment is perfect as long as it doesn't abort and that $\mathcal{B}$ wins the game whenever $\mathcal{A}$ does. The probability that $\mathcal{B}$ aborts during one of $\mathcal{A}$'s decryption queries is bounded by the probability that $\mathcal{A}$ manages to find a collision in one of the random oracles H_i, which is at most $q'_{\mathrm{H}}(q'_{\mathrm{H}} - 1)/(2|\mathcal{ID}|) \leq {q'_{\mathrm{H}}}^2/|\mathcal{ID}|$. The probability that $\mathcal{B}$ does not abort in the final stage of the game is at least $1/(L+1)(q_{\mathrm{H}}+1)^L$ due to the random guesses of $\hat{\ell}$ from $\{0, \ldots, L\}$ and of $\hat{ctr}_i$ from $\{0, \ldots, q'_{\mathrm{H}}\}$. Therefore $\mathcal{B}$'s advantage in winning the game is at least

$$\epsilon \;\geq\; \frac{\epsilon'}{(L+1)(q'_{\mathrm{H}}+1)^L} - \frac{{q'_{\mathrm{H}}}^2}{|\mathcal{ID}|} \;,$$

from which the theorem follows. ▮

7 Chosen-Ciphertext Security

In this section, for completeness, and to avoid making any unsubstantiated claims, we present an adaptation of the result of Canetti-Halevi-Katz [CHK04] to obtain chosen-ciphertext security for WIBE schemes. We show that we may use a IND-WID-CPA-secure WIBE of depth $2L+2$ and a strongly unforgeable signature scheme ($\mathsf{SigGen}, \mathsf{Sign}, \mathsf{Verify}$) to construct an IND-WID-CCA-secure WIBE of depth L.

Definition 7.1 A signature scheme is a triple of algorithms ($\mathsf{SigGen}, \mathsf{Sign}, \mathsf{Verify}$) where

- SigGen takes no input (except for an implicit security parameter) and outputs a signing key sk and a verification key vk;

- Sign takes as input a signing key sk and a message $m \in \{0,1\}^*$ and outputs a signature $\sigma \in \{0,1\}^*$;

- and Verify takes as input a verification key vk, a message $m \in \{0,1\}^*$ and a signature $\sigma \in \{0,1\}^*$ and outputs either `valid` or `invalid`.

For correctness we require that for all $(sk, vk) \stackrel{\$}{\leftarrow} \mathsf{SigGen}$, for all $m \in \{0,1\}^*$ and $\sigma \stackrel{\$}{\leftarrow} \mathsf{Sign}(sk, m)$, we have that $\mathsf{Verify}(vk, m, \sigma) = $ `valid` with probability one.

Definition 7.2 A signature scheme $(\mathsf{SigGen}, \mathsf{Sign}, \mathsf{Verify})$ is (t, ϵ) strongly one-time secure, if no probabilistic adversary $\mathcal{A}$ running in time at most t wins the following game with probability more than ϵ:

1. The challenger generates a key pair $(sk^*, vk^*) \xleftarrow{\$} \mathsf{SigGen}$.
2. The attacker executes $\mathcal{A}$ on input vk^* until it outputs a message m^*.
3. The challenger computes $\sigma^* \xleftarrow{\$} \mathsf{Sign}(sk, m^*)$ and returns σ^* to $\mathcal{A}$. $\mathcal{A}$ terminates by outputting a pair (m, σ).

The attacker wins the game if $\mathsf{Verify}(vk, m, \sigma) = \mathtt{valid}$ and $(m, \sigma) \neq (m^*, \sigma^*)$.

We will also make liberal use of an 'encoding function' Encode. For a WIBE scheme with identity space $\mathcal{ID}^L$, this function will have different actions depending on its input. We assume that $\mathcal{ID}$ contains at least two different elements; for simplicity we assume that $\{0, 1\} \subseteq \mathcal{ID}$. For any identity $ID = (ID_1, \ldots, ID_\ell) \in \mathcal{ID}^{\leq L}$, we define

$$\mathsf{Encode}(ID) = (0, ID_1, \ldots, 0, ID_\ell) .$$

We will also use this encoding function with two arguments to denote

$$\mathsf{Encode}(ID, vk) = (0, ID_1, \ldots, 0, ID_k, 1, vk) .$$

Given a WIBE scheme $\mathcal{WIBE} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ of depth $2L + 2$ with identity space $\mathcal{ID}^{\leq L}$, consider the following WIBE scheme $\mathcal{WIBE}' = (\mathsf{Setup}, \mathsf{KeyDer}', \mathsf{Enc}', \mathsf{Dec}')$ of depth L.

Key Derivation. The secret key of identity $ID = (ID_1, \ldots, ID_\ell)$ under $\mathcal{WIBE}'$ is the secret key corresponding to identity $\mathsf{Encode}(ID) = (0, ID_1, \ldots, 0, ID_\ell)$ under $\mathcal{WIBE}$.

Encryption. To encrypt a message $\mathfrak{m}$ under a pattern P and using a master public key mpk, the following steps are performed: First, we generate a signature key pair $(sk, vk) \xleftarrow{\$} \mathsf{SigGen}$. Then we compute $C \xleftarrow{\$} \mathsf{Enc}(mpk, \mathsf{Encode}(P, vk), m)$ and $\sigma \xleftarrow{\$} \mathsf{Sign}(sk, C)$. The final ciphertext is the tuple (vk, C, σ).

Decryption. To decrypt a ciphertext (vk, C, σ) using a private key d_{ID} for an identity ID, first check that $\mathsf{Verify}(vk, C, \sigma) = \mathtt{valid}$. If not, output $\perp$. Otherwise, compute $d = \mathsf{KeyDer}(d_{ID}, (1, vk))$ and output $\mathsf{Dec}(d, C)$. Note that in this case d is the decryption for the identity $\mathsf{Encode}(ID, vk)$ in $\mathcal{WIBE}$.

Theorem 7.3 If $\mathcal{WIBE}$ is $(t, q_{\mathrm{K}}, \epsilon)$ IND-WID-CPA-secure and $(\mathsf{SigGen}, \mathsf{Sign}, \mathsf{Verify})$ is $(t_{\mathrm{s}}, \epsilon_{\mathrm{s}})$ strongly one-time secure, then $\mathcal{WIBE}'$ is $(t', q'_{\mathrm{K}}, q'_{\mathrm{D}}, \epsilon')$ IND-WID-CCA-secure for all

$$\begin{aligned} t' &\leq \min(t, t_{\mathrm{s}}) - q'_{\mathrm{K}} \cdot t_{\mathsf{KeyDer}} - q'_{\mathrm{D}} \cdot (t_{\mathsf{KeyDer}} + t_{\mathsf{Verify}} + t_{\mathsf{Dec}}) \\ q'_{\mathrm{K}} &\leq q_{\mathrm{K}} - q'_{\mathrm{D}} \\ \epsilon' &\geq \epsilon + \epsilon_{\mathrm{s}} \end{aligned}$$

where t_{KeyDer}, t_{Dec} and t_{Verify} are the running times of the KeyDer, Dec and Verify algorithms, respectively.

Proof: The proof closely follows that of [CHK04]. Let $\mathcal{A}$ be an IND-WID-CCA adversary against the $\mathcal{WIBE}'$ scheme. Suppose P^* is the challenge pattern that $\mathcal{A}$ chooses and (vk^*, C^*, σ^*) is the challenge ciphertext that $\mathcal{A}$ receives during an execution of the attack game. Let FORGE be the event that at some point during its execution $\mathcal{A}$ queries the decryption oracle on an identity $ID \in_* P^*$ and a ciphertext of the form (vk^*, C, σ) such that $\mathsf{Verify}(vk^*, C, \sigma) = \mathtt{valid}$. Then we have that $\mathcal{A}$'s advantage is

$$\left|\Pr[\,\mathcal{A} \text{ wins}\,] - 1/2\right| \;\leq\; \Pr[\,\textsc{Forge}\,] + \left|\Pr[\,\mathcal{A} \text{ wins} \;:\; \neg\textsc{Forge}\,] - 1/2\right| . \tag{1}$$

Claim 1 $\Pr[\,\textsc{Forge}\,] \;\leq\; \epsilon_{\mathrm{s}}$.

Claim 2 $\left|\,\Pr[\,\mathcal{A} \text{ wins} \;:\; \neg\textsc{Forge}\,] - 1/2\,\right| \;\leq\; \epsilon$.

Proof of Claim 1: We prove the first claim by demonstrating an attacker $\mathcal{B}$ that breaks the one-time security of the signature scheme whenever the event FORGE occurs. In the first phase, $\mathcal{B}$ receives a verification key vk^* from the challenger. It generates $(mpk, msk) \xleftarrow{\$} \mathsf{Setup}$ and executes $\mathcal{A}$ on input mpk, responding to its key derivation and decryption queries using the real KeyDer' and Dec' algorithms, which it can do because it knows msk. It keeps a list of $\mathcal{A}$'s decryption queries $(ID, (vk, C, \sigma))$ for later reference. When $\mathcal{A}$ outputs challenge pattern P^* and challenge messages $\mathfrak{m}_0^*, \mathfrak{m}_1^*$, $\mathcal{B}$ chooses a random bit $b \xleftarrow{\$} \{0,1\}$, computes $C^* \xleftarrow{\$} \mathsf{Enc}(mpk, \mathsf{Encode}(P^*, vk^*), \mathfrak{m}_b^*)$ and requests a signature σ^* on message C^* from its own challenger. Algorithm $\mathcal{B}$ then runs $\mathcal{A}$ on input (vk^*, C^*, σ^*) until it halts, responding to $\mathcal{A}$'s oracle queries as before.

At the end of $\mathcal{A}$'s execution, $\mathcal{B}$ checks whether the list of $\mathcal{A}$'s decryption queries contains an entry $(ID, (vk, C, \sigma))$ such that $ID \in_* P^*$, $vk = vk^*$ and $\mathsf{Verify}(vk, C, \sigma) = \mathtt{valid}$, or in other words, checks whether the event FORGE occurred. Note that in this case $(C, \sigma) \neq (C^*, \sigma^*)$ because $\mathcal{A}$ is not allowed to query the decryption oracle on the challenge ciphertext with an identity $ID \in_* P^*$. $\mathcal{B}$ outputs (C^*, σ^*) as its forgery and wins the game. The running time of $\mathcal{B}$ is that of $\mathcal{A}$ plus the time needed for $q'_{\mathrm{K}} + q'_{\mathrm{D}}$ applications of the KeyDer algorithm, q'_{K} applications of the Verify algorithm and q'_{K} applications of the Dec algorithm. ▮

Proof of Claim 2: To prove the second claim, we show that there exists an IND-WID-CPA attacker $\mathcal{C}$ against $\mathcal{WIBE}$ that uses $\mathcal{A}$ as a subroutine and that has advantage ϵ of winning the game whenever the event FORGE does not occur. Algorithm $\mathcal{C}$, on input a master public key mpk, runs $\mathcal{A}$ on input mpk, answering its oracle queries as follows:

- If $\mathcal{A}$ queries the key extraction oracle on the identity ID, then $\mathcal{C}$ queries its key extraction oracle on the identity $\mathsf{Encode}(ID)$ and returns the resulting key to $\mathcal{A}$.
- If $\mathcal{A}$ queries the decryption oracle on the identity ID and ciphertext (vk, C, σ), then $\mathcal{C}$ checks that $\mathsf{Verify}(vk, C, \sigma) = \mathtt{valid}$. If not, then $\mathcal{C}$ returns $\perp$ to $\mathcal{A}$. If the signature is valid, $\mathcal{C}$ queries its key extraction oracle on the identity $\mathsf{Encode}(ID, vk)$ to receive the decryption key d and returns the output of $\mathsf{Dec}(d, C)$ to $\mathcal{A}$.

When $\mathcal{A}$ outputs challenge pattern P^* and challenge messages $\mathfrak{m}_0^*, \mathfrak{m}_1^*$, $\mathcal{C}$ generates a fresh key pair $(sk^*, vk^*) \xleftarrow{\$} \mathsf{SigGen}$ and outputs $\mathsf{Encode}(P^*, vk^*)$ and $\mathfrak{m}_0^*, \mathfrak{m}_1^*$ as its own challenge pattern and messages. In return, it gets a challenge ciphertext C^* from its challenger. $\mathcal{C}$ computes $\sigma^* \xleftarrow{\$} \mathsf{Sign}(sk, C^*)$

and feeds (vk^*, C^*, σ^*) to $\mathcal{A}$, answering its oracle queries exactly as before. When $\mathcal{A}$ outputs a bit b', $\mathcal{C}$ outputs the same bit b'.

It is not hard to see that $\mathcal{C}$'s simulation of $\mathcal{A}$'s environment is perfect and that $\mathcal{C}$ wins the game whenever $\mathcal{A}$ does *as long as* $\mathcal{C}$ does not make any illegal key derivation queries. We have left to argue why the latter fact is true. First consider the queries that $\mathcal{C}$ makes to respond to $\mathcal{A}$'s key derivation query ID. Let $ID' = \mathsf{Encode}(ID)$ and let $P'^* = \mathsf{Encode}(P^*, vk^*)$. If $|ID'| > |P'^*|$ then ID' can never match P'^*. If $|ID'| = |P'^*|$ then still $ID' \notin_* P'^*$ because ID' and P'^* are different on the next to last level (ID' contains a zero there, while P'^* contains a one). If $|ID'| < |P'^*|$ then the only way to have $ID' \in_* P'^*$ is if also $ID \in_* P^*$, which are illegal queries in $\mathcal{A}$'s game as well.

Second, consider the key derivation queries that $\mathcal{C}$ makes in order to respond to $\mathcal{A}$'s decryption queries. If $\mathcal{A}$ makes decryption query $(ID, (vk, C, \sigma))$, then $\mathcal{C}$ makes a key derivation query for $ID' = \mathsf{Encode}(ID, vk)$. Let $P'^* = \mathsf{Encode}(P^*, vk^*)$. If $vk \neq vk^*$ then definitely $ID' \notin_* P'^*$: either ID' has a zero where P'^* has a one, or they differ on the last level. So let's focus on the case that $vk = vk^*$. If $|ID'| > |P'^*|$ then ID' can never match P'^*. If $|ID'| < |P'^*|$ then the next to last level of ID' contains a one while P'^* contains a zero there, so also in that case $ID' \notin_* P'^*$. If $|ID'| = |P'^*|$, then the only way to have $ID' \notin_* P'^*$ is if also $ID \in_* P^*$, but this case is excluded by the event $\neg$FORGE.

The running time of $\mathcal{C}$ is that of $\mathcal{A}$ plus the time needed to compute q'_{D} applications of the Verify and Dec algorithms and one application of the Sign algorithm. It also performs $q_{\mathrm{K}} = q'_{\mathrm{K}} + q'_{\mathrm{D}}$ queries to its key derivation oracle. ▮

The bound on ϵ' in the statement of Theorem 7.3 follows directly from Equation (1) and the two claims above. The bound on the number of key derivation queries q'_{K} is due to the proof of Claim 2 above. For the bound on t' we have to take into account the running times of both algorithms $\mathcal{B}$ and $\mathcal{C}$ in the proofs of Claims 1 and 2. From the proof of Claim 1 we have that

$$t' \;\leq\; t_1 = t - (q'_{\mathrm{K}} + q'_{\mathrm{D}}) \cdot t_{\mathsf{KeyDer}} - q'_{\mathrm{D}} \cdot t_{\mathsf{Verify}} - q'_{\mathrm{D}} \cdot t_{\mathsf{Dec}}$$

and from the proof of Claim 2 we have that

$$t' \;\leq\; t_2 = t_{\mathrm{s}} - q'_{\mathrm{D}} \cdot t_{\mathsf{Verify}} - q'_{\mathrm{D}} \cdot t_{\mathsf{Dec}} \;.$$

To simultaneously satisfy both equations, we need to upper-bound t' by

$$\min(t_1, t_2) \;\geq\; \min(t, t_{\mathrm{s}}) - q'_{\mathrm{K}} \cdot t_{\mathsf{KeyDer}} - q'_{\mathrm{D}} \cdot (t_{\mathsf{KeyDer}} + t_{\mathsf{Verify}} + t_{\mathsf{Dec}}) \;,$$

where we use that $\min(x - z, y - z) \geq \min(x, y) - z$ and $\min(x - w, y - z) \geq \min(x, y) - w - z$. ▮

One may wonder why we require a $(2L{+}2)$-level IND-WID-CPA-secure WIBE in order to construct an L-level IND-WID-CCA-secure WIBE when the original result of Canetti-Halevi-Katz [CHK04] only required an $(L{+}1)$-level IND-ID-CPA-secure HIBE to construct an L-level IND-ID-CCA-secure HIBE. The construction of [CHK04] encodes every identity string ID as $0||ID$ and every verification key vk as $1||vk$. For a HIBE, the different form of the two types of binary string means that when we use the key extraction oracle to decrypt a ciphertext, we never query the key extraction oracle on an ancestor of the challenge identity. However, if we try and use the same trick to construct a chosen-ciphertext secure WIBE, then it is possible that we will query the key extraction oracle on an identity that matches the challenge pattern because both $0||ID$ and $1||vk$ match the pattern string $*$. Hence, we are forced to place the single bits that identify whether the following binary string is an identity or a verification key into their own levels on the WIBE.

Applying the transformation to $\mathcal{W}a\text{-}\mathcal{WIBE}$. If we apply the above transformation to the (IND-WID-CPA-secure) $\mathcal{W}a\text{-}\mathcal{WIBE}$ scheme described in Section 5 and prove the security of the scheme directly, rather than by applying Theorems 5.1 and 7.3, then we may achieve some small efficiency gains. In particular, if we wish to construct an L-level CCA-secure WIBE scheme, then a naive application of the theorems suggests that we have to start from a $(2L+2)$-level $\mathcal{W}a\text{-}\mathcal{WIBE}$ scheme, meaning that the public parameters for the WIBE consist of $(2L+2)(n+1)+3$ group elements, and that we lose a factor of 2^{2L+2} in the security reduction from the $\mathcal{W}a\text{-}\mathcal{WIBE}$ to the $\mathcal{W}a\text{-}\mathcal{HIBE}$ scheme.

However, if we look at the proof techniques used in the theorems, then we can make some efficiency gains. In particular,

- $L+1$ levels of the WIBE are only used to encode either a zero or a one. This means that the public parameters do not require the $n+1$ group elements required to represent an n-bit identity at those levels; they only require the two group elements that are required to encode a single-bit identity. Hence, the public parameters only require $(L+1)(n+3)+3$ group elements.

- the reduction from the CPA-secure WIBE to the CPA-secure HIBE loses a factor of 2^{2L+2} because we need to guess the positions of the wildcards in the challenge identity. However, in this construction, the wildcards can only occur at L different positions, instead of all $2L+2$ positions. Hence, we actually only lose a factor of 2^L in this reduction.

Unfortunately, we still do require a $(2L+2)$-level instantiation of the $\mathcal{W}a\text{-}\mathcal{WIBE}$ scheme. This implies an important security loss because the proof of security for the $\mathcal{W}a\text{-}\mathcal{HIBE}$ loses a factor of $O((nq_{\mathrm{K}})^{2L+2})$ in the reduction to the BDDH assumption.

Versions of this Paper

An extended abstract of this paper appeared at ICALP 2006 [ACD+06]. The first full version was posted on the IACR ePrint archive in September 2006. This is an updated version of December 2006 that corrects a minor mistake in the proof of Theorem 6.3.

Acknowledgments

We would like to thank James Birkett, Jacob Schuldt, Brent Waters and the anonymous referees of ICALP 2006 for their valuable input. We also thank Mihir Bellare for pointing out the relation between WIBE and fuzzy identity-based encryption. This work was supported in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT. The information in this document reflects only the author's views, is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability. The first two authors were supported in part by France Telecom R&D as part of the contract CIDRE, between France Telecom R&D and École normale supérieure. The fifth author is a Postdoctoral Fellow of the Research Foundation – Flanders (FWO-Vlaanderen), and was supported in part by the Concerted Research Action (GOA) Ambiorics 2005/11 of the Flemish Government.

References

[ACD+06] Michel Abdalla, Dario Catalano, Alex Dent, John Malone-Lee, Gregory Neven, and Nigel Smart. Identity-based encryption gone wild. In Michele Bugliesi, Bart Preneel, Vladimiro

Sassone, and Ingo Wegener, editors, *ICALP 2006: 33rd International Colloquium on Automata, Languages and Programming, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 300–311. Springer-Verlag, Berlin, Germany, July 9–16, 2006. (Cited on pages i and 20.)

[BB04] Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238, Interlaken, Switzerland, May 2–6, 2004. Springer-Verlag, Berlin, Germany. (Cited on pages 2, 3, 4, 6, 11, 15, 22 and 23.)

[BBG05] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456, Aarhus, Denmark, May 22–26, 2005. Springer-Verlag, Berlin, Germany. (Cited on pages 2, 6, 11, 12, 15 and 25.)

[BF03] Dan Boneh and Matthew K. Franklin. Identity based encryption from the Weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003. (Cited on pages 1 and 4.)

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press. (Cited on page 2.)

[BR94] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO'93*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249, Santa Barbara, CA, USA, August 22–26, 1994. Springer-Verlag, Berlin, Germany. (Cited on page 4.)

[CHK04] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222, Interlaken, Switzerland, May 2–6, 2004. Springer-Verlag, Berlin, Germany. (Cited on pages 2, 16, 18 and 19.)

[Coc01] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In Bahram Honary, editor, *Cryptography and Coding, 8th IMA International Conference*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363, Cirencester, UK, December 17–19, 2001. Springer-Verlag, Berlin, Germany. (Cited on page 1.)

[CS06] Sanjit Chatterjee and Palash Sarkar. Trading time for space: Towards an efficient IBE scheme with short(er) public parameters in the standard model. In Dongho Won and Seungjoo Kim, editors, *Information Security and Cryptology – ICISC 2005*, volume 3935 of *Lecture Notes in Computer Science*, pages 424–440. Springer-Verlag, Berlin, Germany, 2006. (Cited on page 8.)

[GS02] Craig Gentry and Alice Silverberg. Hierarchical ID-based cryptography. In Yuliang Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566, Queenstown, New Zealand, December 1–5, 2002. Springer-Verlag, Berlin, Germany. (Cited on page 1.)

[HL02] Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 466–481, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer-Verlag, Berlin, Germany. (Cited on page 1.)

[MSK02] Shigeo Mitsunari, Ryuichi Saka, and Masao Kasahara. A new traitor tracing. *IEICE Transactions*, E85-A(2):481–484, February 2002. (Cited on pages 2 and 3.)

[Nac05] David Naccache. Secure and *practical* identity-based encryption. Cryptology ePrint Archive, Report 2005/369, 2005. `http://eprint.iacr.org/`. (Cited on page 8.)

[Sha85] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53, Santa Barbara, CA, USA, August 19–23, 1985. Springer-Verlag, Berlin, Germany. (Cited on page 1.)

[SOK00] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *SCIS 2000*, Okinawa, Japan, January 2000. (Cited on page 1.)

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473, Aarhus, Denmark, May 22–26, 2005. Springer-Verlag, Berlin, Germany. (Cited on page 6.)

[Wat05] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127, Aarhus, Denmark, May 22–26, 2005. Springer-Verlag, Berlin, Germany. (Cited on pages 2, 6, 7 and 8.)

A The $\mathcal{BB}$-$\mathcal{HIBE}$ Scheme

In this section, we present a variant of the HIBE scheme by Boneh and Boyen in Eurocrypt 2004 [BB04].

Setup. The trusted authority chooses random generators g_1, g_2 from $\mathbb{G}^*$, a random $\alpha \in \mathbb{Z}_p$ and sets $h_1 \leftarrow g_1^\alpha$. Next, it picks random elements $u_{1,0}, \ldots, u_{L,0}, u_{1,1}, \ldots, u_{L,1}$ from $\mathbb{G}^*$ and sets $h_2 \leftarrow g_2^\alpha$. The master public key is $mpk = (g_1, h_1, g_2, u_{1,0}, \ldots, u_{L,0}, u_{1,1}, \ldots, u_{L,1})$. The corresponding master secret key is $msk = h_2$.

Key Derivation. A user's identity is given by a vector $ID = (ID_1, \ldots, ID_\ell)$ where each ID_i is an element in $\mathbb{Z}_p$. To compute the decryption key for identity ID from the master secret key, first one chooses random values $r_i \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ for $i = 1, \ldots, \ell$, then the private key d_{ID} is constructed as

$$(a_0, a_1, \ldots, a_\ell) \;=\; \left(h_2 \prod_{i=1}^{\ell} \left(u_{i,0} \cdot u_{i,1}^{ID_i} \right)^{r_i} \;,\; g_1^{r_1} \;,\; \ldots \;,\; g_1^{r_\ell} \right) \;.$$

Notice that, as required, the secret key for identity $ID = (ID_1, \ldots, ID_\ell)$ can be computed from the secret key $(a_0, a_1, \ldots, a_{\ell-1})$ of the parent $(ID_1, \ldots, ID_{\ell-1})$ by choosing a random $r_\ell \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ and outputting

$$d_{ID} \;=\; \left(a_0 \cdot \left(u_{\ell,0} \cdot u_{\ell,1}^{ID_\ell} \right)^{r_\ell} \;,\; a_1 \;,\; \ldots \;,\; a_{\ell-1} \;,\; g_1^{r_\ell} \right)$$

Encryption. To encrypt a message $\mathfrak{m} \in \mathbb{G}_T$ for an identity $ID = (ID_1, \ldots, ID_\ell)$, the sender first chooses $t \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ and outputs the ciphertext $C = (C_1, C_2, C_3)$, where

$$C_1 \;=\; g_1^t \qquad C_2 \;=\; \left(C_{2,i} = (u_{i,0} \cdot u_{i,1}^{ID_i})^t \right)_{i=1,\ldots,\ell} \qquad C_3 \;=\; \mathfrak{m} \cdot \hat{e}(h_1, g_2)^t$$

Decryption. If the receiver is the root authority holding the master key msk, then he can recover the message by computing $C_3/\hat{e}(C_1, msk)$. Any other receiver with identity $ID = (ID_1, \ldots, ID_\ell)$

and decryption key $d_{ID} = (a_0, a_1, \ldots, a_\ell)$ decrypts a ciphertext $C = (C_1, (C_{2,i})_{i=1,\ldots,\ell}, C_3)$ by computing

$$\mathfrak{m} \leftarrow C_3 \cdot \frac{\prod_{i=1}^{\ell} \hat{e}(a_i, C_i)}{\hat{e}(C_1, a_0)} .$$

The fact that decryption works can be seen as follows.

$$\begin{aligned} \frac{\prod_{i=1}^{\ell} \hat{e}(a_i,\ C_i)}{\hat{e}(C_1,\ a_0)} &= \frac{\prod_{i=1}^{\ell} \hat{e}\big(g_1^{r_i},\ (u_{i,0} \cdot u_{i,1}^{ID_i})^t\big)}{\hat{e}\big(g_1^t,\ h_2 \prod_{i=1}^{\ell} (u_{i,0} \cdot u_{i,1}^{ID_i})^{r_i}\big)} \\ &= \frac{\prod_{i=1}^{\ell} \hat{e}\big(g_1^t,\ (u_{i,0} \cdot u_{i,1}^{ID_i})^{r_i}\big)}{\hat{e}(g_1^t,\ h_2) \cdot \hat{e}\big(g_1^t,\ \prod_{i=1}^{\ell} (u_{i,0} \cdot u_{i,1}^{ID_i})^{r_i}\big)} \\ &= \frac{1}{\hat{e}(g_1, h_2)^t} = \frac{1}{\hat{e}(h_1, g_2)^t} \end{aligned}$$

The main difference between the original HIBE scheme of [BB04] and our variant above is that our scheme uses a different value $u_{i,1}$ for each level, while the original scheme uses the same value u_1 for all levels. Adding wildcard functionality to the original scheme would require us to include u_1^t in the ciphertext, but this ruins security as it can be used to change the recipient identity for non-wildcard levels as well.

Theorem A.1 If the (t, ϵ) BDDH assumption holds in $\mathbb{G}$, then the $\mathcal{BB\text{-}HIBE}$ scheme with hierarchy depth L is (t', q_K, ϵ) IND-sID-CPA-secure, where $t' = t - \Theta(L \cdot q_K \cdot t_{\exp})$ and $t_{\exp}$ is the maximum time for an exponentiation in $\mathbb{G}$.

Proof: The present proof follows very closely the proof of security for the original scheme in [BB04]. As before, we assume that there exist an adversary $\mathcal{A}$ that breaks the IND-sID-CPA-security of the HIBE scheme $\mathcal{BB\text{-}HIBE}$ and then we show how to efficiently build another adversary $\mathcal{B}$ that, using $\mathcal{A}$ as a black box, manages to solve the BDDH problem in $\mathbb{G}$.

Algorithm $\mathcal{B}$ first receives as input a random tuple $(g, A = g^a, B = g^b, C = g^c, Z)$ and its goal is to determine whether $Z = \hat{e}(g, g)^{abc}$ or $\hat{e}(g, g)^z$ for a random element z in $\mathbb{Z}_p$. Algorithm $\mathcal{B}$ should output 1 if $Z = \hat{e}(g, g)^{abc}$ and 0, otherwise. Algorithm $\mathcal{B}$ works as follows.

Initialisation. Algorithm $\mathcal{B}$ starts interacting with $\mathcal{A}$ in the IND-sID-CPA game. Let $ID^* = (ID_1^*, \ldots, ID_{\ell^*}^*)$, where $0 \leq \ell^* \leq L$ be the challenge identity outputted by $\mathcal{A}$. If necessary, $\mathcal{B}$ appends random elements in $\mathbb{Z}_p$ to ID^* so that ID^* is a vector of length L.

Setup. To generate the systems parameters, $\mathcal{B}$ first sets $g_1 \leftarrow g$, $h_1 \leftarrow A$, and $g_2 \leftarrow B$. Algorithm $\mathcal{B}$ then chooses $\alpha_{1,0}, \ldots, \alpha_{L,0}, \alpha_{1,1}, \ldots, \alpha_{L,1} \xleftarrow{\$} \mathbb{Z}_p^*$ at random and sets $u_{i,0} \leftarrow g_1^{\alpha_{i,0}} \cdot h_1^{-ID_i^* \alpha_{i,1}}$ and $u_{i,1} \leftarrow h_1^{\alpha_{i,1}}$ for $i = 1, \ldots, L$. Next, $\mathcal{B}$ gives to $\mathcal{A}$ as the master public key the value $mpk \leftarrow (g_1, h_1, g_2, u_{1,0}, \ldots, u_{L,0}, u_{1,1}, \ldots, u_{L,1})$. Note that the corresponding master secret key $msk = g_2^a$ is unknown to $\mathcal{B}$.

Key Derivation queries. During the phases of its attack against the IND-sID-CPA-security of $\mathcal{BB\text{-}HIBE}$, $\mathcal{A}$ can query up to q_K queries to its key derivation oracle. Let $ID = (ID_1, \ldots, ID_\ell)$, where $ID_i \in \mathbb{Z}_p$ and $\ell \leq L$, be one such query. Thus, ID cannot be a prefix of ID^*. Let j be the smallest index such that $ID_j \neq ID_j^*$. It follows necessarily that $1 \leq j \leq \ell$. To reply to this query, $\mathcal{B}$ first computes the key for identity $ID|_{\leq j} = (ID_1, \ldots, ID_j)$ and then derive the key for ID as in the key derivation algorithm. To derive the key for identity $ID|_{\leq j}$, $\mathcal{B}$ chooses the values $r_1, \ldots, r_j \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ at random and sets

$$\begin{aligned} d_{ID|_{\leq j}} &= (a_0, a_1, \ldots, a_j) \\ &= \left(g_2^{\frac{-\alpha_{j,0}}{\alpha_{j,1}(ID_j - ID_j^*)}} \cdot \prod_{i=1}^{j} \left(u_{i,0} \cdot u_{i,1}^{ID_i}\right)^{r_i} \,,\; g_1^{r_1} \,,\; \ldots \,,\; g_1^{r_{j-1}} \,,\; g_2^{\frac{1}{\alpha_{j,1}(ID_j - ID_j^*)}} g_1^{r_j} \right) . \end{aligned}$$

To see why $(a_0, a_1, \ldots, a_j)$ is a valid random private key for identity $ID|_{\leq j}$, let $\tilde{r}_j = r_j - \frac{b}{\alpha_{j,1}(ID_j - ID_j^*)} \bmod p$. Then, we have that

$$\begin{aligned} g_2^{\frac{-\alpha_{j,0}}{\alpha_{j,1}(ID_j - ID_j^*)}} \cdot (u_{j,0} \cdot u_{j,1}^{ID_j})^{r_j} &= g_2^{\frac{-\alpha_{j,0}}{\alpha_{j,1}(ID_j - ID_j^*)}} \cdot \left(u_{j,0} \cdot u_{j,1}^{ID_j}\right)^{\tilde{r}_j + \frac{b}{\alpha_{j,1}(ID_j - ID_j^*)}} \\ &= g_2^{\frac{-\alpha_{j,0}}{\alpha_{j,1}(ID_j - ID_j^*)}} \cdot \left(g_1^{\alpha_{j,0}} \cdot h_1^{\alpha_{j,1}(ID_j - ID_j^*)}\right)^{\frac{b}{\alpha_{j,1}(ID_j - ID_j^*)}} \cdot \left(u_{j,0} \cdot u_{j,1}^{ID_j}\right)^{\tilde{r}_j} \\ &= h_1^b \cdot \left(u_{j,0} \cdot u_{j,1}^{ID_j}\right)^{\tilde{r}_j} \\ &= g_2^a \cdot \left(u_{j,0} \cdot u_{j,1}^{ID_j}\right)^{\tilde{r}_j} . \end{aligned}$$

From the above, it follows that

$$a_0 = g_2^a (u_{j,0} \cdot u_{j,1}^{ID_j})^{\tilde{r}_j} \prod_{i=1}^{j-1} (u_{i,0} \cdot u_{i,1}^{ID_i})^{r_i}, \quad a_1 = g_1^{r_1}, \ldots, a_{j-1} = g_1^{r_{j-1}}, \quad a_j = g_1^{\tilde{r}_j},$$

where $r_1, \ldots, r_{j-1}, \tilde{r}_j$ are uniformly distributed over $\mathbb{Z}_p$. From $(a_0, a_1, \ldots, a_j)$, algorithm $\mathcal{B}$ can derive the key for ID as in the key derivation algorithm.

Challenge. Let $(\mathfrak{m}_0^*, \mathfrak{m}_1^*)$ be the pair of messages that $\mathcal{A}$ outputs at the end of the first phase of the IND-sID-CPA game. Algorithm $\mathcal{B}$ then chooses a random bit $b \in \{0, 1\}$ and sends $C^* = (C, \mathfrak{m}_b^* \cdot Z, C^{\alpha_{1,0}}, \ldots, C^{\alpha_{\ell^*,0}})$ to $\mathcal{A}$ as the challenge ciphertext. Since $u_{i,0} \cdot u_{i,1}^{ID_i^*} = g_1^{\alpha_{i,0}}$ for all i, we have that

$$C^* = (g_1^c, \mathfrak{m}_b^* \cdot Z, (u_{1,0} \cdot u_{1,1}^{ID_i^*})^c, \ldots, (u_{\ell^*,0} \cdot u_{\ell^*,1}^{ID_{\ell^*}^*})^c).$$

As a result, when $Z = \hat{e}(g, g)^{abc} = \hat{e}(h_1, g_2)^c$, C^* is a valid encryption of message $\mathfrak{m}_b^*$ for the challenge identity $ID^* = (ID_1^*, \ldots, ID_{\ell^*}^*)$. On the other hand, when $Z = \hat{e}(g, g)^z$ for a random value $z \in \mathbb{Z}_p$, then the challenge ciphertext is independent of b from the view point of the adversary.

Guess. Let b' be the output of $\mathcal{A}$ at the end of the second phase of the IND-sID-CPA game. If $b = b'$, then algorithm $\mathcal{B}$ outputs 1, guessing that $Z = \hat{e}(g, g)^{abc}$. Otherwise, $\mathcal{B}$ outputs 0.

Clearly, when $Z = \hat{e}(g, g)^{abc}$, the view of $\mathcal{A}$ is identical to its view in a real attack and, thus, the probability that $b = b'$ is exactly the probability that $\mathcal{A}$ wins the IND-sID-CPA game. On the other hand, when Z is a random group element in $\mathbb{G}_T$, then the probability that $b = b'$ is exactly 1/2. From the above, the result announced in Theorem A.1 follows immediately. ∎

B The $\mathcal{BBG}$-$\mathcal{HIBE}$ Scheme

In this section we present the HIBE scheme due to Boneh, Boyen and Goh [BBG05], referred to as the $\mathcal{BBG}$-$\mathcal{HIBE}$ scheme here. The Setup and KeyDer algorithms are exactly as in the $\mathcal{BBG}$-$\mathcal{WIBE}$ scheme presented in Section 6.2. Encryption and decryption work as follows.

Encryption. To encrypt a message $\mathfrak{m} \in \mathbb{G}_T$ for an identity $ID = (ID_1, \ldots, ID_\ell)$, the sender first chooses $t \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ and outputs the ciphertext $C = (C_1, C_2, C_3) \in \mathbb{G} \times \mathbb{G} \times \mathbb{G}_T$, where

$$\begin{aligned} C_1 &\leftarrow g_1^t, \\ C_2 &\leftarrow \left(u_0 \prod_{i=1}^{\ell} u_i^{ID_i}\right)^t, \\ C_3 &\leftarrow \mathfrak{m} \cdot \hat{e}(h_1, g_2)^t \ . \end{aligned}$$

Decryption. If the receiver is the root authority holding the master key $msk = h_2$, then he can recover the message by computing $C_3/\hat{e}(C_1, h_2)$. Any other receiver with identity $ID = (ID_1, \ldots, ID_\ell)$ and decryption key $d_{ID} = (a_0, a_{\ell+1}, , \ldots, a_{L+1})$ decrypts a ciphertext $C = (C_1, C_2, C_3)$ as follows.

$$\begin{aligned} \mathfrak{m} &\leftarrow C_3 \cdot \frac{\hat{e}(C_2, a_{L+1})}{\hat{e}(C_1, a_0)} \\ &= C_3 \cdot \frac{\hat{e}\big(u_0 \prod_{i=1}^{\ell} u_i^{ID_i}, g_1\big)^{rt}}{\hat{e}(g_1, h_2)^t \cdot \hat{e}\big(g_1, u_0 \prod_{i=1}^{\ell} u_i^{ID_i}\big)^{rt}} \\ &= \mathfrak{m} \ . \end{aligned}$$

The following theorem about the security of the scheme was proved in (the full version of) [BBG05].

Theorem B.1 If the (t, ϵ) decisional L-BDHI assumption holds in $\mathbb{G}$, then the $\mathcal{BBG}$-$\mathcal{HIBE}$ scheme with hierarchy depth L is $(t', q'_{\mathrm{K}}, \epsilon')$ IND-sID-CPA-secure for arbitrary q'_{K} and for all

$$t' \ \leq \ t - O(Lq'_{\mathrm{K}} \cdot t_{\exp}) \quad \text{and} \quad \epsilon' \ \geq \ \epsilon \ ,$$

where $t_{\exp}$ is the time for an exponentiation in $\mathbb{G}$.