Construction of (Hierarchical) Identity-Based Encryption Protocols Using Bilinear Pairing

Sanjit Chatterjee

Applied Statistics Unit

Indian Statistical Institute

Kolkata, India
August, 2006

Construction of (Hierarchical) Identity-Based Encryption Protocols Using Bilinear Pairing

Thesis submitted to the Indian Statistical Institute in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy

by

Sanjit Chatterjee
Applied Statistics Unit
Indian Statistical Institute
203, B.T. Road,
Kolkata 700108.
INDIA
e-mail: sanjit_t@isical.ac.in

under the supervision of

Dr. Palash Sarkar
Applied Statistics Unit
Indian Statistical Institute
203, B.T. Road,
Kolkata 700108.
INDIA
e-mail: palash@isical.ac.in

The scientist does not study nature because it is useful; he studies it because he delights in it, and he delights in it because it is beautiful. If nature were not beautiful, it would not be worth knowing, and if nature were not worth knowing, life would not be worth living.

Henri Poincaré

ACKNOWLEDGMENT

This dissertation marks the culmination of its author's four years of research effort at the Indian Statistical Institute. One has every right to be sceptical to ask whether the temporality of the work will be able to surpass that of its author's stint as a "purely temporary project-linked personnel" at this "Institute of national importance". Be that what it may, there is no doubt that this is the most opportune moment to express the author's gratitude to everyone concerned for withstanding it all.

The author's initial interest in cryptology was aroused by Dr. Palash Sarkar's expository article on the subject published in *Resonance* in September, 2000. It was, therefore, a great privilege to have him as the PhD supervisor. Palash's academic integrity and passion can well be inspiration for any researcher should (s)he aspire for such things at this warped time. It is needless to say that this joint work would not be in the shape it is today but for him. At times it must be too taxing on his nerve to direct this whimsical horse. However, he must be patting at his own back now for being successful in not just directing the horse near the water but also in persuading it to drink from it too! On the author's part, he can only wish that this pairing would continue in future to bring in (un)expected results.

In his pursuit, the author was fortunate enough to find teachers like Prof. K. S. Vijayan, Prof. K. Sikdar and Prof. Rana Barua. The stoic appearance notwithstanding, they turned out to be treasure-troves of abstract algebra and computer science. The author is further indebted to Prof. Barua for allowing him to co-author a paper with him.

There is, perhaps, no researcher in the field of cryptology in India without owing a debt in one way or the other to Prof. Bimal Roy – the GURU of modern cryptology in India. At times he appears like a kalpataru (the wishing tree of Indian mythology) provided one knows how and where to tap, at others his power to find trivial solution with a fillip of fingers (of if not all problems on the earth at least the ones in the vicinity) is quite enlightening to say the least.

There remain the unnamed others, whose names are not explicitly mentioned nevertheless who stood by the author at the time of his crises. For them, his only recourse is to quote Einstein: *Not everything that can be counted counts, and not everything that counts can be counted.*

Contents

Chapter 1

Introduction

Science, it is argued [65], advances through paradigm shifts. Concepts emerge that open-up new vistas of research, fundamentally changing the way we are used to looking at things. Between these paradigm shifts remain the periods of consolidation. Periods when human mind explores the newly found territory, shedding light on hitherto unknown dimensions. If radical changes are the hallmarks of paradigm shifts, the period within witnesses small but continuous developments, occasionally marked by its own milestones. It is in these periods that human faculty tries to grasp the full significance of the new concepts, consolidates its gains and thereby pushes the boundary of our collective knowledge further. The prospects, nevertheless, bring with it new problems too. Perhaps, by the way, making ground for the next paradigm shift.

Cryptology, as a branch of science, is no exception to this common story. Though known from the antiquity and not without some shining milestones; it encountered a paradigm shift exactly three decades ago. Diffie and Hellman [37], in 1976 introduced the notion of public key cryptography (PKC) through their work, appropriately titled, "New Directions in Cryptography". Prior to this work, cryptology was practiced in the "symmetric" setting only, i.e., the same secret key was used for encryption as well as decryption. This kind of symmetric key cryptography necessitates a secret channel to be established between the sender and receiver. This is, no doubt, a cumbersome business when a large number of users want to communicate secretly with each other.

In the public key setting (also known as asymmetric key cryptography), each user possesses a pair of keys, one public key which is published in a publicly available directory and another private key which is known only to the user concerned. When somebody, say Alice, wants to send an encrypted message to Bob, she looks up in the directory for the public key of Bob, encrypts the message using that public key and sends it to Bob. Bob uses his private key to decrypt the message. Naturally, there should be some mathematical relationship between the two keys, so that given the private key it should be (computationally) easy to decrypt the message encrypted using the public key. On the other hand, it should be (computationally) hard to obtain any information regarding the private key given

the public key. In this setting, anybody can send an encrypted message to anybody else, provided the public key is made available, while only the person in possession of the private key can decrypt the message. This way, the problem of key distribution (that we observe in the symmetric setting) can be solved effectively.

Within two years of publication of Diffie-Hellman's work, the field of public key cryptology got a milestone in the form of RSA cryptosystem [79]. Rivest, Shamir and Adleman based their cryptosystem on the hardness of integer factorisation problem. In 1985, ElGamal [41] proposed a public key cryptosystem based on the discrete logarithm problem over cyclic groups. This was soon followed by Koblitz [61] and Miller [73], who independently proposed a public key cryptosystem called the elliptic curve cryptosystem (ECC). ECC is based on the hardness of the discrete log problem over elliptic curve groups. RSA and ECC are the two most popular public key cryptosystems to date.

This kind of PKC when applied, however, bring with them their own problem of key management. In our example, we assumed that Bob has made available his public key in a publicly available directory. Now suppose Eve impersonates Bob and publishes a public key in the name of Bob. Since there is no need of prior communication between Alice and Bob, there is no way for Alice to verify whether the public key available in the name of Bob is *actually* his and not posted by Eve. To solve this kind of problems public keys are related by digital certificates issued by a certifying authority (CA). A certificate is a digital signature of the CA on the public key. Since the keys need periodic refreshing, the CA needs to maintain a revoked list too, along with a valid list of public keys. Moreover, in actual practice there may be a chain of certificates (and CAs) to verify a public key.

This, perhaps, motivated Shamir [84] to introduce the concept of identity-based encryption (IBE). IBE is a kind of public key encryption scheme where the public key of an user can be any arbitrary string – typically the e-mail id. In our example, when Alice wants to send a message to Bob; she encrypts it using the e-mail id of Bob as the public key. There is no need for Alice to go to the CA to verify the public key of Bob. This way an IBE can greatly simplify certificate management. To quote Shamir [84]:

> *It makes the cryptographic aspects of the communication almost transparent to the user, and it can be used effectively even by laymen who know nothing about keys or protocols.*

An IBE consists of four algorithms: Set-up which generates a set of public parameters together with a master key, Key-Gen which generates the private key of an entity, given her identity, Encrypt that encrypts a message given the identity and Decrypt that decrypts the message using the private key. Instead of a certifying authority, here we have a private key generator (PKG) who possesses the master key. In the above example, Bob authenticates himself to the PKG to obtain a private key corresponding to his identity. (This can be even after he receives the encrypted message from Alice.) Bob uses this private key to decrypt all the messages encrypted using his identity as the public key. Note that, Alice need not verify any certificate relating the public key to send an encrypted message to Bob. What she needs is the identity of Bob along with the public parameters of PKG.

Shamir posed a challenge to the crypto community to come out with a practical IBE scheme. A satisfactory solution of this problem eluded cryptographers till the turn of the millennium. The solution, when it finally arrived, came not from one, but three different quarters – Sakai-Ohgishi-Kasahara [80], Boneh-Franklin [20] and Cocks [34]. Among these, the former two based their cryptosystems on bilinear pairing over elliptic curve groups while the last was based on factoring and is less practical. Boneh and Franklin [20] formalised the notion of IBE, gave a precise definition of its security model and related the security of their construction with a natural analogue of the computational Diffie-Hellman problem in the bilinear setting, called bilinear Diffie-Hellman problem. This work caught the immediate attention of the crypto community world wide and turned out to be a milestone within the paradigm of public key cryptography.

Boneh-Franklin use bilinear pairing over elliptic curve groups to construct their IBE. Initially bilinear maps such as Weil and Tate pairing were introduced in cryptology [71, 43] as weapons in the arsenal of the cryptanalyst – to reduce the discrete log problem in the elliptic curve group to the discrete log problem over finite fields. Joux [58] converted it into a tool of cryptography by proposing a one round tri-partite key agreement protocol using bilinear pairing.

These works, in some sense, ignited an explosion in pairing based public key cryptography. Within a period of around five years we see works in different areas of public key cryptography like key agreement [58, 86, 13], identity-based encryption [20, 56, 49, 17, 18, 89, 19, 48], signature (including identity-based signature) [22, 54, 16], threshold decryption [68, 5], access control [87] etcetera based on pairing. Some nice works on efficient implementation of pairing [8, 45, 39, 52] also appeared in this phase. Even a cursory glance at a compendium [6] of these works is enough to give a feel of the research activity generated.

Identity-based encryption using bilinear pairing, in particular, has become an active research area. One reason of this interest is encryption being a core area in cryptology; the other reason is the versatile applications of this primitive. The concept of IBE was soon extended to hierarchical identity-based encryption (HIBE) [56, 49]. Instead of a single component identity in IBE, here identities are considered to be vectors. In HIBE, an entity having identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ and possessing the private key d_{v} can generate the private key of an entity whose identity is $\mathsf{v}' = (\mathsf{v}_1, \ldots, \mathsf{v}_j, \mathsf{v}_{j+1})$. This way HIBE reduces the workload of the PKG.

Both IBE and its extension HIBE can be used to construct other important cryptographic primitives. Naor observed that any IBE can be converted to a digital signature scheme and applying this transformation on the Boneh-Franklin IBE one gets the signature scheme of Boneh-Lynn-Shacham [22]. Similarly, HIBE can be transformed into hierarchical identity-based signature [49]. Other applications of (H)IBE include key delegation [20, 81], forward secure encryption [26], broadcast encryption [38] to name a few.

The security model of IBE, as formalised by Boneh and Franklin, is in a sense an extension of the corresponding concept of public key encryption. The additional complexity arises due to the identities. In the security model, the adversary has access to a key extraction oracle and a decryption oracle. Intuitively, a protocol is said to be secure if the advantage of an

adversary in distinguishing two messages encrypted using the same identity is negligible unless it knows the private key of that identity. The security reduction of initial (H)IBE protocols used the random oracle heuristic. Since random oracles do not exist in reality, it led to a search for protocols secure without random oracle. Initially this has been achieved in a weaker security model called the selective-ID model. As an aside, it should be mentioned that, an IBE secure against chosen plain-text attack (i.e., the adversary is restricted from accessing the decryption oracle) in the selective-ID model can be used to construct a PKE secure against chosen ciphertext attack. This is another interesting application of IBE.

The (H)IBE protocols secure in the selective-ID model do not suffer any degradation in the security reduction, i.e., the adversarial advantage against such a protocol can be directly related to the advantage against the assumed hard problem. In contrast, protocols secure in the full model (with or without random oracle) suffer from a degradation in the security reduction. For HIBE, this degradation is exponential with the number of components in the identity vector. Reducing or controlling this degradation is one of most challenging problems in this area.

1.1 Plan of the Thesis

Within the paradigm of public key encryption, we further explore the problem of efficient and secure construction of (hierarchical) identity-based encryption protocols using bilinear pairing. The thesis is based on the following works [33, 28, 30, 31, 32, 29] and organised as follows.

In Chapter 2, we define the core concepts that are used through-out the work. We give precise formal definition in some of the cases while providing an informal discussion for some other.

In Chapter 3, we review some important (hierarchical) identity-based encryption protocols available in the literature. Our aim in this chapter is to trace the important developments in the field, without being exhaustive. It also helps to present our work in the proper context. Along with the protocols, in some cases we discuss the salient features of the proof techniques while an intuitive justification of the proof is given for some others. These protocols are proved secure against chosen plaintext attack. There exists generic methods (and non-generic too) for achieving chosen ciphertext security (CCA-security). One of the generic techniques is discussed in details. In view of these techniques, in this work we too concentrate on developing protocols secure against chosen plaintext attack.

All the (hierarchical) identity-based encryption protocols discussed in this dissertation use bilinear map over elliptic curve groups as the primary building block. This is also the most computationally intensive part in a protocol. Our first contribution (in Chapter 4) is an efficient algorithm for the computation of modified Tate pairing in general characteristic fields with embedding degree two. We consider the use of Jacobian coordinates for this computation. The idea of encapsulated double-and-line computation and add-and-line computation is introduced. We also describe the encapsulated version of iterated doubling.

Detailed algorithms are presented in each case and the memory requirement is considered. The inherent parallelism in each of the algorithm has been identified leading to optimal two-multiplier algorithms. The cost comparison of our algorithms with previously best known algorithms shows an efficiency improvement of around 33% in the general case and an efficiency improvement of 20% for the case of the curve parameter $a = -3$.

With this background of pairing computation, we proceed to the problem of efficient and secure protocol design. At Eurocrypt 2005, Brent Waters [89] proposed an efficient identity-based encryption scheme secure in the full model. One limitation of this scheme is that the number of elements in the public parameter is rather large. In Chapter 5, we propose a generalisation of Waters scheme. In particular, we show that there is an interesting trade-off between the tightness of the security reduction and smallness of the public parameter. For a given security level, this implies that if one reduces the number of elements in public parameter then there is a corresponding increase in the computational cost due to the increase in group size. This introduces a flexibility in choosing the public parameter size without compromising in security. In concrete terms, to achieve 80-bit security for 160-bit identities we show that compared to Waters protocol the public parameter size can be reduced by almost 90% while increasing the computation cost by 30%. Our construction is proved secure in the full model without random oracle.

In Chapter 6, we extend the IBE protocol of the previous chapter to a hierarchical IBE (HIBE) protocol which is secure in the full model without random oracle. The only previous suggestion for a HIBE in the same setting is due to Waters. Our construction improves upon Waters' suggestion by significantly reducing the number of public parameters. The implication of the security degradation is also discussed briefly.

A major problem with (H)IBE protocols secure in the full model (with or without random oracle) is the degradation in the security reduction. For HIBE, security degrades exponentially with the number of levels. On the other hand, the selective-ID security model introduced in [26, 27] puts severe restriction on the adversary and thereby avoids any degradation in the security reduction. This motivated us to search for a go between of the two extremes.

In Chapter 7, we generalize the selective-ID security model for HIBE by introducing two new security models, $\boldsymbol{\mathcal{M}}_1$ and $\boldsymbol{\mathcal{M}}_2$. Broadly speaking, both these models allow the adversary to commit to a set of identities and in the challenge phase choose any one of the previously committed identities. Two constructions of HIBE are presented which are secure in the two models. Further, we show that the HIBEs can be modified to obtain a multiple receiver IBE which is secure in the selective-ID model without random oracle.

Boneh, Boyen and Goh [19] presented an interesting construction of HIBE (which we call BBG-HIBE) where the ciphertext expansion is independent of the number of levels in the identity tuple. The security of the protocol was proved in the selective-ID model without random oracle.

In Chapter 8, we present two variants of this constant size ciphertext HIBE. Both the constructions have constant size ciphertext expansion. The first variant (which we call ccHIBE) is proved to be secure in the generalized selective-ID model $\boldsymbol{\mathcal{M}}_2$ introduced in the

previous chapter. The second variant (called FullccHIBE) is secure in the full model. Like BBG-HIBE, both are proved secure without random oracle.

In Chapter 9, we augment the selective-ID security model for HIBE by allowing the adversary some flexibility in choosing the target identity tuple during the challenge phase of the security reduction. We denote this model by selective$^+$-ID model (s$^+$ID model). We show that the HIBE proposed by Boneh-Boyen [17] satisfies this notion of security. The case of the constant size ciphertext HIBE of Boneh, Boyen and Goh is different – the original reduction in [19] can be modified to achieve this goal where security degrades by a multiplicative factor of the number of levels in the HIBE. Further we modify the BBG-HIBE to construct a new protocol called $\mathcal{G}_1$, secure in s$^+$ID model without any degradation. $\mathcal{G}_1$ maintains all the attractive features of BBG-HIBE. We build on this new construction another constant size ciphertext HIBE $\mathcal{G}_2$. The security of $\mathcal{G}_2$ is proved under the augmented version of $\boldsymbol{\mathcal{M}}_2$. Our third construction in this chapter is a "product" HIBE $\mathcal{G}_3$ that allows a controllable trade-off between the ciphertext size and the private key size.

We conclude the dissertation by summing-up our contributions in Chapter 10. We also mention some open problems in the area.

Chapter 2

Preliminaries

In this chapter, we define the basic notions used throughout the dissertation. The identity-based encryption protocols that we describe all use *cryptographic* bilinear map as the primary building block. Hence, we start with a definition of bilinear map.

2.1 Bilinear Map

Let G_1 and G_2 be two cyclic groups of same prime order p for some large p where we write G_1 additively and G_2 multiplicatively. Let P be a generator of G_1, i.e., $G_1 = \langle P \rangle$. A mapping $e : G_1 \times G_1 \rightarrow G_2$ is called an admissible bilinear map if it satisfies the following properties:

- Bilinearity: $e(aQ, bR) = e(Q, R)^{ab}$ for all $Q, R \in G_1$ and $a, b \in \mathbb{Z}_p$.
- Non-degeneracy: If $G_1 = \langle P \rangle$, then $G_2 = \langle e(P, P) \rangle$.
- Computability: There exist efficient algorithms to compute the group operations in G_1, G_2 as well as the map $e()$.

Let, $Q = xP$ and $R = yP$; then $e(Q, R) = e(P, P)^{xy} = e(yP, xP) = e(Q, R)$. So the map $e()$ also satisfies the symmetry property.

Known examples of $e()$ usually have G_1 to be a group of Elliptic Curve (EC) or Hyper Elliptic Curve points and G_2 to be a subgroup of a multiplicative group of a finite field. Modified Weil pairing [20], Tate pairing [8, 45], Eta pairing [7], Ate pairing [55] are examples of bilinear maps.

In mathematical literature, it is usual to write elliptic curve group operation additively whereas it is usual to write the finite field group operation multiplicatively. Consequently, in works on pairing computation and implementation of pairing based protocols [8, 45, 11], it is customary to write G_1 additively and G_2 multiplicatively. Initial works on protocol design such as [20, 49] also adhered to this notation. However, later works on protocol design [17, 18, 89] write both G_1 and G_2 multiplicatively. Here we follow the first convention

as it is closer to the known examples of bilinear maps and also because our work includes (Chapter 4) an algorithm to compute the modified Tate pairing. To maintain the consistency of notation through out the work, we rewrite some of the protocols reviewed in Chapter 3 in the additive-multiplicative notation.

Note that, the bilinear pairing can be more generally defined in the *asymmetric* setting [22, 24]. Let $\hat{e} : G_1 \times \hat{G_1} \rightarrow G_2$ be one such bilinear map, where G_1 and $\hat{G_1}$ are two (possibly) distinct groups. The properties of bilinearity, non-degeneracy and computability hold for $\hat{e}()$, but not the symmetry property. In this work we use symmetric bilinear map. Hence, asymmetric bilinear map is not discussed any further.

2.2 Hardness Assumption

Security of identity-based encryption protocols described in this dissertation rely on the hardness of the bilinear Diffie-Hellman (BDH) problem formally introduced by Boneh and Franklin in [20] or one of its variants. In this section we define the problems and the corresponding security assumptions relevant to our work.

2.2.1 Bilinear Diffie-Hellman Assumption

Let $\mathcal{G}$ be a randomized algorithm which takes input a security parameter $\kappa \in \mathbb{Z}^+$, runs in time polynomial in κ and outputs a prime number p, descriptions of two groups G_1, G_2 of order p and an admissible bilinear map $e : G_1 \times G_1 \rightarrow G_2$ as defined in Section 2.1. Here $\log_2(p) = \Theta(\kappa)$, so κ is used to determine the cardinality of the groups G_1, G_2 (i.e., p). Let $\mathcal{G}(1^\kappa) = \langle p, G_1, G_2, e\rangle$ denote the output of $\mathcal{G}$ on input κ. We assume that the descriptions include polynomial time (in κ) algorithms to compute the group operations in G_1, G_2 as well as a polynomial time algorithm for evaluating $e()$. Let $\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$ and $\mathbb{Z}_p^* = \mathbb{Z}_p \setminus \{0\}$.

Bilinear Diffie-Hellman Problem: Let $G_1, G_2, e()$ be as defined in Section 2.1.The bilinear Diffie-Hellman problem (BDH) in $\langle G_1, G_2, e()\rangle$ is the following.

> Given $P, aP, bP, cP \in G_1$ where P is a generator of G_1 and a, b, c are random elements of $\mathbb{Z}_p^*$; compute $e(P, P)^{abc}$.

BDH Assumption: An algorithm $\mathcal{A}$ is said to have an advantage $\epsilon(\kappa)$ in solving the BDH problem in $\langle G_1, G_2, e()\rangle$, generated by $\mathcal{G}(1^\kappa)$ if

$$\mathsf{Adv}_{\mathcal{G},\mathcal{A}}(\kappa) \;=\; \Pr\left[\mathcal{A}(p, G_1, G_2, e, P, aP, bP, cP) = e(P,P)^{abc} \;\middle|\; \begin{array}{c} \langle p, G_1, G_2, e()\rangle \leftarrow \mathcal{G}(1^\kappa), \\ P \xleftarrow{\$} G_1, a, b, c \xleftarrow{\$} \mathbb{Z}_p^* \end{array}\right] \geq \epsilon(\kappa)$$

The $(t(\kappa), \epsilon(\kappa))$-bilinear Diffie-Hellman assumption holds in $\langle G_1, G_2, e()\rangle$ generated by $\mathcal{G}$ if no $t(\kappa)$-time algorithm $\mathcal{A}$, where $t(\kappa)$ is a polynomial in κ, has at least a non-negligible

advantage $\epsilon(\kappa)$ in solving the BDH problem. (As usual, a function $f : \mathbb{R} \rightarrow \mathbb{R}$ is negligible if for any $d > 0$ we have $f(\kappa) < 1/\kappa^d$ for sufficiently large κ.) To simplify the notation, we omit κ and write this as (t, ϵ)-BDH assumption or sometimes just the BDH assumption.

The BDH problem is said to be (t, ϵ)-hard in $\langle G_1, G_2, e() \rangle$ if the (t, ϵ)-BDH assumption holds in $\langle G_1, G_2, e() \rangle$.

The BDH problem has a corresponding decision version which we call the *decisional* bilinear Diffie-Hellman problem (DBDH) [58, 17].

Decisional BDH Problem: Let G_1, G_2 and $e()$ be as defined in Section 2.1. The DBDH problem in $\langle p, G_1, G_2, e() \rangle \leftarrow \mathcal{G}(1^\kappa)$ is as follows:

> Given a tuple $\langle P, aP, bP, cP, Z \rangle \in (G_1)^4 \times G_2$, where P is a generator of G_1 and a, b, c are random elements of $\mathbb{Z}_p$, decide whether $Z = e(P, P)^{abc}$ (which we denote as Z is real) or Z is random.

A probabilistic algorithm $\mathcal{B}$, which takes as input a tuple $\langle P, aP, bP, cP, Z \rangle \in (G_1)^4 \times G_2$, runs in time $t(\kappa)$ (i.e., polynomial in κ) and outputs a bit, has an advantage $\epsilon(\kappa)$ in solving the DBDH problem in $\langle G_1, G_2, e() \rangle$ if

$$\begin{aligned} \mathsf{Adv}^{\mathsf{DBDH}}_{\mathcal{G},\mathcal{B}}(\kappa) \;=\; & |\Pr[\mathcal{B}(P, aP, bP, cP, Z) = 1 | \text{Z is real}] \\ & - \Pr[\mathcal{B}(P, aP, bP, cP, Z) = 1 | \text{ Z is random}]| \geq \epsilon(\kappa) \end{aligned}$$

where the probability is calculated over the random choices of $a, b, c \in \mathbb{Z}_p$ as well as the random bits used by $\mathcal{B}$.

DBDH Assumption: The (t, ϵ)-DBDH assumption holds in $\langle G_1, G_2, e() \rangle$ if no t-time algorithm has advantage at least ϵ in solving the DBDH problem in $\langle G_1, G_2, e() \rangle$. We say that the DBDH problem is (t, ϵ)-hard in $\langle G_1, G_2, e() \rangle$ if the (t, ϵ)-DBDH assumption holds in $\langle G_1, G_2, e() \rangle$.

2.2.2 Variants of BDH

Some variants of bilinear Diffie-Hellman problem have been suggested in the literature. Here we give an informal description of some of these problems, along with their decision version.

Bilinear Diffie-Hellman Exponent Assumption

The Bilinear Diffie-Hellman Exponent problem was introduced by Boneh-Boyen-Goh in [19]. The h-BDHE problem over $\langle G_1, G_2, e() \rangle$ is as follows:

> Given a generator P of G_1, $Q \in G_1$ and $a^i P \in G_1$ for $i = 1, 2, \ldots, h-1, h+1, \ldots, 2h$, compute $e(P, Q)^{a^h}$.

Let $P_i = a^i P$. An algorithm $\mathcal{A}$ has advantage ϵ in solving h-BDHE if

$$\Pr\left[\mathcal{A}(P, Q, P_1, \ldots, P_{h-1}, P_{h+1}, \ldots, P_{2h}) = e(P, Q)^{a^h}\right] \geq \epsilon$$

where the probability is calculated over the random choices of $P, Q \in G_1$, $a \in \mathbb{Z}_p$ as well as the random bits used by $\mathcal{A}$. The h-BDHE assumption holds if there is no such algorithm $\mathcal{A}$ with a non-negligible advantage.

The h-BDHE problem has a corresponding decision version – the *decisional* bilinear Diffie-Hellman exponent problem (h-DBDHE).

> An instance of the h-DBDHE problem over $G_1 = \langle P \rangle$ and G_2 consists of the tuple $(P, Q, aP, a^2P, \ldots, a^{h-1}P, a^{h+1}P, \ldots, a^{2h}P, Z)$ for some $a \in \mathbb{Z}_p$ and the task is to decide whether $Z = e(P, Q)^{a^h}$ or Z is random.

The advantage of a probabilistic algorithm $\mathcal{B}$ that outputs a bit in solving this decision problem is defined as

$$\mathsf{Adv}_{\mathcal{B}}^{\mathsf{DBDHE}} = \left|\Pr[\mathcal{B}(P, Q, \overrightarrow{Y}, e(P, Q)^{a^h}) = 1] - \Pr[\mathcal{B}(P, Q, \overrightarrow{Y}, Z) = 1]\right|$$

where $\overrightarrow{Y} = (aP, a^2P, \ldots a^{h-1}P, a^{h+1}P, \ldots, a^{2h}P)$, and Z is a random element of G_2. The probability is calculated over the random choices of $a \in \mathbb{Z}_p$ and $Z \in G_2$ and also the random bits used by B. The quantity $\mathsf{Adv}^{\mathsf{DBDHE}}(t)$ denotes the maximum of $\mathsf{Adv}_{\mathcal{B}}^{\mathsf{DBDHE}}$ where the maximum is taken over all adversaries, $\mathcal{B}$ running in time at most t.

Bilinear Diffie-Hellman Inversion Assumption

This problem was introduced by Boneh-Boyen [17] and also by Mitsunari, Sakai and Kasahara [76].

> An instance of the bilinear Diffie-Hellman inversion (BDHI) problem consists of the tuple $P, P_1, \ldots, P_h$ where $P_i = a^i P$ for some random $a \in \mathbb{Z}_p$. Here the task is to compute $e(P, P)^{1/a}$.

Informally speaking, the h-BDHI assumption holds if there is no efficient algorithm that can solve the h-BDHI problem with non-negligible advantage.

A slightly weaker but related problem which is called the weak BDHI* problem over $\langle G_1, G_2, e() \rangle$ is as follows:

> Suppose we are given a random generator P of G_1, another random element $Q \in G_1$ and $aP, \ldots, a^h P$ for a random $a \in \mathbb{Z}_p$. The task is to compute $e(P, Q)^{a^{h+1}}$.

Decisional Weak Bilinear Diffie-Hellman Inversion Problem This is the decision version of the weak BDHI* problem. The decisional weak bilinear Diffie-Hellman inversion problem (h-wDBDHI*) was used by Boneh-Boyen-Goh in [19].

> An instance of the h-wDBDHI* problem over $G_1 = \langle P \rangle$ and G_2 consists of the tuple $(P, Q, aP, a^2P, \ldots, a^hP, Z)$ for some $a \in \mathbb{Z}_p$ and the task is to decide whether $Z = e(P,Q)^{a^{h+1}}$ or Z is random.

The advantage of a probabilistic algorithm $\mathcal{B}$ that outputs a bit in solving this decision problem is defined as

$$\mathsf{Adv}_{\mathcal{B}}^{h\text{-wDBDHI}^*} = \left| \Pr[\mathcal{B}(P, Q, \overrightarrow{Y}, e(P,Q)^{a^{h+1}}) = 1] - \Pr[\mathcal{B}(P, Q, \overrightarrow{Y}, Z) = 1] \right|$$

where $\overrightarrow{Y} = (aP, a^2P, \ldots a^hP)$, and Z is a random element of G_2. The probability is calculated over the random choices of $a \in \mathbb{Z}_p$ and $Z \in G_2$ and also the random bits used by $\mathcal{B}$. The quantity $\mathsf{Adv}^{h\text{-wDBDHI}^*}(t)$ denotes the maximum of $\mathsf{Adv}_{\mathcal{B}}^{h\text{-wDBDHI}^*}$ where the maximum is taken over all adversaries, $\mathcal{B}$ running in time at most t.

The (t, ϵ, h)-wDBDHI* assumption holds in $\langle G_1, G_2, e() \rangle$ if no t-time algorithm has advantage at least ϵ in solving the h-wDBDHI* in $\langle G_1, G_2, e() \rangle$. When the context is clear, we drop t and ϵ and refer to this as h-wDBDHI* assumption.

2.3 Public Key Encryption

The concept of public key cryptography was introduced by Diffie and Hellman in 1976 [37]. A public key encryption scheme can be specified by three probabilistic algorithms described below. A more formal description can be found in [50].

Key-Gen: This algorithm takes input a security parameter $\kappa \in \mathbb{Z}^+$ and generates a public and private key pair (pk, sk) from the appropriate key space. The public key is made available in a public directory while the private key is kept secret.

Encrypt: It takes input a message M from the appropriate message space and the public key pk of the recipient and outputs a ciphertext C. Let $E()$ be the corresponding encryption function; then $C = E(pk, M)$.

Decrypt: It takes input a ciphertext C from the appropriate cipher space and the private key sk of the recipient of C. Suppose $D()$ is the corresponding decryption function. The algorithm outputs $M = D(sk, C)$ if $C = E(pk, M)$. Otherwise, it outputs "reject".

Indistinguishability against adaptive chosen ciphertext attack [50, 14] is the strongest accepted notion of security for a public key encryption scheme. An encryption scheme secure against such an attack is said to be IND-CCA2 secure. We give an informal description of IND-CCA2 security in terms of the following game between a challenger and an adversary $\mathcal{A}$.

Given the security parameter κ, the challenger runs the Key-Generation algorithm to generate a public and private key pair (pk, sk). It gives $\mathcal{A}$ the public key pk. Given the public key, $\mathcal{A}$ adaptively issues decryption queries, which the challenger must properly answer. At some point, $\mathcal{A}$ outputs two equal length messages M_0, M_1 and the challenger responds with an encryption C^* of M_γ, where γ is a random bit. The adversary continues with adaptive decryption queries but not on C^*. Finally, $\mathcal{A}$ outputs its guess γ' of γ and wins if $\gamma' = \gamma$.

The advantage of $\mathcal{A}$ against the encryption scheme is $\mathsf{Adv}_{\mathcal{A}} = |\Pr[\gamma = \gamma'] - 1/2|$. An encryption scheme is said to be (t, q, ϵ)-IND-CCA2 secure, if no t time randomized algorithm $\mathcal{A}$ that makes at most q decryption queries, has advantage at least ϵ in the above game.

A weaker notion of security, called semantic security or security against chosen plaintext attack (in short IND-CPA security) of a public key encryption scheme is available in the literature [50, 14]. In the IND-CPA security game the adversary is not allowed to place any decryption query. Given with a random public key, here the adversary outputs two equal length messages M_0, M_1 and the challenger responds with an encryption C^* of M_γ. The adversary wins if it can predict γ.

2.4 Identity-Based Encryption Protocols

Following [84, 20] an identity-based encryption scheme $\mathcal{E}$ is specified by four probabilistic polynomial time algorithms: Setup, Key-Gen, Encrypt and Decrypt.

Setup: This randomized algorithm takes input a security parameter 1^κ, and returns the system parameters PP together with the master key msk. The system parameters include a description of the message space $\mathcal{M}$, the ciphertext space $\mathcal{C}$ and the identity space $\mathcal{I}$. They are publicly known while the master key is known only to the private key generator (PKG).

Key-Gen: This randomized algorithm takes as input an identity $\mathsf{v} \in \mathcal{I}$ together with the system parameters PP and the master key msk and returns a private key d_{v}, using the master key. The identity v is used as the public key while d_{v} is the corresponding private key.

Encrypt: This randomized algorithm takes as input an identity $\mathsf{v} \in \mathcal{I}$ and a message $M \in \mathcal{M}$ and produces a ciphertext $C \in \mathcal{C}$ using the system parameters PP.

Decrypt: This is a deterministic algorithm which takes as input a ciphertext $C \in \mathcal{C}$, the private key d_{v} of the corresponding identity v and the system parameters PP. It returns the message M or bad if the ciphertext is not valid.

These set of algorithms must satisfy the standard consistency requirement:
For (PP,msk) output by Setup, let d_{v} be a private key generated by Key-Gen given input the identity v, then

$$\forall M \in \mathcal{M} : \text{Decrypt}(\text{Encrypt}(M, \mathsf{v}, \mathsf{PP}), d_{\mathsf{v}}, \mathsf{PP}) = M$$

2.4.1 Hierarchical Identity-Based Encryption

Hierarchical identity-based encryption (HIBE) is an extension of IBE. In contrast to IBE, here identities are represented as vectors. So for a HIBE of maximum height h (henceforth denoted as h-HIBE) any identity v is a tuple $(\mathsf{v}_1, \ldots, \mathsf{v}_\tau)$ where $1 \leq \tau \leq h$. Let, $\mathsf{v}' = \mathsf{v}'_1, \ldots, \mathsf{v}'_j$, $j < \tau$ be another identity tuple. We say v' is a prefix of v if $\mathsf{v}'_i = \mathsf{v}_i$ for all $1 \leq i \leq j$.

Like IBE, here also the PKG has a set of public parameters PP and a master key msk. For all identities at the first level the private key is generated by the PKG using msk. For identities at the second level on-wards, the private key can be generated by the PKG or any of its ancestors. In the above example, the private key d_v of v can be generated by an entity whose identity is a prefix of v and who has obtained the corresponding private key.

The generation of private key can be a computationally intensive task. The identity of an entity must be authenticated before issuing a private key and the private key needs to be transmitted securely to the concerned entity. HIBE reduces the workload of the PKG by delegating the task of private key generation and hence authentication of identity and secure transmission of private key to its lower levels. However, only the PKG has a set of public parameters. The identities at different levels do not have any public parameter associated with them.

Apart from being a standalone cryptographic primitive, HIBE has many interesting applications such as forward secure encryption [26] and broadcast encryption [38].

Following [56, 49] an h-HIBE scheme $\mathcal{H}$ is specified by four PPT algorithms: Setup, Key-Gen, Encrypt and Decrypt.

Setup: This randomized algorithm takes input a security parameter 1^κ and returns the system parameters PP together with the master key msk. The system parameters include a description of the finite message space $\mathcal{M}$, the finite ciphertext space $\mathcal{C}$ and the finite identity space $\mathcal{I}$. These are publicly known while the master key is known only to the private key generator (PKG).

Key-Gen: This randomized algorithm takes as input an identity tuple $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_\tau)$ and the private key $d_{\mathsf{v}|\tau-1}$ for the identity $(\mathsf{v}_1, \ldots, \mathsf{v}_{\tau-1})$ and returns a private key d_v using $d_{\mathsf{v}|\tau-1}$, or directly using the master key. The identity v is used as the public key while d_v is the corresponding private key.

Encrypt: This randomized algorithm takes as input the identity v and a message M from the message space and produces a ciphertext C in the ciphertext space.

Decrypt: This is a deterministic algorithm which takes as input the ciphertext C and a private key d_v of the corresponding identity v and returns the message or bad if the ciphertext is not valid.

The standard consistency requirement that we mention for IBE in Section 2.4 is also applicable for HIBE. If d_v is a private key corresponding to the identity tuple v generated by

the Key-Gen algorithm and C is the output of the Encrypt algorithm for a message $M \in \mathcal{M}$ using v as a public key and PP; then the Decrypt algorithm must return M on input d_{v} and C.

2.5 Security Model of (H)IBE

As we have already noted, HIBE is a generalisation of IBE i.e., an IBE can be thought of as a single level HIBE. So instead of describing the security models of IBE and HIBE separately, we only describe the security model of HIBE.

2.5.1 Security Against Chosen Ciphertext Attack

In case of public key encryption, we have seen that security against adaptive chosen ciphertext attack is the standard notion of security. Boneh and Franklin extended this notion of security to the identity-based setting [20]. They termed this as $\mathsf{IND\text{-}ID\text{-}CCA}$ security.

Let $\mathcal{H}$ be an h-HIBE scheme as defined in the previous section. The $\mathsf{IND\text{-}ID\text{-}CCA}$ security for $\mathcal{H}$ is defined [56, 49, 17] in terms of the following game between a challenger and an adversary of the HIBE. The adversary is allowed to place two types of oracle queries – decryption queries to a decryption oracle $\mathcal{O}_d$ and key-extraction queries to a key-extraction oracle $\mathcal{O}_k$.

Setup The challenger takes input a security parameter 1^κ and runs the Setup algorithm of the HIBE. It provides $\mathcal{A}$ with the system parameters PP while keeping the master key msk to itself.

Phase 1: Adversary $\mathcal{A}$ makes a finite number of queries where each query is one of the two types:

- key-extraction query $\langle \mathsf{v} \rangle$: This query is placed to the key-extraction oracle $\mathcal{O}_k$. $\mathcal{O}_k$ generates a private key d_{v} of v and returns it to $\mathcal{A}$.
- decryption query $\langle \mathsf{v}, C \rangle$: This query is placed to the decryption oracle $\mathcal{O}_d$. It returns the resulting plaintext to $\mathcal{A}$.

$\mathcal{A}$ is allowed to make these queries adaptively, i.e., any query may depend on the previous queries as well as their answers.

Challenge: When $\mathcal{A}$ decides that Phase 1 is complete, it fixes an identity v^* and two equal length messages M_0, M_1 under the (obvious) constraint that it has not asked for the private key of v^* or any prefix of v^*. The challenger chooses uniformly at random a bit $\gamma \in \{0, 1\}$ and obtains a ciphertext C^* corresponding to M_γ, i.e., C^* is output of the Encrypt algorithm on input $(M_\gamma, \mathsf{v}^*, \mathsf{PP})$. It returns C^* as the challenge ciphertext to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ now issues additional queries just like Phase 1, with the (obvious) restriction that it cannot place a decryption query for the decryption of C^* under v^* or any of its prefixes nor a key-extraction query for the private key of v^* or any prefix of v^*. All other queries are valid and $\mathcal{A}$ can issue these queries adaptively just like Phase 1. The challenger responds as in Phase 1.

Guess: $\mathcal{A}$ outputs a guess γ' of γ.
The advantage of the adversary $\mathcal{A}$ in attacking the HIBE scheme $\mathcal{H}$ is defined as:

$$\mathsf{Adv}_{\mathcal{A}}^{\mathcal{H}} = |\Pr[(\gamma = \gamma')] - 1/2| \,.$$

An h-HIBE scheme $\mathcal{H}$ is said to be $(t, q_{\mathsf{ID}}, q_{\mathsf{C}}, \epsilon)$-secure against adaptive chosen ciphertext attack ($(t, q_{\mathsf{ID}}, q_{\mathsf{C}}, \epsilon)$-IND-ID-CCA secure) if for any t-time adversary $\mathcal{A}$ that makes at most q_{ID} private key queries and at most q_{C} decryption queries, $\mathsf{Adv}_{\mathcal{A}}^{\mathcal{H}} \leq \epsilon$. In short, we say $\mathcal{H}$ is IND-ID-CCA secure or when the context is clear, simply CCA-secure.

2.5.2 Security Against Chosen Plaintext Attack

Security reduction of (H)IBE protocols available in the literature [49, 17, 19, 89] generally concentrate on proving security in a weaker model. This is called security against chosen plaintext attack. Boneh and Franklin [20] defines this notion as IND-ID-CPA security. The corresponding game is similar to the game defined above, except that the adversary is *not* allowed access to the decryption oracle $\mathcal{O}_d$. The adversary is allowed to place adaptive private key extraction queries to the key-extraction oracle $\mathcal{O}_k$ and everything else remains the same. For the sake of completeness, we give a description of the IND-ID-CPA game for an h-HIBE $\mathcal{H}$ below.

Setup The challenger takes input a security parameter 1^κ and runs the Setup algorithm of the HIBE. It provides $\mathcal{A}$ with the system parameters PP while keeping the master key msk to itself.

Phase 1: Adversary $\mathcal{A}$ makes a finite number of key-extraction query to $\mathcal{O}_k$. For a private key query corresponding to an identity v, the key-extraction oracle generates the private key d_{v} of v and returns it to $\mathcal{A}$. $\mathcal{A}$ is allowed to make these queries adaptively, i.e., any query may depend on the previous queries as well as their answers.

Challenge: At this stage $\mathcal{A}$ fixes an identity, v^* and two equal length messages M_0, M_1 under the (obvious) constraint that it has not asked for the private key of v^* or any of its prefixes. The challenger chooses uniformly at random a bit $\gamma \in \{0, 1\}$ and obtains a ciphertext (C^*) corresponding to M_γ, i.e., C^* is the output of the Encryption algorithm on input $(M_\gamma, \mathsf{v}^*, \mathsf{PP})$. It returns C^* as the challenge ciphertext to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ now issues additional queries just like Phase 1, with the (obvious) restriction that it cannot place a key-extraction query for the private key of v^* or any prefix of v^*. All other queries are valid and $\mathcal{A}$ can issue these queries adaptively just like Phase 1.

Guess: $\mathcal{A}$ outputs a guess γ' of γ.
Like the IND-ID-CCA game, the advantage of the adversary $\mathcal{A}$ in attacking the HIBE scheme $\mathcal{H}$ is defined as:

$$\mathsf{Adv}_{\mathcal{A}}^{\mathcal{H}} = |\Pr[(\gamma = \gamma')] - 1/2| \,.$$

An h-HIBE scheme $\mathcal{H}$ is said to be (t, q, ϵ) secure against adaptive chosen plaintext attack if for any t-time adversary $\mathcal{A}$ that makes at most q private key extraction queries, $\mathsf{Adv}_{\mathcal{A}}^{\mathcal{H}} < \epsilon$. In short we say $\mathcal{H}$ is (t, q, ϵ)-IND-ID-CPA secure or simply CPA-secure if the context is clear.

There are generic techniques [27, 21] to convert an $(h+1)$-HIBE $\mathcal{H}'$ secure in the sense IND-ID-CPA to an h-HIBE $\mathcal{H}$ which is secure in the sense IND-ID-CCA. One of these techniques is described in Section 3.5. In view of such generic techniques to achieve CCA security, in this dissertation we concentrate on achieving CPA security only.

2.5.3 Selective-ID Model

Canetti, Halevi and Katz [26, 27] gave a new and weaker definition of security for identity based encryption schemes – the so called *selective*-ID model. In this model, the adversary $\mathcal{A}$ commits to a target identity before the system is set up. This notion of security is called the selective identity, chosen ciphertext security (IND-sID-CCA security in short). Following [26, 27, 17] we define IND-sID-CCA security for an h-HIBE in terms of the game described below.

Initialization: The adversary outputs a target identity tuple $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_u^*)$, $1 \leq u \leq h$ on which it wishes to be challenged.

Setup: The challenger sets up the HIBE and provides the adversary with the system public parameters PP. It keeps the master key msk to itself.

Phase 1: Adversary $\mathcal{A}$ makes a finite number of queries where each query is either a decryption or a key-extraction query. In a decryption query, it provides the ciphertext as well as the identity under which it wants the decryption. Similarly, in a key-extraction query, it asks for the private key of the identity it provides. Further, $\mathcal{A}$ is allowed to make these queries adaptively, i.e., any query may depend on the previous queries as well as their answers. The only restriction is that it cannot ask for the private key of v^* or any of its prefixes.

Challenge: At this stage, $\mathcal{A}$ outputs two equal length messages M_0, M_1 and gets a ciphertext C^* corresponding to M_γ encrypted under the public key v^*, where γ is chosen by the challenger uniformly at random from $\{0, 1\}$.

Phase 2: $\mathcal{A}$ now issues additional queries just like Phase 1, with the (obvious) restriction that it cannot ask for the decryption of C^* under v^* or any of its prefixes nor the private key of v^* or any prefix of v^*.

Guess: $\mathcal{A}$ outputs a guess γ' of γ.
The advantage of the adversary $\mathcal{A}$ in attacking the HIBE scheme $\mathcal{H}$ is defined as:

$$\mathsf{Adv}_{\mathcal{A}}^{\mathcal{H}} = |\Pr[(\gamma = \gamma')] - 1/2| \,.$$

The HIBE scheme $\mathcal{H}$ is said to be $(t, q_{\mathsf{ID}}, q_{\mathsf{C}}, \epsilon)$-secure against selective identity, adaptive chosen ciphertext attack (in short, $(t, q_{\mathsf{ID}}, q_{\mathsf{C}}, \epsilon)$-IND-sID-CCA secure) if for any t-time adversary $\mathcal{A}$ that makes at most q_{ID} private key queries and at most q_{C}) decryption queries, $\mathsf{Adv}_{\mathcal{A}}^{\mathcal{H}} < \epsilon$.

We may restrict the adversary from making any decryption query. An h-HIBE scheme $\mathcal{H}$ is said to be (t, q, ϵ)-secure against selective identity, adaptive chosen plaintext attack (in short, $(t, q_{\mathsf{ID}}, \epsilon)$-IND-sID-CPA secure) if for any t-time adversary $\mathcal{A}$ that makes at most q_{ID} private key queries, $\mathsf{Adv}_{\mathcal{A}}^{\mathcal{H}} < \epsilon$.

Henceforth, we will call this restricted model the selective-ID (sID) model and the unrestricted model to be the full model.

The generic technique [27, 21] for converting a CPA-secure HIBE to a CCA-secure HIBE is also applicable to the sID model. Hence, as in the full model it is more convenient in the sID model also to initially construct a CPA-secure (H)IBE and then convert it into a CCA-secure one.

Chapter 3

Previous Works in Identity-Based Encryption

In this chapter, we review some of the (hierarchical) identity-based encryption schemes, based on bilinear pairing, available in the literature. This being a relatively new area, there are quite a large number of works studying the different aspects of identity-based encryption, including security notions and applications of this primitive to other areas in cryptography. This chapter in no way tries to be encyclopedic, rather we concentrate on those works that help to place the concepts presented in this work in a proper context.

3.1 Identity-Based Encryption

It has already been observed that Boneh and Franklin [20] were first to propose a practical identity-based encryption scheme using bilinear pairing with a proper security model and a proof. Sakai, Ohgishi and Kashahara had also independently proposed an IBE [80] using pairing. Their work being in Japanese was not much noticed at the time it appeared. Nevertheless, it is due to the work of Boneh and Franklin that IBE caught immediate attention of the crypto community.

Boneh and Franklin proposed two protocols – BasicIdent and FullIdent in their terminology. The first protocol is secure in the sense IND-ID-CPA while the second is secure in the sense IND-ID-CCA. Both the protocols use cryptographic hash functions that are modelled as random oracles in the security reduction. While describing the Boneh-Franklin IBE, as well as the later protocols, we concentrate on protocols achieving security against chosen plain text attack (i.e., CPA secure). There are mainly two reasons to do so. Firstly, these CPA secure protocols and their security reduction capture the essential ideas that we are interested in. Secondly, once we get a CPA-secure IBE, there are standard techniques to achieve CCA security.

For all the protocols that we describe in this chapter as well as later in the dissertation,

it is assumed that, given a security parameter κ, the descriptions of G_1, G_2 and the bilinear map $e() : G_1 \times G_1 \rightarrow G_2$ are publicly available. The descriptions include polynomial time (in κ) algorithms to compute the group operations in G_1, G_2 and $e()$.

We now describe BasicIdent followed by the salient features of the security reduction.

Setup: Let P be a generator of G_1. Pick a random $s \in \mathbb{Z}_p^*$ and set $P_{\mathsf{pub}} = sP$. Choose cryptographic hash functions $H_1 : \{0,1\}^* \rightarrow G_1^*$, $H_2 : G_2 \rightarrow \{0,1\}^n$. The master secret is s and the public parameters are $\mathsf{PP} = \langle P, P_{\mathsf{pub}}, H_1, H_2 \rangle$.

Key-Gen: Given an identity $\mathsf{v} \in \{0,1\}^*$, compute $Q_{\mathsf{v}} = H_1(\mathsf{v})$ and set the private key to $d_{\mathsf{v}} = sQ_{\mathsf{v}}$.

Encrypt: To encrypt $M \in \{0,1\}^n$, choose a random $r \in \mathbb{Z}_p^*$ and set the ciphertext:

$$C = \langle rP, M \oplus H_2(e(Q_{\mathsf{v}}, P_{\mathsf{pub}})^r) \rangle$$

Decrypt: To decrypt $C = \langle U, V \rangle$ using d_{v} compute

$$V \oplus H_2(e(d_{\mathsf{v}}, U)) = M$$

.

If C is an encryption of M under the public key v then we have

$$e(d_{\mathsf{v}}, U) = e(sQ_{\mathsf{v}}, rP) = e(Q_{\mathsf{v}}, sP)^r = e(Q_{\mathsf{v}}, P_{\mathsf{pub}})^r.$$

Hence the decryption algorithm returns M.

Security of the above scheme against chosen plaintext attack (i.e., IND-ID-CPA security) is proved in [20] assuming $H_1()$ and $H_2()$ to be random oracles. The proof is a reduction and proceeds in two steps. In the first step, a public key encryption scheme BasicPub is defined as follows.

Key-Gen: Let P be a generator of G_1. Choose a random $s \in \mathbb{Z}_p$ and compute $P_{\mathsf{pub}} = sP$; also choose a random $Q_{\mathsf{v}} \in G_1^*$. Next choose a cryptographic hash function $H_2 : G_2 \rightarrow \{0,1\}^n$. The private key is $d_{\mathsf{v}} = sQ_{\mathsf{v}}$ and the public key is $pk = \langle P, P_{\mathsf{pub}}, Q_{\mathsf{v}}, H_2 \rangle$.

Encrypt: Encrypt $M \in \{0,1\}^n$ as $C = \langle rP, M \oplus H_2(e(Q_{\mathsf{v}}, P_{\mathsf{pub}})^r) \rangle$ where r is a random element of $\mathbb{Z}_p^*$.

Decrypt: Decrypt $C = \langle U, V \rangle$ using the private key d_{v} as $V \oplus H_2(e(d_{\mathsf{v}}, U)) = M$.

Suppose, for some identity v in BasicIdent, $H_1(\mathsf{v})$ is mapped to Q_{v} of BasicPub. Then the Key Generation, Encryption and Decryption algorithms of BasicPub essentially corresponds to the respective algorithms of BasicIdent for the identity v.

Let $\mathcal{A}_1$ be an IND-ID-CPA adversary against BasicIdent and $\mathcal{A}_2$ is an IND-CPA adversary against BasicPub, while $\mathcal{B}$ is an algorithm that solves the given BDH problem. The reduction proceeds in two steps. In the first step which we denote as Game 1, $\mathcal{A}_1$ is used to construct $\mathcal{A}_2$. In the next step which we call Game 2, $\mathcal{A}_2$ is used to construct $\mathcal{B}$.

$\mathcal{B}$ plays the role of challenger in the IND-CPA game with $\mathcal{A}_2$. It runs the Key Generation algorithm of BasicPub and gives $pk = \langle P, P_{\mathsf{pub}}, Q_{\mathsf{v}}, H_2 \rangle$ to $\mathcal{A}_2$. The secret key $d_{\mathsf{v}} = sQ_{\mathsf{v}}$ is not revealed. From pk, $\mathcal{A}_2$ passes on P, P_{pub} and H_2 to $\mathcal{A}_1$. $\mathcal{A}_2$ keeps Q_{v} to itself and uses it to form H_1. The crux of the proof in the first step is in the construction of $H_1()$.

To pose as a proper challenger to $\mathcal{A}_1$, $\mathcal{A}_2$ should be able to answer the key extraction queries and also to generate a valid challenge. In simulating $H_1()$, $\mathcal{A}_2$ randomly partitions the identity space $\mathcal{I}$ into two disjoint subsets $\mathcal{I}_1$ and $\mathcal{I}_2$ in such a way that it is able to form a proper private key if and only if the queried identity is from $\mathcal{I}_1$. Similarly, it can form a proper challenge ciphertext if and only if the challenge identity is from $\mathcal{I}_2$. It aborts the game if the key extraction query is for an identity from $\mathcal{I}_2$ or the challenge identity is from $\mathcal{I}_1$. This in turn results in a degradation in the security reduction.

This is an intuitive explanation of the principal strategy in the security reduction of Game 1. In fact, partitioning of the identity space into two disjoint subsets, such that the private key queries can be answered for one subset, while a proper challenge can only be generated for a member of the second – this is a hallmark of the security reduction (with or without random oracle) of all the identity-based encryption schemes that we describe in this chapter.

Given this intuitive understanding, we now proceed for a more formal description.

Game 1

H_1-queries: $\mathcal{A}_1$ can query the random oracle $H_1()$ at any time during the game. $\mathcal{A}_2$ maintains a list called H_1^{list} to answer such queries. The ith entry to the list is a 4-tuple, $\langle \mathsf{v}_i, Q_i, b_i, c_i \rangle \in \{0,1\}^* \times G_1^* \times \mathbb{Z}_p^* \times \{0,1\}$. Suppose $\mathcal{A}_1$ places a query to $H_1()$ for the identity v_j. $\mathcal{A}_2$ responds to this query in the following way.

- If v_j already exists in H_1^{list} as $\langle \mathsf{v}_j, Q_j, b_j, c_j \rangle$ then $\mathcal{A}_2$ returns $H_1(\mathsf{v}_j) = Q_j$.
- Otherwise $\mathcal{A}_2$ takes the following steps.
  - Generate a random $c \in \{0,1\}$ where $\Pr[c = 0] = \delta$ for some δ which is fixed a-priori for all queries.
  - Pick a random $b \in \mathbb{Z}_p^*$ and set $Q_j = bP$ if $c = 0$; otherwise set $Q_j = bQ_{\mathsf{v}}$.
  - Add $\langle \mathsf{v}_j, Q_j, b, c \rangle$ to H_1^{list} and return $H_1(\mathsf{v}_j) = Q_j$ to $\mathcal{A}_1$.

Note that, based on $\Pr[c = 0] = \delta$, the identity space $\mathcal{I}$ is partitioned into two disjoint subsets $\mathcal{I}_1$ and $\mathcal{I}_2$. For identities in $\mathcal{I}_1$, we have $c = 0$ and $\mathcal{A}_2$ can answer the private key extraction queries as we show in the next phase. While for identities in $\mathcal{I}_2$, $c = 1$ and $\mathcal{A}_2$ can generate a proper challenge ciphertext.

Phase 1: Suppose $\mathcal{A}_1$ asks for the private key of v_i in the ith query. $\mathcal{A}_2$ runs the algorithm to answer the H_1-queries. Suppose $\langle \mathsf{v}_i, Q_i, b_i, c_i \rangle$ be the corresponding tuple in H_1^{list}. If $c_i = 1$, $\mathcal{A}_2$ aborts the game. Otherwise, we have $c_i = 0$ and so $Q_i = b_i P$. Define $d_{\mathsf{v}_i} = b_i P_{\mathsf{pub}}$ and return d_{v_i} to $\mathcal{A}_1$. Note that, $d_{\mathsf{v}_i} = b_i sP = sQ_i$, where $H_1(\mathsf{v}_i) = Q_i$. So this is a proper private key for v_i.

Challenge: When $\mathcal{A}_1$ decides that Phase 1 is over; it outputs a challenge identity v^* and two equal length messages M_0, M_1. v^* should not be an identity for which $\mathcal{A}_1$ placed a private key extraction query in Phase 1. $\mathcal{A}_2$ relays M_0, M_1 as its own challenge to $\mathcal{B}$. $\mathcal{B}$ chooses γ uniformly at random from $\{0, 1\}$ and responds with a $\mathsf{BasicPub}$ ciphertext $C = \langle U, V \rangle$ of M_γ. Next, $\mathcal{A}_2$ runs the H_1-queries to find a tuple $\langle \mathsf{v}^*, Q, b, c \rangle$ in the H_1^{list}. If $c = 0$, $\mathcal{A}_2$ aborts the game. Otherwise, $c = 1$, so $Q = bQ_{\mathsf{v}}$ and $H_1(\mathsf{v}^*) = Q$. $\mathcal{A}_2$ now sets $C^* = \langle b^{-1}U, V \rangle$ and returns C^* to $\mathcal{A}_1$ as the challenge ciphertext.

Let $U = rP$ and $V = M_\gamma \oplus H_2(e(Q_{\mathsf{v}}, P_{\mathsf{pub}})^r$ for some random $r \in \mathbb{Z}_p$. $\mathcal{A}_2$ sets $U' = b^{-1}U = b^{-1}rP = r'P$. So $e(Q, P_{\mathsf{pub}})^r = e(b^{-1}Q, P_{\mathsf{pub}})^r = e(Q, P_{\mathsf{pub}})^{b^{-1}r} = e(Q, P_{\mathsf{pub}})^{r'}$. Hence, $C^* = \langle U', V \rangle$ is a proper encryption of M_γ under the identity v^*.

Phase 2: $\mathcal{A}_1$ places additional private key extraction queries with the restriction that it cannot ask for the private key of v^*. $\mathcal{A}_2$ responds as in Phase 1.

Guess: Finally $\mathcal{A}_1$ outputs its guess γ' for γ. $\mathcal{A}_2$ relays this γ' as its own guess for γ.

If $\mathcal{A}_2$ does not abort the above game, then from the view point of $\mathcal{A}_1$ the situation is identical to that of a real attack. So we have

$$|\Pr[\gamma = \gamma'] - 1/2| \geq \epsilon$$

where ϵ is the advantage of $\mathcal{A}_1$ against $\mathsf{BasicIdent}$. Boneh and Franklin show that the probability that $\mathcal{A}_2$ does not abort is $\delta^q(1 - \delta)$, where q is the number of key extraction queries and a lower bound is $\Pr[\overline{\mathsf{abort}}] \geq 1/(e \times (1 + q))$, where e is the base of natural logarithms and should not be confused with the notation $e()$ of bilinear map. Hence, $\mathcal{A}_2$'s advantage against $\mathsf{BasicPub}$ is at least $\epsilon/(e \times (1 + q))$. The above technique is similar to Coron's analysis of Full Domain Hash Signature [35].

In the next step, Boneh-Franklin constructs an algorithm $\mathcal{B}$ which solves the BDH problem given the adversary $\mathcal{A}_2$ against $\mathsf{BasicPub}$; relating the advantage of $\mathcal{B}$ with that of $\mathcal{A}_2$. The details of the reduction follow.

Game 2

Suppose $\mathcal{B}$ is given a BDH problem instance $\langle P, aP, bP, cP\rangle$. Its task is to compute $Z = e(P,P)^{abc}$. $\mathcal{B}$ tries to solve this problem by interacting with $\mathcal{A}_2$ in the IND-CPA game.

Setup: $\mathcal{B}$ forms the BasicPub public key $K_{\mathsf{pub}} = \langle P, P_{\mathsf{pub}}, Q_{\mathsf{v}}, H_2\rangle$ where $P_{\mathsf{pub}} = aP$, $Q_{\mathsf{v}} = bP$ and H_2 is a random oracle controlled by $\mathcal{B}$. The private key $d_{\mathsf{v}} = abP$ is unknown to $\mathcal{B}$.

H_2-queries: To respond to $\mathcal{A}_2$'s queries to the random oracle $H_2()$, $\mathcal{B}$ maintains a list called H_2^{list}. The ith entry to the list is a tuple $\langle X_i, H_i\rangle \in G_1^* \times \{0,1\}^n$. $\mathcal{B}$ responds to any query for X_j in the following manner:

- If there is already a tuple $\langle X_j, H_j\rangle$ in H_2^{list}, then return $H_2(X_j) = H_j$.
- Otherwise, pick a random $H_j \in \{0,1\}^n$, add $\langle X_j, H_j\rangle$ to H_2^{list} and then return $H_2(X_j) = H_j$.

Challenge: $\mathcal{A}_2$ outputs two n-bit messages M_0, M_1. $\mathcal{B}$ picks a random string $R \in \{0,1\}^n$ and forms the ciphertext $C = \langle cP, R\rangle$ and returns it to $\mathcal{A}_2$. Note that, decryption of C is $R \oplus H_2(e(cP, d_{\mathsf{v}})) = R \oplus H_2(e(P,P)^{abc}$.

Guess: $\mathcal{A}_2$ outputs its guess $\gamma' \in \{0,1\}$.
$\mathcal{B}$ picks a random tuple $\langle X_i, H_i\rangle$ from H_2^{list} and outputs X_i as the solution of the given BDH problem.

This completes the description of Game 2. Boneh and Franklin next go on to estimate the advantage of $\mathcal{B}$ against the BDH problem. This is done by comparing $\mathcal{A}_2$'s behavior in the above game with it's behavior in a real IND-CPA attack.

Let $\mathcal{H}$ be the event that $\mathcal{A}_2$ places a query for $Z = e(P,P)^{abc}$ to H_2 at some point in the above game. By induction on the number of queries Boneh-Franklin show that $\Pr[\mathcal{H}]$ in Game 2 is equal to $\Pr[\mathcal{H}]$ in a real attack. In the next step, they show that in a real attack $\Pr[\mathcal{H}] \geq 2\epsilon'$ where ϵ' is the advantage of $\mathcal{A}_2$ against BasicPub. In Game 2, $\mathcal{A}_2$ makes at most q_{H_2} queries to H_2. So, the probability that $\mathcal{B}$ outputs Z is at least $2\epsilon'/q_{H_2}$.

Combining the analysis of Game 1 and 2, one comes to the final conclusion:

$$\mathsf{Adv}_{\mathcal{A}_1}^{\mathsf{BasicIdent}} \leq \frac{1}{2} e \times (1+q) q_{H_2} \epsilon_{\mathsf{BDH}}$$

Note that, the security degrades by roughly a factor of $(1+q)q_{H_2}$ which is the product degradation in the reduction of Game 1 and 2.

Boneh and Franklin next propose an IBE with chosen ciphertext security: FullIdent in their terminology. FullIdent is the result of applying the so called Fujisaki-Okamoto transformation [44] to BasicIdent. We do not provide details of the construction. Interested readers are referred to the original work of Boneh and Franklin [20] and also to Galindo's work [47], who fixed a bug in the original reduction. Further works on reducing the degradation in the security reduction of the Boneh and Franklin scheme is also available in the literature [60, 3]

3.1.1 Hierarchical Identity-Based Encryption

The concept of IBE has been generalised to hierarchical identity-based encryption (HIBE) by Horwitz-Lynn and Gentry-Silverberg [56, 49]. Extending in the line of Boneh-Franklin, Horwitz and Lynn gave precise definitions of HIBE and its security model. They also proposed a two-level HIBE that achieves total collusion resistance in the first level and partial collusion resistance in the second level. This means a certain number of users at the second level can collude to find the private key of their ancestor at the first level. Their scheme was proved secure using random oracles.

Gentry and Silverberg proposed a HIBE [49] scheme that is totally collusion resistant for arbitrary number of levels. This too was proved secure using random oracles. The Gentry-Silverberg HIBE bears much resemblance with the Boneh-Franklin HIBE in terms of construction as well as security proof. We now detail their scheme followed by the essential idea of the security reduction.

BasicHIBE

Setup: Let P be an arbitrary generator of G_1. Pick a random $x_0 \in \mathbb{Z}_p^*$ and set $P_{\mathsf{pub}} = x_0 P$. Also choose cryptographic hash functions $H_1 : \{0,1\}^* \to G_1$ and $H_2 : G_2 \to \{0,1\}^n$. The public parameters are $\mathsf{PP} = \langle P, P_{pub}, H_1, H_2 \rangle$, while the master secret is x_0.

Key-Gen: An identity v at the jth level is represented as $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$. The PKG chooses random elements $x_1, \ldots, x_{j-1} \in \mathbb{Z}_p^*$ and computes $d_i = x_i P$ for $1 \leq i \leq j-1$ and $d_j = \sum_{i=1}^{j} x_{i-1} Q_i$, where $Q_i = H_1(\mathsf{v}_1, \ldots, \mathsf{v}_i)$. It gives $d_{\mathsf{v}} = \langle d_1, \ldots, d_j \rangle$ to v.

A private key for v can also be generated by its parent. Let $\mathsf{ID}|_{j-1} = (\mathsf{v}_1, \ldots, \mathsf{v}_{j-1})$ be the parent of v, i.e., $\mathsf{ID}|_{j-1}$ is one level up in the hierarchy with respect to v. Let the private key of $\mathsf{v}|_{j-1}$ be $d_{\mathsf{v}|_{j-1}} = (d'_1, \ldots, d'_{j-1})$. Then d_{v} can be formed by $\mathsf{ID}|_{j-1}$ as follows: Compute $Q_{\mathsf{v}} = H_1(\mathsf{v}_1, \ldots, \mathsf{v}_j)$, choose a random $x_{j-1} \in \mathbb{Z}_p^*$ and set $d_j = d'_{j-1} + x_{j-1} Q_{\mathsf{v}}$ and set $d_i = d'_i$ for $1 \leq i \leq j-2$ and $d_{j-1} = x_{j-1} P$. The private key, $d_{\mathsf{v}} = \langle d_1, \ldots, d_j \rangle$ is given to v.

Encrypt: To encrypt M under the identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$, compute $Q_i = H_1(\mathsf{v}_1, \ldots, \mathsf{v}_i)$ for $1 \leq i \leq j$. Then choose a random $r \in \mathbb{Z}_p^*$ and set the ciphertext

$$C = \langle rP, rQ_2, \ldots, rQ_j, M \oplus H_2(e(P_{\mathsf{pub}}, Q_1)^r) \rangle$$

Decrypt: Given $C = \langle U_0, U_2, \ldots, U_j, V \rangle$ and $d_{\mathsf{v}} = \langle d_1, \ldots, d_j \rangle$, compute

$$V \oplus H_2 \left(\frac{e(U_0, d_j)}{\prod_{i=2}^{j} e(d_{i-1}, U_i)} \right) = M$$

If the HIBE is restricted to the first level only, then this is exactly the BasicIdent scheme of Boneh and Franklin [20].

Security of BasicHIBE against chosen plaintext attack (i.e., IND-ID-CPA security) can be proved in the same manner as that of the BasicIdent of previous section. In the first stage, an IND-ID-CPA adversary, $\mathcal{A}_1$ against BasicHIBE is used to construct an IND-CPA adversary $\mathcal{A}_2$ against BasicPub (the same public key encryption scheme defined in the context of Boneh-Franklin IBE). In the next stage, this $\mathcal{A}_2$ is utilised to construct an algorithm $\mathcal{B}$ that solves the BDH problem.

However, Gentry and Silverberg first prove the security in the non-adaptive setting and later extend it to the adaptive setting. In the non-adaptive setting, $\mathcal{A}_1$ a-priori fixes a target identity $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_j^*)$, $j \geq 1$. Given v^*, $\mathcal{A}_2$ forms the H_1^{list} in such a way that given any identity v it can generate the private key of v, provided v is not a prefix of v^*. Similarly in the challenge phase, it can generate a proper encryption of M_γ under v^*, given a proper encryption of M_γ in BasicPub. This way, the advantage of $\mathcal{A}_1$ against BasicHIBE can be directly converted into the advantage of $\mathcal{A}_2$ against BasicPub without any degradation.

The security reduction in the adaptive setting, however, suffers from a large degradation factor. Let's briefly see why this is so. Here, instead of a single identity, we have an identity tuple of arbitrary levels. Recall that in the reduction for BasicIdent of Section 3.1, a typical entry in H_1^{list} is of the form $\langle \mathsf{v}, Q, b, c \rangle$. In case of BasicHIBE an entry for $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ is of the form $\langle \mathsf{v}_i \rangle, \langle Q_{\mathsf{v}_i} \rangle, \langle b_i \rangle, \langle c_i \rangle$ where $1 \leq i \leq j$, i.e., each $\langle \cdots \rangle$ contains j many entries, whereas in case of BasicIdent of Section 3.1 they contained only a single term.

Let the target identity be $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_h^*)$ then the corresponding entry in the H_1^{list} has terms $c_1, \ldots, c_h \in \{0,1\}^h$. So, instead of a single c, we have to maintain h many c_is in H_1^{list}. $\mathcal{A}_2$ can generate a proper challenge ciphertext if and only if all these h c_is have the same value, 1. But these c_is are chosen independently at random, so the probability that all are 1 is the product of the probabilities that each is 1. Hence, we get a security degradation which is exponential in the number of levels in the target identity tuple.

Once $\mathcal{A}_2$ has been constructed, either in the adaptive or the non-adaptive setting, the next stage of the reduction is exactly that of Game 2 in case of Boneh-Franklin IBE. Finally, in the adaptive setting one gets the following result:

$$\mathsf{Adv}_{\mathcal{A}_1}^{\mathsf{BasicHIBE}} \leq \frac{(e \times (q+h))^h q_{H_2}}{h} \epsilon_{BDH}$$

where e is the base of natural logarithm.

This means, if there is an IND-ID-CPA adversary $\mathcal{A}_1$ in the adaptive setting having advantage ϵ against BasicHIBE and that makes at most q_{H_2} queries to H_2 and q private key extraction queries and H_1, H_2 are random oracles then there is an algorithm $\mathcal{B}$ that solves the BDH problem with advantage at least $(\epsilon(h/e \times (q+h))^h) q_{H_2}^{-1}$, where h is the number of levels in the target identity. So the security degrades exponentially with the number of levels of the HIBE.

Applying the Fujisaki-Okamoto transformation to the BasicHIBE, Gentry and Silverberg obtains an IND-ID-CCA secure HIBE which is called FullHIBE. The construction as well as the security proof is analogous to that of FullIdent and not detailed here. Similarly, Galindo's observation [47] with respect to FullIdent is also applicable for the scheme FullHIBE.

3.2 From Random Oracle to Standard Model

Both the Boneh-Franklin IBE and the Gentry-Silverberg HIBE schemes depend on the random oracle heuristic for their proof of security. The first move to get rid of the random oracle assumption was taken by Canetti, Halevi and Katz [26, 27]. However, they had to weaken the security notion from the full model to the so called *selective*-ID model of Section 2.5.3 while moving in this goal.

We have already observed that two random oracles (namely H_1, H_2) are used in the security proof of BasicIdent and BasicHIBE. We need another two random oracles, H_3 and H_4 to achieve CCA security for the above protocols. These four random oracle serve three distinct purposes. H_1 is used to map an identity to a random element of G_1, H_2 is used to relate the security of the scheme with that of BDH problem while H_3, H_4 come into play because of the Fujisaki-Okamoto transformation to achieve chosen ciphertext security from a given chosen plaintext secure scheme. H_2 can be done away with if one is prepared to rely on the (presumably stronger) decisional bilinear Diffie-Hellman assumption (DBDH). Canetti, Halevi and Katz [27] proposed a different generic transformation to achieve CCA security, thereby removing the need to rely on random oracles H_3, H_4. The efficiency of this generic transformation was later improved by Boneh and Katz [21]. We discuss this generic transformation in Section 3.5. Here we concentrate on constructions that achieve security against chosen plaintext attack under the *selective*-ID model.

By restricting the adversary's behaviour, it is possible to construct protocols secure in the *selective*-ID model that avoid any degradation in the security reduction. The BasicHIBE of Gentry-Silverberg [49] secure in the non-adaptive setting can be seen as secure in the *selective*-ID model, though the model was not formalised when the scheme was first proposed.

Canetti, Halevi and Katz [26] introduced the notion of binary tree encryption (BTE). A BTE differs from HIBE in the sense that each node in the BTE has two children – the left child and the right child. BTE is a public key encryption scheme and they also show [26] how an (H)IBE can be constructed from any BTE. This gives the first (H)IBE construction that is secure without random oracle in the *selective*-ID model. We do not detail their construction and security proof here. One limitation of the construction is large computational overhead – one pairing computation for each bit of identity during decryption.

3.2.1 *Selective*-ID Secure HIBE

Boneh and Boyen proposed two efficient IBE schemes that are secure in the *selective*-ID model without random oracle. The first construction was presented as an HIBE which we call BB-HIBE. We give a detailed description of the BB-HIBE protocol and its security reduction below. One reason being that we generalize this protocol to propose two new constructions in Chapter 7. Another reason is that the algebraic technique introduced in this work for private key extraction was adapted in later works (including ours).

BB-HIBE

Here individual components of an identity tuple are elements of $\mathbb{Z}_p$.

Setup Select a random generator $P \in G_1^*$, a random $x \in \mathbb{Z}_p$ and set $P_1 = xP$. Also pick random elements $Q_1, \ldots, Q_h, P_2 \in G_1$. Then

$$\mathsf{PP} = \langle P, P_1, P_2, Q_1, \ldots, Q_h \rangle; \quad \mathsf{msk} = xP_2$$

The maximum height of the HIBE is h. Define publicly computable family of functions $F_j : \mathbb{Z}_p \to G_1$ for $j \in \{1, \ldots, h\}$: $F_j(\alpha) = \alpha P_1 + Q_j$.

Key-Gen: Given an identity $\mathsf{v} = \langle \mathsf{v}_1, \ldots, \mathsf{v}_j \rangle$ of depth $j \leq h$, pick random $r_1, \ldots, r_j \in \mathbb{Z}_p$ and compute

$$d_{\mathsf{v}} = \left(xP_2 + \sum_{i=1}^{j} r_i F_i(\mathsf{v}_i), r_1 P, \ldots, r_j P \right)$$

d_{v} can also be generated given the private key $d_{\mathsf{v}|j-1}$ of $\mathsf{v}|_{j-1} = \langle \mathsf{v}_1, \ldots, \mathsf{v}_{j-1} \rangle$.

Encrypt: Encrypt $M \in G_2$ for $\mathsf{v} = \langle \mathsf{v}_1, \ldots, \mathsf{v}_j \rangle$ as

$$C = (e(P_1, P_2)^s \cdot M, sP, sF_1(\mathsf{v}_1), \ldots, sF_j(\mathsf{v}_j))$$

where s is a random element of $\mathbb{Z}_p$.

Decrypt: Decrypt $C = \langle A, B, C_1, \ldots, C_j \rangle$ using the private key d_{v} as

$$A \cdot \frac{\prod_{i=1}^{j} e(C_i, d_i)}{e(B, d_0)} = M$$

Security

CPA security of BB-HIBE is proved in the *selective*-ID model. Let $\mathcal{A}$ be an adversary against the above HIBE with advantage ϵ. $\mathcal{A}$ commits to an identity tuple before the system is set-up. In the security reduction, $\mathcal{A}$ is used to construct an algorithm $\mathcal{B}$ that solves the DBDH problem. $\mathcal{B}$ is given as input a 5-tuple $\langle P, aP, bP, cP, Z \rangle$. The task of $\mathcal{B}$ is to determine whether Z is equal to $e(P, P)^{abc}$ or a random element of G_2. $\mathcal{B}$ solves this problem by interacting with $\mathcal{A}$ in the *selective*-ID game. The essential idea is to form the public parameters using the target identity tuple and the DBDH instance in such a way that all the key extraction queries of $\mathcal{A}$ (except on any prefix of the target identity) can be answered by $\mathcal{A}$. A valid challenge, on the other hand, can be generated for the target identity only. So, based on the target identity, the identity space is partitioned into two disjoint subsets.

Initialization: $\mathcal{A}$ commits to a target identity $\mathsf{v}^* = \langle \mathsf{v}_1^*, \ldots, \mathsf{v}_{h'}^* \rangle$ of height $h' \leq h$. If $h' < h$, $\mathcal{B}$ adds extra random elements from $\mathbb{Z}_p$ to make v^* an identity of height h. Let us denote these extra $(h - h')$ elements by $\mathsf{v}_{h'+1}^*, \ldots, \mathsf{v}_h^*$.

Setup: $\mathcal{B}$ sets $P_1 = aP$ and $P_2 = bP$. It then picks random $\alpha_1, \ldots, \alpha_h \in \mathbb{Z}_p$ and defines $Q_j = \alpha_j P - \mathsf{v}_j^* P_1$ for $1 \leq j \leq h$. It gives $\mathcal{A}$ the public parameters $\mathsf{PP} = \langle P, P_1, P_2, Q_1, \ldots, Q_h \rangle$. Here the $\mathsf{msk} = aP_2 = abP$ is unknown to $\mathcal{B}$. Define the function $F_j(x) = xP_1 + Q_j = (x - \mathsf{v}_j^*)P_1 + \alpha_j P$ for $1 \leq j \leq h$.

Phase 1: $\mathcal{A}$ makes up to q private key queries. In a private key query corresponding to an identity $\mathsf{v} = \langle \mathsf{v}_1, \ldots, \mathsf{v}_u \rangle$, with $u \leq h$ the only restriction is that v is not a prefix of v^*. Let, j be the smallest index such that $\mathsf{v}_j \neq \mathsf{v}_j^*$. $\mathcal{B}$ chooses random $r_1, \ldots, r_j \in \mathbb{Z}_p$ and first computes

$$\begin{aligned}
d_{0|j} &= \frac{-\alpha_j}{(\mathsf{v}_j - \mathsf{v}_j^*)} P_2 + r_j F_j(\mathsf{v}_j) \\
&= \frac{-\alpha_j}{(\mathsf{v}_j - \mathsf{v}_j^*)} P_2 + r_j((\mathsf{v}_j - \mathsf{v}_j^*)P_1 + \alpha_j P) \\
&= abP - abP + \frac{-\alpha_j}{(\mathsf{v}_j - \mathsf{v}_j^*)} bP + r_j((\mathsf{v}_j - \mathsf{v}_j^*)P_1 + \alpha_j P) \\
&= aP_2 + \left(r_j - \frac{b}{\mathsf{v}_j - \mathsf{v}_j^*} \right) ((\mathsf{v}_j - \mathsf{v}_j^*)P_1 + \alpha_j P) \\
&= aP_2 + \tilde{r}_j F_j(\mathsf{v}_j)
\end{aligned}$$

where $\tilde{r}_j = r_j - \frac{b}{\mathsf{v}_j - \mathsf{v}_j^*}$. So $\mathcal{B}$ forms the private key of $\langle \mathsf{v}_1, \ldots, \mathsf{v}_j \rangle$as

$$d_0 = d_{0|j} + \sum_{i=1}^{j-1} r_i F_i(\mathsf{v}_i), d_1 = r_1 P, \ldots, d_{j-1} = r_{j-1} P, d_j = -\frac{1}{\mathsf{v}_j - \mathsf{v}_j^*} P_2 + r_j P = \tilde{r}_j P$$

It is easy to verify that $\langle d_0, d_1, \ldots, d_j \rangle$ is a valid private key for $\langle \mathsf{v}_0, \ldots, \mathsf{v}_j \rangle$. From this $\mathcal{B}$ forms a private key for v and return it to $\mathcal{A}$.

Note that, $\mathcal{B}$ can derive a valid private key for an identity v without the knowledge of the master secret. This is possible as long as v is not a prefix of v^*. The above algebraic technique of private key derivation is one of the major technical novelties of this work. Also note that, if the original target identity $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_{h'}^*)$ is of height less than h, then $\mathsf{v} = (\mathsf{v}_1^*, \ldots, \mathsf{v}_{h'}^*, \mathsf{v}_{h'+1}^*, \ldots, \mathsf{v}_h^*)$ can be a valid query for private key extraction. In such eventuality, $\mathcal{B}$ has to abort the game. Probability of such eventuality is, however, very low – of the order q/p.

Challenge: After completion of Phase 1, $\mathcal{A}$ outputs two messages $M_0, M_1 \in G_2$. $\mathcal{B}$ chooses a random bit γ and forms the ciphertext $C = \langle M_\gamma \cdot Z, cP, \alpha_1 cP, \ldots, \alpha_{h'} cP \rangle$. Note that,

$F_i(\mathsf{v}_i^*) = \alpha_i P$, so

$$C = \langle M_\gamma \cdot Z, cP, cF_1(\mathsf{v}_1^*), \ldots, cF_{h'}(\mathsf{v}_{h'}^*)\rangle.$$

If $Z = e(P,P)^{abc} = e(P_1, P_2)^c$ then C is a valid encryption of M_γ.

Phase 2: $\mathcal{A}$ makes additional queries which $\mathcal{B}$ answers just like Phase 1. Total number of queries in Phase 1 and 2 together should not exceed q.

Guess: Eventually, $\mathcal{A}$ outputs its guess γ' of γ. If $\gamma' = \gamma$, $\mathcal{B}$ outputs 1, otherwise it outputs 0.

When $Z = e(P,P)^{abc}$, then $\mathcal{A}$'s view in the above game is identical to that in a real attack. In that case $|\Pr[\gamma = \gamma'] - 1/2| \geq \epsilon$. on the other hand if Z is a random element of G_2 then $\Pr[\gamma = \gamma'] = 1/2$. Hence we get,

$$\mathsf{Adv}_{\mathcal{B}}^{\mathsf{DBDH}} \geq \epsilon.$$

In other words, if the (t, ϵ)-DBDH assumption holds in G_1, G_2 then the h-HIBE of Boneh-Boyen is (t', q, ϵ)-IND-sID-CPA secure for arbitrary h and q and any $t' < t - O(\tau hq)$ where τ is the time for a scalar multiplication in G_1.

3.3 Full Model

Boneh and Boyen were first to propose an IBE [18] whose proof of security in the full model does not rely on the random oracle heuristic. The construction, however, is not very efficient and as the authors observed, should be seen as a proof of concept. We reproduce their construction because of its chronological importance.

Construction

Here the identities are elements of $\{0,1\}^w$. These identities are mapped to a random n bit string through a hash function H_k. H_k is chosen from a family of hash functions $\{H_k : \{0,1\}^w \to \{0,1\}^n\}_{k \in \mathcal{K}}$, where $\mathcal{K}$ is the key space for the family of hash functions.

Setup: Choose an arbitrary generator $P \in G_1$, pick a random $x \in \mathbb{Z}_p$ and set $P_1 = xP$. Also choose a random element $P_2 \in G_1$. Construct a random $n \times 2$ matrix $\mathcal{U} = (U_{i,j}) \in G_1^{n \times 2}$ where each $U_{i,j}$ is uniform in G_1. Finally pick a random hash function key $k \in \mathcal{K}$. The public parameters are $\mathsf{PP} = \langle P, P_1, P_2, \mathcal{U}, k\rangle$ and the master key is $\mathsf{msk} = xP_2$.

Key-Gen: To generate the private key d_{v} for an identity $\mathsf{v} \in \{0,1\}^w$, compute $\overrightarrow{a} = H_k(\mathsf{v}) = a_0, \ldots, a_n \in \{0,1\}^n$ and pick random $r_1, \ldots, r_n \in \mathbb{Z}_p$. The private key is $d_{\mathsf{v}} = \langle xP_2 + \sum_{i=1}^n r_i U_{i,a_i}, r_1 P, \ldots, r_n P\rangle$. Note that the private key consists of $n+1$ elements of G_1.

Encrypt: To encrypt a message $M \in G_2$ for the identity $\mathsf{v} \in \{0,1\}^w$, set $\overrightarrow{a} = H_k(\mathsf{v}) = a_0, \ldots, a_n$, pick a random $s \in \mathbb{Z}_p$ and compute

$$C = \langle e(P_1, P_2)^s \times M, sP, sU_{1,a_1}, \ldots, sU_{n,a_n} \rangle \in G_2 \times G_1^{n+1}$$

Decrypt: To decrypt $C = \langle A, B, C_1, \ldots, C_n \rangle$ using the private key $d_{\mathsf{v}} = \langle d_0, d_1, \ldots, d_n \rangle$ compute

$$A \times \frac{\prod_{j=1}^{n} e(C_j, d_j)}{e(B, d_0)} = M.$$

It is easy to verify that this gives a proper decryption.

Security

We only provide an intuitive understanding of the security reduction. This construction has much resemblance with the BB-HIBE discussed in the previous section. Consider a BB-HIBE of maximum depth n where all the identity tuples are of the form $\mathsf{v} = (i_1, \ldots, i_n)$, i.e., we allow only the identity tuples of full depth. Then what we essentially get is the above construction. Here, however, $i_j \in \{0,1\}$ where as in the original BB-HIBE construction, domain of the identities is $\mathbb{Z}_p$. Boneh-Boyen use additional randomization, such as using the hash function $H_k()$ and taking an $n \times 2$ matrix $\mathcal{U}$, to tackle this situation.

The proof idea also follows from that of the BB-HIBE. However, recall that the earlier construction is proved to be secure only in the *selective*-ID model. Some additional subtleties are needed to achieve security in the full model as we presently discuss. Suppose the challenger $\mathcal{B}$ chooses an n bit string $\mathsf{v}^c = i_1^c, \ldots, i_n^c$ at random. $\mathcal{B}$ then forms the matrix $\mathcal{U}$ in such a way that it can form the private key for any identity $\mathsf{v} = (i_1, \ldots, i_n)$ only if there is some j, $1 \leq j \leq n$ such that $i_j = i_j^c$. In challenge phase it can form a proper encryption only if the challenge identity, v^* is the complement string of v^c, i.e., there is no j such that $\mathsf{v}_j^* = \mathsf{v}_j^c$ for $1 \leq j \leq n$. This is sort of plug-n-pray technique. The security reduction suffers from a (huge) degradation because the simulator can generate a proper encryption only for the identity plugged in, i.e., v^*.

3.3.1 Waters' Protocol

The public parameters, private key and ciphertext sizes are quite large for the Boneh-Boyen IBE. Waters introduced a nice protocol [89] where the private key and ciphertext sizes are drastically reduced. In this protocol, identities are represented as bit strings of length n.

Setup: Randomly choose a secret $x \in \mathbb{Z}_p$. Set $P_1 = xP$, then choose $P_2 \in G_1$ at random. Further, choose a random element $U' \in G_1$ and a random n-length vector $\overrightarrow{U} = \{U_1, \ldots, U_n\}$, whose elements are from G_1. The master secret is xP_2 whereas the public parameters are $\langle P, P_1, P_2, U', \overrightarrow{U} \rangle$.

Key-Gen: Let $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_n) \in \{0,1\}^n$ be any identity. A secret key for v is generated as follows. Choose a random $r \in \mathbb{Z}_p^*$, then the private key for v is

$$d_{\mathsf{v}} = (xP_2 + rV, rP).$$

where

$$V = U' + \sum_{\{i:\mathsf{v}_i=1\}} U_i.$$

Encrypt: Any message $M \in G_2$ is encrypted for an identity v as

$$C = (e(P_1, P_2)^t M, tP, tV),$$

where t is a random element of $\mathbb{Z}_p$ and V is as defined in key generation algorithm.

Decrypt: Let $C = (C_1, C_2, C_3)$ be a ciphertext and v be the corresponding identity. Then we decrypt C using secret key $d_{\mathsf{v}} = (d_1, d_2)$ by computing

$$C_1 \frac{e(d_2, C_3)}{e(d_1, C_2)}.$$

We give an intuitive understanding of the security reduction of this protocol. One of the contributions of the present work is a generalisation of Waters construction. We give a security proof of this generalised construction in Chapter 5.

Waters construction has some similarity with the BB-HIBE of Section 3.2. Using a similar algebraic technique of Boneh-Boyen, Waters forms a simulator $\mathcal{B}$, that solves the DBDH problem given an adversary $\mathcal{A}$ against the IBE with advantage ϵ.

In the simulation, $\mathcal{B}$ constructs a function $F : \mathcal{I} \to \mathbb{Z}_p$, where $\mathcal{I}$ is the set of identities, in such a way that given an identity v it can form a proper private key d_{v} only if $F(\mathsf{v}) \neq 0$. In contrast, it can form a proper challenge for an identity v^* only if $F(\mathsf{v}^*) = 0$. We have already observed that this complementary condition for key generation and challenge generation is a hallmark of all the encryption protocols described so far. Because of this complementary condition there are certain identities for which the simulator cannot generate the private key and for some other identities it is unable to generate a proper challenge. In such situations, the simulator has to abort the game and we have no way to correlate the adversarial advantage against the encryption protocol to that of solving the underlying hard problem. This is the cause of degradation in the security reduction.

The complementary condition of key generation and target generation is also true for the protocols secure under the *selective*-ID model. The simulator cannot generate the private key of the target identity or any of its prefix. In sID model, however, we do not have any security degradation because of the restriction of the model. The adversary commits to a target identity ahead of the system setup. So the simulator chooses the system parameters in such a way that it can answer all the key extraction queries of the adversary and also generate the target ciphertext with probability one.

3.4 HIBE with Shortened Ciphertext

3.4.1 Constant Size Ciphertext HIBE

Boneh, Boyen and Goh proposed a HIBE in the *selective*-ID model where the length of the ciphertext is always constant. We refer to this protocol as BBG-HIBE. The construction is described below.

Identities at a depth u are of the form $(\mathsf{v}_1, \ldots, \mathsf{v}_u) \in (\mathbb{Z}_p^*)^u$. Messages are elements of G_2.

Setup: Let $\langle P \rangle = G_1$. Choose a random $\alpha \in \mathbb{Z}_p$ and set $P_1 = \alpha P$. Choose random elements $P_2, P_3, Q_1, \ldots, Q_h \in G_1$. Set the public parameter as $\mathsf{PP} = (P, P_1, P_2, P_3, Q_1, \ldots, Q_h)$ while the master key is αP_2.

Key-Gen: Given an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k) \in (\mathbb{Z}_p^*)^k$ of depth $k \leq h$, pick a random $r \in \mathbb{Z}_p$ and output

$$d_{\mathsf{v}} = (\alpha P_2 + r(\mathsf{v}_1 Q_1, \ldots, I_k Q_k + P_3), rP, rQ_{k+1}, \ldots, rQ_h).$$

The private key for v can also be generated given the private key for $\mathsf{v}_{|k-1}$ as is the general requirement of any HIBE.

Encrypt: To encrypt $M \in G_2$ under the identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k) \in (\mathbb{Z}_p^*)^k$, pick a random $s \in \mathbb{Z}_p$ and output

$$\mathsf{CT} = \left(e(P_1, P_2)^s \times M, sP, s(\mathsf{v}_1 Q_1 + \ldots + \mathsf{v}_k Q_k + P_3)\right).$$

Decrypt: To decrypt $\mathsf{CT} = (A, B, C)$ using the private key $d_{\mathsf{v}} = (a_0, a_1, b_{k+1}, \ldots, b_h)$, compute

$$A \times \frac{e(a_1, C)}{e(B, a_0)} = M.$$

Note that, apart from the masked message, the ciphertext in BBG-HIBE consists of only two elements of G_1 irrespective of the number of components in the corresponding identity. In other HIBEs, the length of the ciphertext is proportional to the length of the identity tuple. The BBG-HIBE offers new and important applications for constructing other cryptographic primitives like forward secure encryption [26] and broadcast encryption [78, 38].

Security of BBG-HIBE against adaptive chosen plaintext attack is proved in the selective-ID model under the h-wDBDHI* assumptions described in Section 2.2.2. The security reduction uses an algebraic technique similar to that of BB-HIBE. We do no provide the details of the reduction. In Chapter 8 and Chapter 9 we augment the BBG-HIBE to stronger security models with detailed argument about the reductions.

Composite HIBE

Boneh, Boyen and Goh also suggested a "product" construction of the constant ciphertext BBG-HIBE and BB-HIBE [19]. In case of BBG-HIBE the private key size decreases with the increase in identity level. While in case of BB-HIBE the private key size increases with the height of an identity. Utilizing the algebraic similarities of both the systems they construct a composite scheme where the inner HIBE is the BBG-HIBE and the outer HIBE is the BB-HIBE. The composite scheme allows a trade-off between the ciphertext size and the private key size. We suggest a variant of this composite construction in Chapter 9.

3.5 Chosen Ciphertext Security

Security against chosen-ciphertext attack (IND-ID-CCA security) is the strongest notion of security for any (hierarchical) identity-based encryption scheme. We have already observed that the initial proposals such as the Boneh-Franklin IBE and Gentry-Silverberg HIBE used the Fujisaki-Okamoto transformation to their basic schemes secure in the sense of IND-ID-CPA to achieve this goal. However, the Fujisaki-Okamoto transformation uses cryptographic hash functions that are modelled as random oracles. Protocols that achieve security against chosen-plaintext attack without random oracle require a different technique to achieve chosen-ciphertext security.

Canetti, Halevi and Katz introduced a generic transformation [27] to achieve CCA security. Given any $(h+1)$ level HIBE which is secure against chosen-plaintext attack, this generic transformation yields an h level HIBE which is secure against chosen ciphertext attack. This transformation uses a strongly unforgable one time signature scheme. Boneh and Katz suggested a modification [21] where the one time signature scheme is replaced by a MAC, there by increasing the efficiency of the transformation.

We now detail the signature based approach. A signature scheme is defined by three probabilistic polynomial time algorithms as follows:

Key-Gen: On input the security parameter 1^κ, this probabilistic polynomial time algorithm outputs a pair of signing key (sk) and verification key (vk).

Sign: This algorithm takes input a signing key sk and a message M from the appropriate message space $\mathcal{M}$ and outputs a signature σ.

Verify: This is a deterministic algorithm which on input a verification key vk, a message M and a signature σ on M outputs accept or reject depending on whether σ is a proper signature on M or not.

A signature scheme (Key-Gen, Sign, Verify) is a strong, one-time scheme if the success probability of any probabilistic polynomial time adversary $\mathcal{A}$ is negligible in κ in the following game.

1. Key-Gen(1^κ) outputs (vk, sk). The adversary $\mathcal{A}$ is given κ and vk.

2. $\mathcal{A}(1^\kappa, vk)$ may take one of the following actions:

   (a) $\mathcal{A}$ outputs a pair (M^*, σ^*) and halts. In this case (M, σ) is undefined.

   (b) $\mathcal{A}$ outputs a message M and in return is given a signature of M under the signing key sk, i.e., $\sigma \leftarrow Sign_{sk}(M)$. Then $\mathcal{A}$ outputs a pair (M^*, σ^*).

$\mathcal{A}$ succeeds in the game if σ^* is a proper signature of M^* under the verification key vk, i.e., $\mathsf{Verify}_{vk}(M^*, \sigma^*) = \mathsf{accept}$ but $(M^*, \sigma^*) \neq (M, \sigma)$. $\mathcal{A}$ may succeed even if $M^* = M$.

Let $\mathcal{H}' = (\mathsf{Setup}, \mathsf{Der}', \mathcal{E}', \mathcal{D}')$ be an $(h+1)$-HIBE for arbitrary $h \geq 1$ handling $(n+1)$-bit identities. Let $\mathsf{Sig} = (\mathsf{Key\text{-}Gen}, \mathsf{Sig}, \mathsf{Verify})$ be a signature scheme which outputs an n-bit signature. If $\mathcal{H}'$ is secure in the sense IND-sID-CPA and Sig is a strong one-time signature scheme, then one can construct an h-HIBE secure in the sense IND-sID-CCA that handles n-bit identities. Given an identity tuple $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j) \in (\{0,1\}^n)^j$ of $\mathcal{H}$ we map it to an identity tuple of $\mathcal{H}'$ as

$$\mathsf{Encode}(\mathsf{v}) = (0\mathsf{v}_1, \ldots, 0\mathsf{v}_j) \in (\{0,1\}^{n+1})^j$$

and $\mathsf{Encode}(\varepsilon) = \varepsilon$, i.e the null string is mapped to itself. Let $\hat{\mathsf{v}} = \mathsf{Encode}(\mathsf{v})$, the HIBE $\mathcal{H}$ is constructed in such a way that the private key d_{v} of an identity tuple v in $\mathcal{H}$ is equal to the private key $d'_{\hat{\mathsf{v}}}$ of $\hat{\mathsf{v}}$ in $\mathcal{H}'$.

Construction of $\mathcal{H}$

Setup: Same as the Setup algorithm of $\mathcal{H}'$. The master key of $\mathcal{H}'$, $\mathsf{msk}_{\mathcal{H}'}$ is the master key, $\mathsf{msk}_{\mathcal{H}}$ of $\mathcal{H}$.

Key-Gen: Let d_{v} be the private key of v. To derive the private key of (v, r) find $\hat{\mathsf{v}} = \mathsf{Encode}(\mathsf{v})$ and $\hat{\mathsf{r}} = \mathsf{Encode}(\mathsf{r})$. Run $\mathsf{Der}'_{d_{\mathsf{v}}}(\hat{\mathsf{v}}, \hat{\mathsf{r}})$ and output the result as $d_{\mathsf{v},\mathsf{r}}$.

Note that, $d_{\mathsf{v},\mathsf{r}} = d_{\hat{\mathsf{v}},\hat{\mathsf{r}}}$, given $d_{\mathsf{v}} = d'_{\hat{\mathsf{v}}}$.

Encrypt: To encrypt a message M to an identity tuple v, run the key generation algorithm of Sig, $\mathsf{Key\text{-}Gen}(1^\kappa)$ to obtain (vk, sk). Let $\hat{\mathsf{v}} = \mathsf{Encode}(\mathsf{v})||(1vk)$, compute $C = \mathcal{E}'_{\mathsf{PP}}(\hat{\mathsf{v}}, M)$ and $\sigma = \mathsf{Sign}_{sk}(C)$. The ciphertext is the tuple $\langle vk, C, \sigma \rangle$.

Decrypt: Given the ciphertext $\langle vk, C, \sigma \rangle$, first check whether $\mathsf{Verify}_{vk}(C, \sigma) = \mathsf{accept}$. If not reject the ciphertext. Otherwise, let $\hat{\mathsf{v}} = \mathsf{Encode}(\mathsf{v})$ and run $\mathsf{Der}'_{d_{\mathsf{v}}}(\hat{\mathsf{v}}, (1vk))$ to generate the private key $d* = d'_{\hat{\mathsf{v}}||(1vk)}$. Then output $M = \mathcal{D}'_{d*}(\hat{\mathsf{v}}, C)$.

Security

Given an identity tuple $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$, $j \leq h$ in $\mathcal{H}$, the sender encrypts the message M to a $(j+1)$ level identity $\hat{\mathsf{v}} = (\mathsf{Encode}(\mathsf{v}), (1vk))$ of $\mathcal{H}'$ where vk is the verification key of the underlying signature scheme. The receiver having identity v first derives the private key of $\hat{\mathsf{v}}$ in $\mathcal{H}'$ from the private key d_{v} in $\mathcal{H}$ using the key generation algorithm of $\mathcal{H}'$. We assume that the probability of forging a signature, Pr[Forge] is negligible. Then by a reductionist security argument Boneh, Canetti, Halevi and Katz show that if there is an IND-sID-CCA adversary $\mathcal{A}$ against $\mathcal{H}$ in the *selective*-ID model then one can construct an IND-sID-CPA adversary $\mathcal{A}'$ against $\mathcal{H}'$ in the same model. Here we reproduce their argument:

1. $\mathcal{A}'$ runs the IND-sID-CCA adversary $\mathcal{A}$ which outputs a target identity tuple $\mathsf{v}^* = \langle \mathsf{v}_1^*, \ldots, \mathsf{v}_j^* \rangle$, $j \leq h$. $\mathcal{A}'$ next runs the key generation algorithm Key-Gen of Sig to generate (vk^*, sk^*). It outputs $\mathsf{Encode}(\mathsf{v}^*), (1vk^*)$ as its target identity.

2. The challenger gives $\mathcal{A}'$ the public parameter PP, which it relays to $\mathcal{A}$.

3. $\mathcal{A}$ asks for the private key of an identity v, which is not a prefix of v^*. $\mathcal{A}'$ asks its challenger for the private key $d_{\hat{\mathsf{v}}}$ where $\hat{\mathsf{v}} = \mathsf{Encode}(\mathsf{v})$ and returns it to $\mathcal{A}$.

4. For a decryption query of the form $(\mathsf{v}, \langle vk, C, \sigma \rangle)$ from $\mathcal{A}$, $\mathcal{A}'$ takes the following action:

   (a) If $\mathsf{v} = \mathsf{v}^*$ and $vk = vk^*$, return reject.

   (b) If $\mathsf{v} \neq \mathsf{v}^*$ or if $\mathsf{v} = \mathsf{v}^*$ but $vk \neq vk^*$, then $\mathcal{A}'$ sets $\hat{\mathsf{v}} = \mathsf{Encode}(\mathsf{v})$ and requests its challenger for the private key of $\hat{\mathsf{v}}, (1vk)$. It decrypts the ciphertext using this private key and returns the result to $\mathcal{A}$.

5. In the challenge stage $\mathcal{A}$ outputs two messages M_0, M_1. The same messages are also output by $\mathcal{A}'$. It receives a challenge ciphertext C^*. Now $\mathcal{A}'$ computes $\sigma^* = \mathsf{Sign}_{sk^*}(C^*)$ and returns the ciphertext $\langle vk^*, C^*, \sigma^* \rangle$ to $\mathcal{A}$.

6. In phase 2 $\mathcal{A}$ makes additional decryption queries and private key extraction queries. These queries are answered as before.

7. Finally $\mathcal{A}$ outputs its guess γ'. The same γ' is output by $\mathcal{A}'$.

In the above simulation, $\mathcal{A}'$ poses as a real challenger for $\mathcal{A}$. Since we have assumed that the probability of forging a signature is negligible, the advantage of $\mathcal{A}$ against $\mathcal{H}$ translates into the advantage of $\mathcal{A}'$ against $\mathcal{H}'$.

Note that, if $\mathcal{H}'$ is adaptive chosen plaintext secure in the full model (i.e., IND-ID-CPA secure), then $\mathcal{H}$ will be adaptive chosen ciphertext secure (i.e., IND-ID-CCA secure) in the full model.

Given this generic transformation to achieve CCA-security, protocol designers generally concentrate on constructing protocols that achieve CPA-security (be it in the full model or the selective-ID model) without random oracle and then apply this transformation to

achieve CCA-security. Protocols such as the Boneh-Boyen HIBE of Section 3.2.1 and the Boneh-Boyen-Goh HIBE of Section 3.4.1 accomplish this in the selective-ID model, while the Boneh-Boyen IBE and Waters IBE of Section 3.3 accomplish this in the full model.

Boyen, Mei and Waters proposed an endogenous transformation to achieve CCA security [23]. Their technique uses the structure of the underlying HIBE and there by avoid the use of one time signature or MAC. In their work they have suggested that when applied to BB-HIBE or BBG-HIBE this technique yields a selective-ID CCA-secure hierarchical identity-based key encapsulation mechanism (HIB-KEM) and for Waters protocol modified to HIBE, an adaptive identity CCA-secure HIBE.

In view of these developments, we concentrate on only designing CPA-secure protocols in this dissertation. Any of the transformations mentioned above can be applied on the CPA-secure protocols discussed in this work to achieve CCA-security.

3.6 Conclusion

In this chapter, we reviewed some of the important (hierarchical) identity-based encryption protocols proposed in the literature. Along with the protocols, in some cases we have discussed the salient features of the proof technique while an intuitive justification of the proof is given for some others. These protocols are proved secure against chosen plaintext attack. Methods for achieving CCA-security is also mentioned. Our aim was not an exhaustive survey of the area, but an exposition of the basic issues in identity-based encryption, developments in the proof techniques, the prospects as well as the problems. It also helps to present our work in the proper context. This being an emerging research area, new concepts are still evolving. The IBE proposed by Gentry in [48] , the concept of anonymous IBE [1] and the anonymous HIBE proposed by Boyen and Waters in [24] are some important recent developments not discussed here.

Chapter 4

Tate Pairing in General Characteristic Fields

4.1 Introduction

Implementation of pairing based protocols requires efficient algorithm for computing pairing. An initial breakthrough in this direction was made in [8] and [45], which introduced some nice optimisation ideas leading to dramatic improvement in pairing computation time. Since then, there have been quite a few works on implementation aspects. Most of the initial implementation works have focussed on Tate pairing as it is faster than Weil pairing. However, in [63] it has been observed that, for higher security levels Weil pairing can be faster than Tate pairing. This again was contested in [51] who argued that Tate pairing can indeed be faster.

In this chapter, we consider only elliptic curves over large prime fields having embedding degree 2. For such fields, we consider the use of Jacobian coordinates for Tate pairing computation. The new idea that we introduce is encapsulated double-and-line computation and encapsulate add-and-line computation. We also describe encapsulated version of iterated double and line computation.

From an implementation point of view, we divide curves of the form $y^2 = x^3 + ax + b$ into three cases: small a, $a = -3$ and the case where a is a general element of the field. In each case, we present detailed algorithm for pairing computation.

For hardware applications having special purpose crypto co-processors, it might be desirable to consider parallel versions of the algorithms. We identify the inherent parallelism in our algorithms and two-multiplier parallel version of our algorithms are optimal with respect to the number of parallel rounds.

In comparison with earlier work, we are able to obtain approximately 33% speed-up over the best known algorithm [57] for the case of general a. In the case $a = -3$ and for non-supersingular curves with embedding degree 2, the work by Scott [82] provides the most

efficient algorithm. In comparison, for the case $a = -3$, we are able to obtain approximately 20% speed-up over the algorithm in [82].

Related Work: The important work in [8] and [45] has been mentioned before. Projective coordinates were seriously considered by Izu and Takagi [57], where mainly non-supersingular curves with large embedding degrees were considered. Further, projective coordinates in conjunction with non-supersingular curves with embedding degree 2 were also considered in the work by Scott [82] mentioned earlier. The work by Scott and Baretto [83] considers the issue of computing trace of pairings. This paper also describes a laddering algorithm for exponentiation in $\mathbb{F}_{p^2}$ based on Lucas sequences. For general characteristics, this exponentiation algorithm is the fastest known and has to be used with the algorithm that we develop. The algorithm proposed by Eisenträerger et. al. [40] uses the double-add trick with parabolas for fast computation of pairing in affine coordinates. There are several other works on Tate pairing computation like [39, 52, 66]. However, most of these work with affine coordinates and over characteristic two or three. Hence, they are not much relevant in the present context and therefore are not discussed here.

Implementation of pairing being an active and emerging research area, there are important advances in different aspects. These include construction of elliptic curves suitable for pairing implementation [10, 12], efficient algorithms for curves with larger embedding degree [9] and also efficient implementation of pairing based protocols [11].

4.2 Preliminaries

We start with a definition of modified Tate pairing. We limit the discussion at a level necessary to understand the implementation issues discussed in this chapter. For a detailed description of Tate pairing the interested reader is referred to Chapter IX of [15].

4.2.1 The Tate Pairing

Let $\mathbb{F}_p$ (with p prime) be a finite field of integers modulo p and $\mathbb{F}_{p^k}$ be the degree k extension of $\mathbb{F}_p$. We define an elliptic curve E over $\mathbb{F}_p$ by the equation [53]

$$y^2 = x^3 + ax + b \tag{4.2.1}$$

where $a, b \in \mathbb{F}_p$ satisfy the condition $4a^3 + 27b^2 \not\equiv 0 \bmod p$. A pair (x, y) with $x, y \in \mathbb{F}_p$ is said to be a point on the elliptic curve if (x, y) satisfies Equation 4.2.1. The points on the elliptic curve together with a special point at infinity (denoted by $\mathcal{O}$) forms an additive abelian group.

Let r be a large prime divisor of $(p + 1)$, such that r is coprime to p and for some $k > 0$, $r|p^k - 1$ but $r \not| p^s - 1$ for any $1 \leq s < k$; k is called the embedding degree (MOV degree). Suppose P is a point of order r on the elliptic curve $E(\mathbb{F}_p)$ and Q is a point of same

order on the elliptic curve $E(\mathbb{F}_{p^k})$, linearly independent of P. We denote the (modified) Tate pairing of order r as $e_r(P,Q) \in \mathbb{F}_{p^k}$.

$e_r(P,Q)$ is defined in terms of divisors of a rational function. A divisor is a formal sum: $D = \sum_{P \in E} a_P \langle P \rangle$, where $P \in E(\mathbb{F}_p)$. The degree of a divisor D is $deg(D) = \sum_{P \in E} a_P$. The set of divisors forms an abelian group by the addition of their coefficients in the formal sum. Let f be a (rational) function on E, then the divisor of f, $\langle f \rangle = \sum_P ord_P(f)\langle P \rangle$, where $ord_P(f)$ is the order of the zero or pole of f at P. A divisor $D = \sum_{P \in E} a_P \langle P \rangle$ is called a principal divisor if and only if it is a divisor of degree 0 (zero divisor) and $\sum_{P \in E} a_P P = \mathcal{O}$. If D is principal then there is some function f such that $D = \langle f \rangle$. Two divisors D_1 and D_2 are said to be equivalent if $D_1 - D_2$ is a principal divisor. Let $\mathcal{A}_\mathcal{P}$ be a divisor equivalent to $\langle P \rangle - \langle \mathcal{O} \rangle$ (similarly $\mathcal{A}_\mathcal{Q}$). Then it is easy to see that $r\mathcal{A}_\mathcal{P}$ is principal; thus there is a rational function f_P with $\langle f_P \rangle = r\mathcal{A}_\mathcal{P} = r\langle P \rangle - r\langle \mathcal{O} \rangle$. The (modified) Tate pairing of order r is defined as

$$e_r(P,Q) = f_P(\mathcal{A}_\mathcal{Q})^{(p^k-1)/r} \tag{4.2.2}$$

4.2.2 Miller's Algorithm

Miller proposed a polynomial time algorithm to compute the Weil pairing [74, 75]. This algorithm can be adapted easily to compute the modified Tate pairing. Let f_a be a (rational) function with divisor $\langle f_a \rangle = a\langle P \rangle - \langle aP \rangle - (a-1)\langle \mathcal{O} \rangle$, $a \in Z$. It can be shown that $f_{2a}(Q) = f_a(Q)^2 . h_{aP,aP}(Q)/h_{2aP}(Q)$ where, $h_{aP,aP}$ is the line tangent to $E(\mathbb{F}_p)$ at aP, it intersects $E(\mathbb{F}_p)$ at the point $-2aP$, and h_{2aP} is the (vertical) line that intersects $E(\mathbb{F}_p)$ at $2aP$ and $-2aP$. Now, $\langle f_r \rangle = r\langle P \rangle - \langle rP \rangle - (r-1)\langle \mathcal{O} \rangle = \langle f_P \rangle$, since $rP = \mathcal{O}$. Given P and the binary representation of r, Miller's algorithm computes $f_P(Q) = f_r(Q)$ in $\lg r$ steps by the standard double-and-add through line-and-tangent method for elliptic curve scalar multiplication. Under the condition, $r \not| (p-1)$ we can further have $e_r(P,Q) = f_P(Q)^{(p^k-1)/r}$ for $Q \neq \mathcal{O}$, as long as $k > 1$.

In the implementation of Tate pairing over super-singular elliptic curves $E(\mathbb{F}_p)$, the usual practice is to take $Q \in E(\mathbb{F}_p)$ of order r and then use a distortion map $\phi()$ [88], to get a point $\phi(Q) \in E(\mathbb{F}_{p^k})$ of order r which is linearly independent of P. A major finding in [8] is that, for particular choices of the curve parameters and distortion map $\phi()$ we can freely multiply or divide the intermediate result of pairing computation by any $\mathbb{F}_p$ element and consequently completely ignore the denominator part in the computation of Tate pairing.

4.2.3 Choice of Curves

Let E_1 be the elliptic curve given by the equation

$$y^2 = x^3 + ax \tag{4.2.3}$$

over $\mathbb{F}_p$. E_1 is super-singular if $p \equiv 3 \bmod 4$. For these curves, the curve order is $\#E_1(\mathbb{F}_p) = p + 1$ and embedding degree is $k = 2$. For such curves, a distortion map [8] is $\phi(x,y) =$

$(-x, iy) \in \mathbb{F}_{p^2} \times \mathbb{F}_{p^2}$ with $i^2 = -1$. Let r be the integer, for which we wish to compute $e_r(,)$. Then $r|(p+1)$ and the final powering in Tate pairing computation is of the form $(p^2-1)/r$. As observed in [8], this implies $(p-1)$ divides the final powering exponent.

Let, E_2 be the elliptic curve given by the equation

$$y^2 = x^3 - 3x + B \tag{4.2.4}$$

over $\mathbb{F}_p$, $p \equiv 3 \bmod 4$. Scott in his paper [82] considered this form of non super-singular EC with embedding degree, $k = 2$ with $\#E_2(\mathbb{F}_p) = p+1-t$, where t is the trace of Frobenius [69]. It is shown in [82] that the r for which $e_r(,)$ is computed over E_2 also satisfies $r|(p+1)$ and hence again $p-1$ divides the final powering exponent of Tate pairing.

Thus for both E_1 and E_2 the following observation holds. Since $x^{p-1} = 1$ for any $x \in \mathbb{F}_p$, this implies that we can freely multiply or divide any intermediate Tate pairing result by any nonzero element of $\mathbb{F}_p$. This has been previously used to improve the efficiency of Tate pairing computation in affine coordinates. In Section 4.3, we point out the importance of this observation in the context of projective coordinates.

Note that, for E_1 as well as E_2, the embedding degree is $k = 2$ and the elements of the quadratic extension field ($\mathbb{F}_{p^2}$) is represented as $a + ib$, where $a, b \in \mathbb{F}_p$ and i is the *square root* of a quadratic non-residue. For $p \equiv 3 \bmod 4$ we can further have $i^2 = -1$. Essentially, the same algorithm that we develop for E_1 also applies to E_2 by evaluating $e(,)$ at P and $(-x_Q, iy_Q)$.

4.2.4 Non-Adjacent Form Representation

The Non-Adjacent Form (NAF) representation of an integer has been suggested for elliptic curve scalar multiplication. In this representation, the digits $\{0, \pm 1\}$ are used to represent an integer with the property that no two adjacent digits are non-zero. The advantage is that, on an average, the number of non-zero digits is one-third of the length of the representation, while it is one-half in the case of binary representation. For details of NAF representation we refer the reader to [53].

For Tate pairing applications, the representation of r should be sparse, i.e., the number of non-zero digits should be small. The NAF representation is sparser than the corresponding binary representation. Hence in our algorithms, we work entirely with the NAF representation.

4.3 Encapsulated Computation

In the computation of Tate pairing one needs to perform an implicit scalar multiplication of the EC point P. For this, as well as for computation of the line function $h_{aP,aP}()$ one requires to perform base field inversion. But inversion for large characteristic is quite costly. The standard method to avoid inversion is to move from affine to projective coordinate system. Among the different available projective coordinate systems, the Jacobian gives

the best result. In [57] the authors suggested to take the so called *simplified Jacobian-Chudnovsky coordinate* J^s as they store (X, Y, Z, Z^2) instead of (X, Y, Z). However, we have found out that if one encapsulates EC addition/doubling with line computation then there is no need to additionally store Z^2 – one can simply work in the Jacobian coordinate. Here we give the explicit formulae required for the encapsulated computation of double/add-and-line computation. In what follows, by [M] and [S], we respectively denote the cost of one multiplication and one squaring in $\mathbb{F}_p$.

4.3.1 Encapsulated Point Doubling and Line Computation

Here $P = (X_1, Y_1, Z_1)$ correspond to $(X_1/Z_1^2, Y_1/Z_1^3)$ in affine coordinate. We encapsulate the computation of $2P$ given P together with the computation corresponding to the associated line.

Point Doubling : From the EC point doubling rule we have the following formula:

$$\begin{aligned} X_3' &= \frac{(3X_1^2 + aZ_1^4)^2 - 8X_1Y_1^2}{4Y_1^2Z_1^2} \\ Y_3' &= \frac{3X_1^2 + aZ_1^4}{2Y_1Z_1}(\frac{X_1}{Z_1^2} - X_3') - \frac{Y_1}{Z_1^3} \\ X_3 &= (3X_1^2 + aZ_1^4)^2 - 8X_1Y_1^2 \\ Y_3 &= (3X_1^2 + aZ_1^4)(4X_1Y_1^2 - X_3) - 8Y_1^4 \\ Z_3 &= 2Y_1Z_1 \end{aligned}$$

Using temporary variables, we compute:

1. $t_1 = Y_1^2$;
2. $t_2 = 4X_1t_1$;
3. $t_3 = 8t_1^2$;
4. $t_4 = Z_1^2$;
5. $t_5 = 3X_1^2 + aZ_1^4$;
6. $X_3 = t_5^2 - 2t_2$;
7. $Y_3 = t_5(t_2 - X_3) - t_3$;
8. $Z_3 = 2Y_1Z_1$.

So, we require 6[S] + 4[M] for EC doubling. Now consider t_5. If a is a general element of $\mathbb{F}_p$, then we have to count the multiplication $a \times (Z_1^4)$. However, if a is small, i.e., it can be represented using only a few (say ≤ 8) bits, then we do not count this multiplication. In this case, aZ_1^4 can be obtained summing Z_1^4 a total of a times. This reduces the operation count to 6[S]+3[M]. Further, if $a = -3$, then $t_5 = 3(X_1 - Z_1^2)(X_1 + Z_1^2) = 3(X_1 - t_4)(X_1 + t_4)$ and the operation count reduces to 4[S]+4[M]. These facts are known and can be found in [53].

Line Computation: Note that, the slope μ of $h_{P,P}$, the line through P and $-2P$, is $\mu = t_5/Z_3$. So,

$$h_{P,P}(x, y) = (y - \frac{Y_1}{Z_1^3}) - \mu(x - \frac{X_1}{Z_1^2}).$$

Hence, $h_{P,P}(-x_Q, iy_Q) = (y_Q i - \frac{Y_1}{Z_1^3}) + \mu(x_Q + \frac{X_1}{Z_1^2})$.
By defining $g_{P,P}(x, y) = (2Y_1Z_1^3)h_{P,P}(x, y)$, we get,

$$\begin{aligned} g_{P,P}(-x_Q, iy_Q) &= (2Y_1Z_1)Z_1^2y_Qi - 2Y_1^2 + (3X_1^2 + aZ_1^4)(Z_1^2x_Q + X_1) \\ &= Z_3t_4y_Qi - (2t_1 - t_5(t_4x_Q + X_1)) \end{aligned}$$

The ultimate result is raised to the power $(p^2-1)/r$, where $r|(p+1)$ (see Section 4.2.1). Thus we have to compute

$$\begin{aligned}(h_{P,P}(-x_Q, iy_Q))^{(p^2-1)/r} &= ((h_{P,P}(-x_Q, iy_Q))^{(p-1)})^{(p+1)/r} \\ &= ((g_{P,P}(-x_Q, iy_Q)/(2Y_1Z_1^3))^{(p-1)})^{(p+1)/r} \\ &= ((g_{P,P}(-x_Q, iy_Q))^{(p-1)})^{(p+1)/r}.\end{aligned}$$

The last equation holds since $(2Y_1Z_1^3) \in \mathbb{F}_p$ and consequently $(2Y_1Z_1^3)^{p-1} = 1$. Thus, we can work entirely with $g_{P,P}(-x_Q, iy_Q)$ instead of $h_{P,P}(-x_Q, iy_Q)$. Since t_1, t_4 and t_5 have already been computed, $g_{P,P}(-x_Q, iy_Q)$ can be obtained using 4 additional multiplications.

Hence, *encapsulated point doubling and line computation* requires 8[M] + 6[S]. In the case a is small, this cost is 7[M]+6[S] and for the case $a = -3$, this cost is 8[M]+4[S].

4.3.2 Encapsulated (Mixed) Point Addition and Line Computation

We encapsulate the computation of $P+R$ given P in affine and R in Jacobian together with the computation corresponding to the associated line.

Mixed Addition: Given $R = (X_1, Y_1, Z_1)$ and $P = (X, Y, 1)$ we compute $R + P = (X_3, Y_3, Z_3)$ as follows.

$$\begin{aligned}X_3' &= \left(\frac{Y - \frac{Y_1}{Z_1^3}}{X - \frac{X_1}{Z_1^2}}\right)^2 - \frac{X_1}{Z_1^2} - X \\ &= \left(\frac{YZ_1^3 - Y_1}{(XZ_1^2 - X_1)Z_1}\right)^2 - \frac{X_1}{Z_1^2} - X \\ Y_3' &= \left(\frac{YZ_1^3 - Y_1}{(XZ_1^2 - X_1)Z_1}\right)(\frac{X_1}{Z_1^2} - X_3') - \frac{Y_1}{Z_1^3} \\ X_3 &= X_3'Z_3 \\ &= (YZ_1^3 - Y_1)^2 - X_1(XZ_1^2 - X_1)^2 - X(XZ_1^2 - X_1)^2Z_1^2 \\ &= (YZ_1^3 - Y_1)^2 - (XZ_1^2 - X_1)^2(X_1 + XZ_1^2) \\ Y_3 &= Y_3'Z_3 \\ &= (YZ_1^3 - Y_1)((XZ_1^2 - X_1)^2X_1 - X_3) - Y_1(XZ_1^2 - X_1)^3 \\ Z_3 &= (XZ_1^2 - X_1)Z_1\end{aligned}$$

Using temporary variables we compute:

1. $t_1 = Z_1^2$;
2. $t_2 = Z_1t_1$;
3. $t_3 = Xt_1$;
4. $t_4 = Yt_2$;
5. $t_5 = t_3 - X_1$;
6. $t_6 = t_4 - Y_1$;
7. $t_7 = t_5^2$;
8. $t_8 = t_5t_7$;
9. $t_9 = X_1t_7$;
10. $X_3 = t_6^2 - (t_8 + 2t_9)$;
11. $Y_3 = t_6(t_9 - X_3) - Y_1t_8$;
12. $Z_3 = Z_1t_5$.

Hence, we require 3[S] + 8[M] for EC point addition. See [53] for details.

Line Computation: Note that, the slope μ of $h_{R,P}$, the line through R and P is $\mu = t_6/Z_3$. So,

$$h_{R,P}(x, y) = (y - Y) - \mu(x - X).$$

Hence, $h_{R,P}(-x_Q, iy_Q) = (y_Q i - Y) + \mu(x_Q + X)$. Define $g(x, y)$ as $g(x, y) = Z_3 h_{R,P}(x, y)$. Thus, we get

$$g_{R,P}(-x_Q, iy_Q) = Z_3 y_Q i - (Z_3 Y - t_6(x_Q + X))$$

As explained in the case of doubling, we can simply work with $g_{R,P}$ instead of $h_{R,P}$. Since we have already computed t_6 and Z_3 during point addition, $g_{R,P}(-x_Q, iy_Q)$ can be computed using additional three multiplications. Hence, *encapsulated point addition and line computation* requires 11[M] + 3[S].

4.4 Algorithm

We consider three situations: $a = -3$; a small (i.e., multiplication by a need not be counted); and the case where a is a general element of $\mathbb{F}_p$. For the first two cases, double-and-add algorithm is considered. For the general case, we adopt an iterated doubling technique used by Izu and Takagi [57].

4.4.1 Double-and-Add

We slightly modify the Miller's algorithm as improved in [8]. We will call this algorithm the modified BKLS algorithm. In the algorithm the NAF representation of r is taken to be $r_t = 1, r_{t-1}, \ldots, r_0$.

Algorithm 1 (Modified BKLS Algorithm):
1. set $f = 1$ and $V = P$
2. for $i = t - 1$ downto 0 do
3. $\quad (u, V) = \mathsf{EncDL}(V)$;
4. $\quad$ set $f = f^2 \times u$;
5. $\quad$ if $r_i \neq 0$, then
6. $\quad\quad (u, V) = \mathsf{EncAL}(V, r_i)$;
7. $\quad\quad$ set $f = f \times u$;
8. $\quad$ end if;
9. end for;
10. return f;

end Algorithm 1.

The subroutine $\mathsf{EncDL}(V)$ performs the computation of Section 4.3.1 and returns $(g_{V,V}(\phi(Q)), 2V)$. The subroutine $\mathsf{EncAL}(V, r_i)$ takes V and r_i as input. If $r_i = 1$, it performs the computation of Section 4.3.2 and returns $(g_{V,P}(\phi(Q)), V + P)$; if $r_i = -1$, it first negates the point

$P = (\alpha, \beta)$ to obtain $P' = -P = (\alpha, -\beta)$, then it performs the computation of Section 4.3.2 with P' instead of P and returns $(g_{V,-P}(\phi(Q)), V - P)$. The correctness of the algorithm follows easily from the correctness of the original BKLS algorithm.

We consider the cost. The subroutine EncDL is invoked a total of t times while EncAL is invoked a total of s times where s is the Hamming weight of $r_{t-1} \dots r_0$. The cost of updation in Line 4 is one $\mathbb{F}_{p^2}$ squaring and one $\mathbb{F}_{p^2}$ multiplication. These operations can be completed in five $\mathbb{F}_p$ multiplications (see [83]). The cost of updation in Line 7 is three $\mathbb{F}_p$ multiplications.

The cost of EncDL depends upon the value of the curve parameter a. We analyse the total cost for the following two cases.
Case $a = -3$:

- Cost of EncDL is 8[M]+4[S].
- Cost of update in line 4 is 5[M].
- Cost of EncAL is 11[M]+3[S].
- Cost of update in line 7 is 3[M].
- Total cost is t(13[M]+4[S]) + s(14[M]+3[S]).

Case a is small:

- Cost of EncDL is 7[M]+6[S].
- Cost of update in line 4 is 5[M].
- Cost of EncAL is 11[M]+3[S].
- Cost of update in line 7 is 3[M].
- Total cost is t(12[M]+6[S]) + s(14[M]+3[S]).

4.4.2 Iterated Doubling

In the case where we have to consider the multiplication by the curve parameter a, we employ the technique of iterated doubling to reduce the total number of operations. As before we consider the NAF representation of r. We write the NAF representation of r as

$$(r_t = 1, r_{t-1}, \dots, r_0) = (l_s = 1, 0^{w_{s-1}}, l_{s-1}, \dots, 0^{w_0}, l_0)$$

where the l_i's are ± 1. The following algorithm is an iterated doubling version of the modified BKLS algorithm described in Section 4.4.1. The points $P = (\alpha, \beta)$ and $Q = (x_Q, y_Q)$ are globally available.

Algorithm 2 (iterated doubling):
Input: $P = (\alpha, \beta, 1)$ in Jacobian coordinates; $Q = (x_Q, y_Q)$.
Output: $f_P(\phi(Q))$.
1. Set $f = 1$; $g = 1$;
2. $X = \alpha$; $Y = \beta$; $Z = 1$; set $R = (X, Y, Z)$;
3. for $j = s - 1$ down to 0
4. $\quad (f, R) = \mathsf{EncIdbl}(f, R, w_j)$;
5. $\quad (g, R) = \mathsf{EncAL}(R, l_j)$;
6. $\quad f = f \times g$;
7. end for;
8. return f;
end Algorithm 2.

The Subroutine EncAL has already been discussed in Section 4.4.1. We now describe Subroutine EncIdbl.

Subroutine EncIdbl
Input: $R = (X, Y, Z)$, f and w.
Output: updated f and $2^{w+1}R$.
1. $t_1 = Y^2$; $t_2 = 4Xt_1$; $t_3 = 8t_1^2$; $t_4 = Z^2$; $w = aZ^4$; $t_5 = 3X^2 + w$;
2. $A = -(2t_1 + t_5(t_4 x_Q - X))$; $X = t_5^2 - 2t_2$;
   $Y = t_5(t_2 - X) - t_3$; $Z = 2YZ$; $B = Zt_4 y_Q$;
3. $f = f^2 \times (A + iB)$;
4. for $j = 1$ to w do
5. $\quad w = 2t_3 w$; $t_1 = Y^2$; $t_2 = 4Xt_1$; $t_3 = 8t_1^2$; $t_4 = Z^2$; $t_5 = 3X^2 + w$;
6. $\quad A = -(2t_1 + t_5(t_4 x_Q - X))$; $X = t_5^2 - 2t_2$;
   $\quad Y = t_5(t_2 - X) - t_3$; $Z = 2YZ$; $B = Zt_4 y_Q$;
7. $\quad f = f^2 \times (A + iB)$;
8. end for;
9. $R = (X, Y, Z)$;
9. return (f, R);
end Subroutine EncIdbl.

Algorithm 2 is essentially the same as Algorithm 1 except for the use of iterated doubling. The technique of iterated doubling is considered to reduce computation cost but does not affect the correctness of the algorithm. We consider the cost of the algorithm. As before let the Hamming weight of $r_{t-1}, \ldots, r_0$ be s.

- Steps 5 and 6 of Algorithm 2 are invoked s times. The total cost of these two steps is s(14[M]+3[S]).
- Step 4 of Algorithm 2 is invoked a total of s times. The cost of the jth invocation of Step 4 is computed as follows:
  - Cost of Steps 3 and 7 in EncIdbl is 5[M].

- Cost of Steps 1 and 2 in EncIdbl is 8[M]+6[S].
- Cost of Steps 5 and 6 in EncIdbl is 8[M]+5[S].
- Total cost of jth invocation of EncIdbl is $13[M]+6[S]+w_j(13[M]+5[S])=1[S]+(w_j+1)(13[M]+5[S])$.

- Total cost of Algorithm 2 is $s(14[M]+3[S])+\sum_{j=0}^{s-1}(1[S]+(w_j+1)(13[M]+5[S]))$ $=s(14[M]+4[S])+t(13[M]+5[S])$.

4.4.3 Memory Requirement

The memory requirement of Algorithm 1 and Algorithm 2 are similar with Algorithm 2 requiring slightly more memory. We consider the memory requirement of Algorithm 2. To find the minimum memory requirement, first note that in Algorithm 2 we have to store and update $f \in \mathbb{F}_{p^2}$ and $X, Y, Z \in \mathbb{F}_p$ – they require 5 $\mathbb{F}_p$ storage space. We also need to store $Q = (x_Q, y_Q)$. In addition, we require some temporary variables to keep the intermediate results produced in the subroutines EncIdbl and EncAL. These subroutines are called one after another – we first call EncIdbl and update f together with X, Y, Z, release the temporary variables and then call EncAL where these temporary variables can be reused. The maximum of the number of temporary variables required by the two subroutines determines the number of temporary variables required in Algorithm 2. We ran a computer program to separately find these requirements. Given a straight line code what the program essentially does is to exhaustively search (with some optimisations) for all possible execution paths and output the path pertaining to minimum number of temporary variables. This turns out to be 9 for EncIdbl, while it is 7 for EncAL. So, at most we require to store 16 $\mathbb{F}_p$ elements.

4.4.4 Parallelism

We consider the parallelism in the encapsulated computations of Sections 4.3.1 and 4.3.2. While considering parallelism, we assume that a multiplier is used to perform squaring.

First we consider the case of encapsulated double and line computation. The situation in Section 4.3.1 has three cases – small a, $a = -3$ and general a. For the last case we use the iterated doubling technique of Section 4.4.2. We separately describe parallelism for the three cases. In each case, we first need to identify the multiplications which can be performed together. This then easily leads to parallel algorithms with a fixed number of multipliers.

Small a

In this case, multiplication by a will be performed as additions. The multiplication levels are as follows.

Level 1 : $t_1 = Y_1^2$; $t_4 = Z_1^2$; X_1^2; $Z_3 = 2Y_1Z_1$; square f;
Level 2 : $t_2 = 4X_1t_1$; $t_3 = 8t_1^2$; $t_5 = 3X_1^2 + aZ_1^4$; $t_6 = t_4x_Q$; $t_7 = t_4y_Q$;
Level 3 : $-(2t_1 + t_5(t_6 - X_1))$; $X_3 = t_5^2 - 2t_2$; $Y_3 = t_5(t_2 - X_3) - t_3$; Z_3t_7;
Level 4 : update f.

Case $a = -3$

In this case, $t_5 = 3(X_1^2 - Z_1^4) = 3(X_1 - Z_1^2)(X_1 + Z_1^2)$. The multiplication levels are as follows.

Level 1 : $t_1 = Y_1^2$; $t_4 = Z_1^2$; $Z_3 2Y_1Z_1$; square f;
Level 2 : $t_2 = 4X_1t_1$; $t_3 = 8t_1^2$; $t_5 = 3(X_1 - t_4)(X_1 + t_4)$; $t_6 = t_4x_Q$; $t_7 = t_4y_Q$;
Level 3 : $-(2t_1 + t_5(t_6 - X_1))$; $X_3 = t_5^2 - 2t_2$; $Y_3 = t_5(t_2 - X_3) - t_3$; Z_3t_7;
Level 4 : update f.

General a

In this case, Subroutine EncIdbl is used. This consists of an initial part plus computation inside the *for* loop. The parallel version of both these parts are similar and we describe the parallel version of the loop computation. The multiplication levels are as follows.

Level 1 : $w = 2t_3w$; $t_1 = Y^2$; $t_4 = Z^2$; X^2; $Z_3 = 2YZ$; square f;
Level 2 : $t_2 = 4Xt_1$; $t_3 = 8t_1^2$; $t_5 = 3X^2 + w$; $t_6 = t_4x_Q$; $t_7 = t_4y_Q$;
Level 3 : $A = -(2t_1 + t_5(t_6 - X))$; $X = t_5^2 - 2t_2$; $Y = t_5(t_2 - X) - t_3$; Z_3t_7;
Level 4 : update f.

In each of the above cases, with two multipliers the entire operation can be performed in 9 rounds and with four multipliers it can be performed in 5 rounds. Since the total number of operations is either 17 or 18 squarings and multiplications, the number of rounds is optimal for the given number of operations and given number of multipliers.

Addition

We now consider the case of encapsulated add-and-line computation. See Section 4.3.2 for the details of the temporary variables and the operations. Here we mainly list the multiplication/squaring operations.

Level 1 : $t_1 = Z_1^2$;
Level 2 : $t_2 = Z_1t_1$; $t_3 = Xt_1$;
Level 3 : $t_4 = Yt_2$; $t_7 = t_5^2$; $Z_3 = Z_1t_5$;
Level 4 : $t_8 = t_5t_7$; $t_9 = X_1t_7$; $X_3 = t_6^2 - (t_8 + 2t_9)$; Z_3y_Q; Z_3Y; $t_6(x_Q - X)$;
Level 5 : $Y_3 = t_6(t_9 - X_3) - Y_1t_8$; update f;

There are a total of 17 multiplications including the update operation. Using two multipliers, these can be performed in 9 rounds. On the other hand, the four multiplier algorithm is sub-optimal in the number of rounds.

Thus, for parallel version of pairing computation algorithm, one obtains optimal two-multiplier algorithms for both doubling and addition. For doubling, the four multiplier algorithm is optimal, while for addition, the four multiplier algorithm is sub-optimal. However, the Hamming weight of r will be small and hence if we use four multipliers then the sub-optimal performance will be amortized over the length of the representation of r and will not be significantly reflected in the final cost analysis.

4.5 Comparison

For the purpose of comparison, we assume that $r = (r_t = 1, r_{t-1}, \ldots, r_0)$ is represented in NAF having length t and Hamming weight s.

The irrelevant denominator optimisation was introduced in [8]. Further, [8] uses affine representation. The total cost including point/line computation and updation is

$$t(1[\mathrm{I}]+8[\mathrm{M}]+2[\mathrm{S}]) + s(1[\mathrm{I}]+6[\mathrm{M}]+1[\mathrm{S}]),$$

where [I] is the cost of inversion over $\mathbb{F}_p$ and is at least 30[M], see [72].

Izu-Takagi [57] uses projective coordinates for pairing computation in general characteristics for large embedding degree k. They also consider the BKLS optimisations for supersingular curves with embedding degree $k = 2$ for general a. They assume that one $\mathbb{F}_{p^k}$ multiplication takes k^2[M]. For $k = 2$, this can be improved to 3[M]. In the following calculation, we use this fact. Their cost for w-iterated doubling is $6w$[M]+$4w$[S]+$13w$[M]+$(5w+1)$[S] and addition is 6[M]+16[M]+3[S]. Summing over w's, the total cost comes to $t(19[\mathrm{M}]+9[\mathrm{S}])+s(22[\mathrm{M}]+4[\mathrm{S}])$.

The work of Scott [82] also proposes the use of projective coordinates in the case $a = -3$ for certain non-supersingular curves. The paper does not distinguish between multiplication and squaring. The total cost is $21t$[M]+$22s$[M].

In Table 4.1, we summarize the above costs along with the costs obtained by our algorithms for the various cases for the curve parameter a. The best case occurs for Algorithm 1 with $a = -3$. Also the cases for Algorithm 1 for small a and Algorithm 2 are marginally slower than the best case. However, all three of these cases are much more efficient than any of the previous algorithms. The algorithms of Izu-Takagi [57] and Scott [82] are more efficient than the basic BKLS algorithm with affine coordinates.

For Tate pairing applications, r is generally chosen so that the Hamming weight s is small. On the other hand, for a general r, the Hamming weight s is approximately $s = t/3$. In either of these two situations, we summarize the superiority of our method as follows.

- Algorithm 1 with $a = -3$ is approximately 20% faster compared to the algorithm by Scott.
- Algorithm 2 is approximately 33% faster compared to the algorithm by Izu and Takagi.

Table 4.1: Cost Comparison. Note 1[I]$\geq$ 30[M] [72].

Method	Cost
BKLS [8] (affine)	t(1[I]+8[M]+2[S])+s(1[I]+6[M]+1[S])
Izu-Takagi [57] (general a)	t(19[M]+9[S])+s(22[M]+4[S])
Scott [82] ($a = -3$)	21t[M]+22s[M]
Algorithm 1 (a small)	t(12[M]+6[S])+s(14[M]+3[S])
Algorithm 1 ($a = -3$)	t(13[M]+4[S])+s(14[M]+3[S])
Algorithm 2 (general a)	t(13[M]+5[S])+s(14[M]+4[S])

We consider the cost comparison to EC scalar multiplication. For the purpose of security, scalar multiplication has to be resistant to side channel attacks. One simple method of attaining resistance to simple power analysis is to use Coron's dummy addition using binary representation of multiplier. Under the (realistic) assumption that the length of the binary representation of the multiplier is equal to the length of the NAF representation of r for Tate pairing, the cost of dummy-addition countermeasure is t(2[M]+7[S]) for the case of $a = -3$. This cost is comparable to the cost of Algorithm 1 for $a = -3$ when s is at most around $t/8$. Again for practical situation r can usually be chosen so that $s \leq t/8$. Thus the efficiency of our algorithm is almost comparable to the efficiency of simple SPA resistant EC scalar multiplication. On the other hand, there is a wide variety of techniques for EC scalar multiplication providing very efficient algorithms. Whether the cost of Tate pairing computation can be made comparable to the most efficient EC scalar multiplication is currently a challenging research problem.

4.6 Conclusion

In this chapter, we have considered the use of Jacobian coordinates for Tate pairing computation in general characteristics with embedding degree two. The main idea that we have introduced is encapsulated double-and-line computation and encapsulated add-and-line computation. We have also developed encapsulated version of iterated double algorithm. The algorithms are presented in details and memory requirement has been considered. Inherent parallelism in these algorithms have been identified leading to optimal two-multiplier parallel algorithms. Our algorithms lead to an improvement of around 33% over previously best known algorithm for the general case where the curve parameter a is an arbitrary field element. In the special case where $a = -3$, our techniques provide an efficiency improvement of around 20% over the previously best known algorithm.

Chapter 5

Identity-Based Encryption in Full Model

5.1 Introduction

In this chapter, we provide a generalisation of the identity-based encryption scheme of Waters [89]. As already noted in Chapter 3, one disadvantage of Waters' scheme in [89] is the requirement of a rather large public parameter file. If identities are represented by a bit string of length n, then the scheme requires an n length vector of elements of G_1 to be maintained as part of the public parameter.

Our generalisation shows that if one tries to reduce the number of elements in the public parameter then there is a corresponding degradation in the security reduction. In other words, a trade-off is involved in the tightness of security reduction and smallness of public parameter. The trade-off between tightness and smallness can be converted to a trade-off between group size and smallness of public parameter. When desiring a specific security level, the loss of security due to loss of tightness in the security reduction can be compensated by working in a larger group. This increases the bit length of representation of the elements in the public parameter but the number of elements in the public parameters decreases so drastically that there is a significant reduction in the overall size of the public parameter. The increase in group size in turn affects the efficiency of the protocol. Thus, the trade-off is actually between the space required to store the parameters and the time required to execute the protocol. For example, if identities are represented by 160-bit strings, then Waters protocol require to store 160 elements of G_1 as part of the public parameter. Alternatively, using our generalisation if one wants to store 16 elements, then to achieve 80-bit security, compared to Waters protocol the space requirement reduces by around 90% while the computation cost increases by around 30%.

– Like Waters, applying Naor's technique, our scheme can also be easily converted to a signature scheme where the underlying security assumption is the computational Diffie-Hellman

problem.
– Our construction resembles closely the construction of Waters [89] and security against the chosen ciphertext attack (i.e., the CCA security) of the former follows from that of the later by constructing a 2-level HIBE and applying the technique of [27]. As an alternative, we show that CCA security can also be achieved by assuming the hardness of the oracle bilinear decision Diffie-Hellman assumption (OBDH).

5.2 Generalisation of Waters' IBE

Here we describe our generalisation of Waters scheme. The groups $G_1 = \langle P \rangle$, G_2 and the map $e()$ are as already defined in Section 2.1. In the following, we assume the message space $\mathcal{M}$ is G_2, the cipher space $\mathcal{C}$ is $G_2 \times G_1 \times G_1$.

Note that, in Waters scheme identities are represented as n-bit strings. Because of this representation, Waters requires to store n elements of G_1 i.e., $\overrightarrow{U}$ in the public parameter. Depending upon the choice of representation of the identities we can change the size of the public parameter.

Let $N = 2^n$, then we can consider the identities as elements of Z_N and one extreme case would be to consider the identities just as elements of Z_N. A more moderate approach, however, is to fix *a-priori* a size parameter ℓ, where $1 < \ell \leq n$. In this case, an identity v is represented as $\mathsf{v} = (\mathsf{v}_1, \mathsf{v}_2, \ldots, \mathsf{v}_\ell)$, where $\mathsf{v}_i \in Z_{N^{1/\ell}}$ i.e., each v_i is an n/ℓ bit string. (If identities are considered to be bit strings of arbitrary length, then as in Waters protocol we hash them into Z_N using a collision resistant hash function.)

In this case the protocol is changed to the following, which we call $\mathsf{IBE\text{-}SPP}(\ell)$.

$\mathsf{IBE\text{-}SPP}(\ell)$ **with** $1 < \ell \leq n$

Setup: Randomly choose a secret $x \in \mathbb{Z}_p$. Set $P_1 = xP$, then choose $P_2 \in G_1$ at random. Further, choose random elements $U', U_1, U_2, \ldots, U_\ell \in G_1$. The master secret is xP_2 whereas the public parameters are $\langle P, P_1, P_2, U', U_1, U_2, \ldots, U_\ell \rangle$.

Key-Gen: Let v be any identity, a secret key for v is generated as follows. Choose a random $r \in \mathbb{Z}_p^*$, then the private key for v is

$$d_{\mathsf{v}} = (xP_2 + rV, rP).$$

where $V = U' + \sum_{i=1}^{\ell} \mathsf{v}_i U_i$.

Encrypt: Any message $M \in G_2$ is encrypted for an identity v as

$$C = (e(P_1, P_2)^s \times M, sP, sV),$$

where s is a random element of $\mathbb{Z}_p$ and V is as defined in key generation algorithm.

Decrypt: Let $C = (C_1, C_2, C_3)$ be a ciphertext and v be the corresponding identity. Then we decrypt C using secret key $d_{\mathsf{v}} = (d_1, d_2)$ by computing

$$C_1 \times \frac{e(d_2, C_3)}{e(d_1, C_2)} = e(P_1, P_2)^s \times M \frac{e(rP, sV)}{e(xP_2 + rV, sP)} = M$$

Note that, for $\ell = n$ this is exactly Waters protocol. For $\ell = 1$, some minor modifications in the above scheme give a protocol where the additional requirement in the public parameter is just a single element of G_1 as described below.

IBE-SPP(1)

Setup: Randomly choose a secret $x \in Z_N$. Set $P_1 = xP$, then choose $P_2 \in G_1$ at random. Further, choose a random element $U' \in G_1$. The master secret is xP_2 whereas the public parameters are $\langle P, P_1, P_2, U' \rangle$.

Key-Gen: Let v be any identity. A secret key for v is generated as follows. Choose a random $r \in \mathbb{Z}_p^*$, then the private key for v is

$$d_{\mathsf{v}} = (xP_2 + rV, rP).$$

where $V = U' + \mathsf{v}P_2$.

Here the Encrypt and Decrypt algorithms are same as IBE-SPP(ℓ) with the modified definition of V. Note that, this is essentially the Boneh-Boyen HIBE of Section 3.2.1 restricted to IBE in the *adaptive*-ID model.

Efficiency: Consider IBE-SPP(ℓ) with $1 < \ell \leq n$. Let $\mathsf{cost}(V)$ be the cost of computing V. The cost of key generation is two scalar multiplications over G_1 plus $\mathsf{cost}(V)$. By including $e(P_1, P_2)$ instead of P_1, P_2 in the public parameter, we can avoid the pairing computation during encryption. So the cost of encryption is one exponentiation over G_2, two scalar multiplications over G_1 plus $\mathsf{cost}(V)$. The cost of decryption is two pairings, one multiplication and one inversion over G_2. The effect of ℓ is in $\mathsf{cost}(V)$ and affects key generation and encryption costs but does not affect decryption cost.

We first consider the costs of scalar multiplication over G_1 and exponentiation over G_2. As mentioned earlier, G_1 is an elliptic curve group. Let $\mathbb{F}_a$ denote the base field over which G_1 is defined. Then G_2 is a subgroup of $\mathbb{F}_a^k$, where k is the MOV degree. Additions and doublings over G_1 translate into a constant number of multiplications over $\mathbb{F}_a$. The actual number is slightly different for addition and doubling, but we will ignore this difference. Let $|\mathbb{F}_a|$ be the size of the representation of an element of $\mathbb{F}_a$. Assuming the cost of multiplication over G_1 is approximately equal to $|\mathbb{F}_a|^2$, the cost of a scalar multiplication over G_1 is equal to $c_1|\mathbb{F}_a|^3$ for some constant c_1. One can also show that the cost of exponentiation over G_2

is equal to $c_2|\mathbb{F}_a|^3$. Thus, the total cost of scalar multiplication and exponentiation is equal to $c|\mathbb{F}_a|^3$.

The cost of computing V amounts to computing ℓ scalar multiplications where each multiplier is an (n/ℓ)-bit string. On an average, the cost of each such multiplication will be $n/2\ell$ additions and $(n/\ell - 1)$ doublings over G_1. Hence, the total cost of computing V is $n/2$ additions and $(n - \ell)$ doublings over G_1. This cost is equal to $d(3/2 - \ell/n)n|\mathbb{F}_a|^2$ for some constant d.

We consider the cost of encryption. The total cost is

$$c|\mathbb{F}_a|^3 + d(3/2 - \ell/n)n|\mathbb{F}_a|^2 \;=\; \left(c + d \times \frac{n}{|\mathbb{F}_a|}\left(\frac{3}{2} - \frac{\ell}{n}\right)\right)|\mathbb{F}_a|^3. \qquad (5.2.1)$$

This cost is minimum when $\ell = n$ (as in Waters protocol). The maximum value of the coefficient of $|\mathbb{F}_a|^3$ is $(c + (3nd)/(2|\mathbb{F}_a|))$ whereas the minimum value is $(c + (nd)/(2|\mathbb{F}_a|))$. The value of $|\mathbb{F}_a|$ is usually greater than n and hence the value of $(nd)/(2|\mathbb{F}_a|)$ will be a small constant and hence there is not much effect of ℓ on the total cost of encryption. A similar analysis shows that the effect of ℓ is also not very significant on the cost of key generation. We note, however, that key generation is essentially a one-time offline activity.

5.2.1 Security Reduction

The security (in the sense of IND-ID-CPA) of the identity-based encryption scheme developed above (i.e., IBE-SPP(ℓ)) can be reduced from the hardness of the DBDH problem as stated in the following theorem.

THEOREM 5.2.1. *The IBE protocol described in Section 5.2 is* (ϵ_{ibe}, t, q)-IND-ID-CPA *secure assuming that the* (t', ϵ_{dbdh})*-DBDH assumption holds in* $\langle G_1, G_2, e\rangle$*, where* $\epsilon_{ibe} \leq 2\epsilon_{dbdh}/\lambda$*;* $t' = t + O(\tau q) + \chi(\epsilon_{ibe})$ *and*

$\chi(\epsilon) = O(\epsilon^{-2}\ln(\epsilon^{-1})\lambda^{-1}\ln(\lambda^{-1}))$*;*
τ *is the time required for one scalar multiplication in* G_1*;*
$\lambda = 1/(8q(\mu_\ell + 1))$ *with* $\mu_\ell = \ell(N^{1/\ell} - 1)$*,* $N = 2^n$*.*

Proof : Suppose $\mathcal{A}$ is a (t, q, ϵ_{ibe})-IND-ID-CPA adversary for IBE-SPP(ℓ). Then we construct an algorithm $\mathcal{S}$ for DBDH running in time $(t + O(\tau q) + \chi(\epsilon_{ibe}))$ such that, $\epsilon_{ibe} \leq 16q(\mu_\ell + 1)\epsilon_{dbdh}$, where $\mu_\ell = \ell(N^{1/\ell} - 1)$. $\mathcal{S}$ will take as input a 5-tuple $\langle P, aP, bP, cP, Z\rangle$ where P is a generator of G_1, $aP, bP, cP \in G_1$ and $Z \in G_2$. We define the following game between $\mathcal{S}$ and $\mathcal{A}$.

Setup: $\mathcal{S}$ first chooses random $x, x_1, \ldots, x_\ell \in \mathbb{Z}_m$ where $m < 4q$ is a prime; random $y, y_1, \ldots, y_\ell \in \mathbb{Z}_p$ and a random $k \in \{0, \ldots, \mu_\ell\}$. It then defines three functions: $F(\mathsf{v}) =$

$p - mk + x + \sum_{i=1}^{\ell} x_i \mathsf{v}_i$, $J(\mathsf{v}) = y + \sum_{i=1}^{\ell} y_i \mathsf{v}_i$ and

$$K(\mathsf{v}) = \begin{cases} 0 & \text{if } x + \sum_{i=1}^{\ell} x_i \mathsf{v}_i \equiv 0 \bmod m \\ 1 & \text{otherwise} \end{cases}$$

Here, $F(\mathsf{v})$ and $K(\mathsf{v})$ are defined in such a way that $K(\mathsf{v}) \neq 0$ implies $F(\mathsf{v}) \not\equiv 0 \bmod p$. Next, $\mathcal{S}$ assigns $P_1 = aP,\ P_2 = bP$, $U' = (p - mk + x)P_2 + yP$ and $U_i = x_i P_2 + y_i P$ for $1 \leq i \leq \ell$. It provides $\mathcal{A}$ the public parameters $\langle P, P_1, P_2, U', U_1, \ldots, U_\ell \rangle$. Everything else is internal to $\mathcal{S}$. Note that from $\mathcal{A}$'s point of view the distribution of the public parameters is identical to the distribution of the public parameters in an actual setup.

Phase 1: The adversary $\mathcal{A}$ issues key extraction queries. Suppose, the adversary asks for the private key corresponding to an identity v. $\mathcal{S}$ first checks whether $K(\mathsf{v}) = 0$ and aborts in that situation and outputs a random bit. Otherwise, it gives $\mathcal{A}$ the pair

$$(d_1, d_2) = \left(-\frac{J(\mathsf{v})}{F(\mathsf{v})} P_1 + r(F(\mathsf{v})P_2 + J(\mathsf{v})P), \frac{-1}{F(\mathsf{v})} P_1 + rP \right)$$

where r is chosen at random from $\mathbb{Z}_p$. Now,

$$\begin{aligned} d_1 &= -\frac{J(\mathsf{v})}{F(\mathsf{v})} P_1 + r\left(F(\mathsf{v})P_2 + J(\mathsf{v})P\right) \\ &= -\frac{J(\mathsf{v})}{F(\mathsf{v})} P_1 + \left(aP_2 - a\frac{F(\mathsf{v})}{F(\mathsf{v})} P_2 \right) + r\left(F(\mathsf{v})P_2 + J(\mathsf{v})P\right) \\ &= aP_2 - \frac{a}{F(\mathsf{v})}\left(F(\mathsf{v})P_2 + J(\mathsf{v})P\right) + r\left(F(\mathsf{v})P_2 + J(\mathsf{v})P\right) \\ &= aP_2 + \left(r - \frac{a}{F(\mathsf{v})} \right)\left(F(\mathsf{v})P_2 + J(\mathsf{v})P\right) \\ &= aP_2 + r'\left(F(\mathsf{v})P_2 + J(\mathsf{v})P\right) \\ &= aP_2 + r'\left((p - mk + x)P_2 + yP + \sum_{i=1}^{l} v_i(x_i P_2 + y_i P) \right) \\ &= aP_2 + r'V \text{where } r' = r - \frac{a}{F(\mathsf{v})} \end{aligned}$$

and

$$d_2 = \frac{-1}{F(\mathsf{v})} P_1 + rP = \frac{-1}{F(\mathsf{v})} aP + rP = (r - \frac{a}{F(\mathsf{v})})P = r'P$$

Hence, (d_1, d_2) is a valid private key for v following the proper distribution. $\mathcal{S}$ will be able to generate this pair (d_1, d_2) if and only if $F(\mathsf{v}) \not\equiv 0$, for which it suffices to have $K(\mathsf{v}) \neq 0$.

Challenge: At this stage, the adversary $\mathcal{A}$ submits two messages $M_0, M_1 \in G_2$ and an identity v^* with the constraint that it has not asked for the private key of v^* in Phase 1. $\mathcal{S}$

aborts if $F(\mathsf{v}^*) \neq 0$ and outputs a random bit. Otherwise, $\mathcal{S}$ chooses a random bit $\gamma \in \{0,1\}$ and gives $\mathcal{A}$ the tuple $C' = \langle Z \times M_\gamma, cP, J(\mathsf{v}^*)cP \rangle$.

If $\langle P, aP, bP, cP, Z \rangle$ given to $\mathcal{S}$ is a valid DBDH tuple, i.e., $Z = e(P,P)^{abc}$ then C' is a valid encryption for M_γ. Since,

$$e(P,P)^{abc} = e(aP, bP)^c = e(P_1, P_2)^c$$

and using $F(\mathsf{v}^*) = p - mk + x + \sum_{i=1}^{\ell} x_i \mathsf{v}_i^* \equiv 0 \bmod p$ we have,

$$\begin{aligned}
J(\mathsf{v}^*)cP &= c(y + \sum_{i=1}^{l} y_i \mathsf{v}_i^*)P \\
&= c((p - mk + x + \sum_{i=1}^{l} x_i \mathsf{v}_i^*)P_2 + (y + \sum_{i=1}^{l} y_i \mathsf{v}_i^*)P) \\
&= c((p - mk + x)P_2 + yP + \sum_{i=1}^{l} \mathsf{v}_i^*(x_i P_2 + y_i P)) \\
&= c(U' + \sum_{i=1}^{l} \mathsf{v}_i^* U_i) \\
&= cV
\end{aligned}$$

Note that, this condition is satisfied as long as $F(\mathsf{v}^*) \equiv 0 \bmod p$, which holds if $x + \sum_{j=1}^{\ell} x_j \mathsf{v}_j^* = km$. Otherwise, Z is a random element of G_2 and C' gives no information about $\mathcal{S}$'s choice of γ.

Phase 2: This phase is similar to Phase 1, with the obvious restriction that $\mathcal{A}$ cannot ask for the private key of v^*. We note that the total number of key extraction queries together in Phase 1 and 2 should not exceed q.

Guess: $\mathcal{A}$ outputs a guess γ' of γ. Then $\mathcal{S}$ outputs $1 \oplus \gamma \oplus \gamma'$.

Suppose the adversary has not aborted up to this point. Waters introduced a technique whereby the simulator is allowed to abort under certain condition. The simulator samples the transcript it received from the adversary during the attack phase. Based on the sample, it decides whether to abort and output a random string. The rationale for such "artificial abort" is the following: The probability of abort during the attack phase depends on the adversarial transcript and can be different for different transcripts. The purpose of artificial abort is to ensure that the simulator aborts with (almost) the same probability for all adversarial queries. This ensures that the adversary's success is independent of whether the simulator aborts or not. The probability analysis performed by Waters in [89] requires this independence. For details of this method see [89]. Here we just note that the artificial abort stage requires an additional $\chi = O(\epsilon^{-2} \ln(\epsilon^{-1}) \lambda^{-1} \ln(\lambda^{-1}))$ time. Further, it is independent of the parameter ℓ which defines the generalisation over Waters [89] that we introduce here.

Let abort be the event of the simulator aborting during the actual attack (as opposed to artificial abort) and let $\lambda = Pr[\overline{\mathsf{abort}}]$. In Proposition 6.3.3. of Chapter 6, we calculate this lower bound for the general case of HIBE. Here we use this lower bound for the special case of IBE, which turns out to be

$$\lambda \geq \frac{1}{8q(\mu_\ell + 1)}.$$

If $\mathcal{S}$ does not abort the game, then it simulates a real challenger for $\mathcal{A}$.

In the proof of Theorem 6.3.1. of Chapter 6, we show in a more generalised setting how the adversarial advantage against the protocol can be related with the simulator's advantage in solving the DBDH problem. Instantiating this for the case of IBE one gets

$$\epsilon_{ibe} \leq 16q(\mu_\ell + 1)\epsilon_{dbdh}.$$

This completes the proof. ∎

Note that, in the simulation, only the computation of $F(\mathsf{v}), J(\mathsf{v}) \in \mathbb{Z}_p$ depends on the size parameter ℓ. Once $F(\mathsf{v})$ and $J(\mathsf{v})$ are obtained, the key generation in Phase 1 and 2 and cipher text generation in Challenge is done through some scalar multiplications involving $F(\mathsf{v})$ and $J(\mathsf{v})$. Cost of computation of $F(\mathsf{v})$ and $J(\mathsf{v})$ are insignificant compared to the cost of a scalar multiplication. So the simulation time is independent of the size parameter ℓ.

5.3 Concrete Security

From the security reduction of previous section we observe that any (t, q, ϵ) adversary $\mathcal{A}$ against $\mathsf{IBE\text{-}SPP}(\ell)$ can actually be used to build an algorithm $\mathcal{B}$ to solve the DBDH problem over (G_1, G_2, e) which runs in time t' and has a probability of success ϵ'. Then $t' = t + O(\tau q) + \chi(\epsilon) \approx t + c\tau q + \chi'$ for some constant c and $\epsilon' \approx \epsilon/\delta$ where τ is the time for a group operation in G_1 and δ is the corresponding degradation in the security reduction. Resistance of $\mathsf{IBE\text{-}SPP}(\ell)$ against $\mathcal{A}$ can be quantified as $\rho^{(\ell)}_{|_\mathcal{A}} = \lg(t/\epsilon)$. To assert that $\mathsf{IBE\text{-}SPP}(\ell)$ has at least 80-bit security, we must have $\rho^{(\ell)}_{|_\mathcal{A}} \geq 80$ for all possible $\mathcal{A}$. Similarly, the resistance of DBDH against $\mathcal{B}$ can be quantified as

$$\rho_{|_\mathcal{B}} = \lg\left(\frac{t'}{\epsilon'}\right) \approx \lg\left(\delta \times \frac{t + c\tau q + \chi'}{\epsilon}\right) = \lg(\delta(A_1 + A_2))$$

where $A_1 = t/\epsilon$ and $A_2 = (c\tau q + \chi')/\epsilon$. We now use $\max(A_1, A_2) \leq A_1 + A_2 \leq 2\max(A_1, A_2)$. Since a factor of two does not significantly affect the analysis we put $\rho_{|_\mathcal{B}} = \lg(\delta \times \max(A_1, A_2))$. By our assumption, $A_1 = t/\epsilon \geq 2^{80}$ and hence $\max(A_1, A_2) \geq A_1 \geq 2^{80}$. This results in the condition $\rho_{|_\mathcal{B}} \geq 80 + \lg\delta$.

Thus, if we want $\mathsf{IBE\text{-}SPP}(\ell)$ to have 80-bit security, then we must choose the group sizes of G_1, G_2 in such a way that the best possible algorithm for solving DBDH in these groups takes time at least $2^{80+\lg\delta}$. Hence, in particular, the currently best known algorithm for solving the DBDH should also take this time. Currently the only method to solve the

DBDH problem over (G_1, G_2, e) is to solve the discrete log problem (DLP) over either G_1 or G_2. The best known algorithm for the former is the Pollard's rho method while that for the later is number/function field sieve. Thus, if we want IBE-SPP(ℓ) to have 80-bit security, then we must choose the group sizes such that, $2^{80+\lg\delta} \leq \mathsf{min}(t_{G_1}, t_{G_2})$, where t_{G_i} stands for the time to solve DLP in G_i for $i \in \{1, 2\}$.

We have assumed that G_1 is a group of elliptic curve points of order p defined over a finite field $\mathbb{F}_a$ (a is a prime power). Suppose G_2 is a subgroup of order p of the finite field $\mathbb{F}_{a^k}$ where k is the embedding degree. The Pollard's rho algorithm to solve ECDLP takes time $t_{G_1} = O(\sqrt{p})$, while the number/function field seive method to solve the DLP in $\mathbb{F}_{a^k}$ takes time $t_{G_2} = O(e^{c^{1/3}\ln^{1/3}a^k \ln^{2/3}(\ln a^k)})$ where $c = 64/9$ (resp. $32/9$) in large characteristic fields (resp. small characteristic fields).

5.3.1 Space/time trade-off

In this section, we parametrize the quantities by ℓ wherever necessary. Let, $\delta^{(\ell)}$ denote the degradation factor in IBE-SPP(ℓ). We have already noted in Section 5.2 that $\ell = n$ stands for Waters protocol. $\delta^{(\ell)}$ and hence $\rho^{(\ell)}$ is minimum when $\ell = n$ and we use this as a bench mark to compare with other values of ℓ. Suppose $\Delta\rho^{(\ell)} = \rho^{(\ell)} - \rho^{(n)} = \lg(\delta^{(\ell)}/\delta^{(n)}) = (n/\ell) - \lg(n/\ell)$. This parameter $\Delta\rho^{(\ell)}$ gives us an estimate of the extra bits required in case of IBE-SPP(ℓ), to achieve the same security level as that of IBE-SPP(n) i.e., Waters protocol.

Table 5.1: *Approximate group sizes for attaining 80-bit security for* IBE-SPP(ℓ) *for different values of ℓ and relative space and time requirement. The first part corresponds to $n = 160$ and the second to $n = 256$.*

ℓ	$\Delta\rho^{(\ell)}$	$\lvert p^{(\ell)}\rvert$	$\lvert G_2^{(\ell)}\rvert$		$\alpha^{(\ell)}$		$\beta^{(\ell)}$	
			(a)	(b)	(a)	(b)	(a)	(b)
160	–	246	1891(2225)	3284(3872)	–	–	–	–
8	15	276	2443(2831)	4258(4944)	6.5(6.4)	6.5(6.4)	2.16(2.06)	2.18(2.08)
16	6	258	2102(2457)	3655(4288)	11.1(11.0)	11.1(11.1)	1.37(1.35)	1.38(1.35)
32	2	250	1960(2300)	3405(4006)	20.7(20.7)	20.7(20.7)	1.11(1.11)	1.12(1.11)
80	1	248	1924(2262)	3344(3939)	50.9(50.8)	50.9(50.9)	1.05(1.05)	1.06(1.05)
256	–	246	1891(2225)	3284(3872)	–	–	-	–
8	27	300	2948(3381)	5151(5919)	4.9(4.7)	4.9(4.8)	3.79(3.51)	3.86(3.57)
16	12	270	2326(2703)	4051(4717)	7.7(7.6)	7.7(7.6)	1.86(1.79)	1.88(1.81)
32	5	256	2066(2417)	3592(4212)	13.7(13.6)	13.7(13.6)	1.30(1.28)	1.31(1.29)
64	2	250	1960(2300)	3405(4006)	25.9(25.8)	25.9(25.9)	1.11(1.11)	1.11(1.11)
128	1	248	1924(2262)	3344(3939)	50.9(50.8)	50.9(50.9)	1.05(1.05)	1.06(1.05)

Suppose, $|p^{(\ell)}|$ (resp. $|G_2^{(\ell)}|$) denotes the bit length of representation of $p^{(\ell)}$ (resp. an element of $G_2^{(\ell)}$). Like [46], we assume that the adversary $\mathcal{A}$ is allowed to make a maximum of $q = 2^{30}$ number of queries. For a given security level, we can now find the values of $|p^{(\ell)}|$ and $|G_2^{(\ell)}|$ for IBE-SPP(ℓ) based on the bit length of the identities (i.e., n), q and ℓ. Note that, the value of $|p^{(\ell)}|$ (resp. $|G_2^{(\ell)}|$) thus obtained is the *minimum* required to avoid the Pollard's rho (resp. number/function field seive) attack. In actual implementation, these values give an estimate of the size of suitable pairing groups G_1, G_2 and the embedding degree so that the requirements can be optimally met. In our comparison, the embedding degree k is taken to be same for different values of ℓ and $|G_2^{(\ell)}| = k \lg a$ ($G_2^{(\ell)}$ is a multiplicative subgroup of order $p^{(\ell)}$ of the finite field $\mathbb{F}_a^k$). As already noted, the value of $p^{(\ell)}$ is given by Pollard's rho. On the other hand, the logarithm of the size of $G_1^{(\ell)}$ is equal to $\mathsf{max}(p^{(\ell)}, |G_2^{(\ell)}|/k)$. For relatively small embedding degree (i.e., $k \leq 6$), $|G_2^{(\ell)}|/k > |p^{(\ell)}|$ and so the logarithm of the size of $G_1^{(\ell)}$ is equal to $|G_2^{(\ell)}|/k = |\mathbb{F}_a^{(\ell)}|$. For a given ℓ, we have to store ℓ elements of $G_1^{(\ell)}$ in the public parameter file and a scalar multiplication in $G_1^{(\ell)}$ takes time proportional to $(|\mathbb{F}_a^{(\ell)}|)^3$.

Now, we are in a position to compare the space requirement in the public parameter file and the time requirement for a scalar multiplication in $G_1^{(\ell)}$ for different values of ℓ. Let $\alpha^{(\ell)} = \frac{\ell \times |G_1^{(\ell)}|}{n \times |G_1^{(n)}|} \times 100$ i.e., the relative amount of space (expressed in percentage) required to store the public parameters in case of IBE-SPP(ℓ) with respect to IBE-SPP(n) and $\beta^{(\ell)} = |\mathbb{F}_a^{(\ell)}|^3/|\mathbb{F}_a^{(n)}|^3$, i.e., the relative increase in time for scalar multiplication in $G_1^{(\ell)}$in the case of IBE-SPP(ℓ) with respect to IBE-SPP(n). Note that, $\beta^{(\ell)}$ can be computed from $|G_2^{(\ell)}|$ and $|G_2^{(n)}|$ since k cancels out from both numerator and denominator. We have seen in Chapter 4 that pairing computation is also of order $|\mathbb{F}_a^{(\ell)}|^3$ (but with a larger constant factor). So, the ratio $\beta^{(\ell)}$ also holds for pairing computation and exponentiation in case of IBE-SPP(ℓ) with respect to Waters protocol.

In Table 5.1, we sum-up these results for $n = 160$ and 256 for different values of ℓ ranging from 8 to n for 80-bit security. The subcolumns (a) and (b) under $\alpha^{(\ell)}$ and $\beta^{(\ell)}$ stand for the values obtained for general characteristic field and field of characteristic three respectively. The values of $|G_2^{(\ell)}|, \alpha^{(\ell)}, \beta^{(\ell)}$ are computed using the formula as suggested in [46] (see Section 3); while in parenthesis we give the corresponding values as computed from the formula obtained from [67] (as given in Section 3 of [46]). Note that, the values of $\alpha^{(\ell)}$ and $\beta^{(\ell)}$ being the ratio of two quantities remain more or less invariant whether the underlying field is a general characteristic field or a field of characteristic three or which formula (of [46] or of [67]) is used.

Public parameter consists of $(\ell + 4)$ elements of G_1. From Table 5.1, for 80-bit security in general characteristic fields using EC with MOV degree 2, the public parameter size for Waters protocol will be around 37 kilobyte (kb) for 160-bit identities and 59 kb for 256-bit identities. The corresponding values in case of IBE-SPP(ℓ) with $\ell = 16$ will be around 4 kb and 4.5 kb respectively. Similarly, in characteristic three field EC with MOV degree 6, the corresponding values are respectively 21.5 kb and 34.2 kb and for IBE-SPP(ℓ) with $\ell = 16$

these are respectively 2.4 kb and 2.64 kb. There is an associated increase in computation cost by 30%. In typical applications, the protocol will be used in a key encapsulation mechanism (KEM). Thus the encryption and decryption algorithms will be invoked once for a message irrespective of its length. Also the key generation procedure is essentially a one-time offline activity. In view of this, the increase in computation cost will not substantially affect the throughput. On the other hand, the significant reduction in space requirement will be an advantage in implementing the protocol and also in reducing the time for downloading or transmitting the public parameter file over the net. Overall, we suggest $\ell = 16$ to be a good choice for implementing the protocol.

5.4 CCA Security

One way to achieve CCA-security for our scheme is to follow the generic transformation of Canetti, Halevi and Katz [27] or Boneh and Katz [21] as discussed in Section 3.5. As our scheme closely resembles that of Waters [89] it is also possible to apply the endogenous transformation of Boyen, Mei and Waters [23] in essentially the same way and the reduction follows.

We show that it is possible to take a different approach based on the oracle bilinear decision Diffie-Hellman (OBDH) assumption which is a variation of the ODH assumption used in [2]. The OBDH assumption is as follows [81].

- Instance : $\langle P, aP, bP, cP, \mathsf{str} \rangle$ where $a, b, c \in \mathbb{Z}_p$ and $\mathsf{str} \in \{0,1\}^k$.
- Oracle : $\mathcal{H}_a(X, Y)$, with $X, Y \in G_1$. When invoked with $(a_1 P, b_1 P)$ it returns $H(a_1 P, e(a_1 P, b_1 P)^a)$, where $H : G_1 \times G_2 \rightarrow \{0,1\}^k$ is a hash function.
- Restriction : Cannot query $\mathcal{H}_a(,)$ on (cP, bP).
- Task : Determine whether $\mathsf{str} = H(cP, e(cP, bP)^a)$ or str is random.

Any algorithm $\mathcal{A}$ for OBDH takes as input an instance $(P, aP, bP, cP, \mathsf{str})$ of OBDH and produces as output either zero or one. The advantage of an algorithm $\mathcal{A}$ in solving OBDH is defined in the following manner.

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{OBDH}} \;=\; |\mathsf{Pr}[\mathcal{A} \text{ outputs } 1|E_1] - \mathsf{Pr}[\mathcal{A} \text{ outputs } 1|E_2]|$$

where E_1 is the event that $\mathsf{str} = H(cP, e(cP, bP)^a)$ and E_2 is the event that str is random. The quantity $\mathsf{Adv}^{\mathsf{OBDH}}(t, q)$ denotes the maximum of $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{OBDH}}$ where the maximum is taken over all adversaries running in time at most t and making at most q queries to the oracle $\mathcal{H}_a(,)$.

To suit into the OBDH assumption we modify our constructions of Section 5.2 as follows: **Setup** and **Key-Gen** remain unaltered. To encrypt a message, we first generate a symmetric key $\mathsf{sym.key} = H(tP, e(P_1, P_2)^t)$. Then the ciphertext is $C = \langle tP, tV, y) \rangle$, where y is the

encryption of the message using the symmetric key sym.key. To decrypt, all that we need is $e(P_1, P_2)^t = e(d_1, tP)/e(d_2, tV)$ and then find sym.key using H.

Security: Breaking the (modified) IBE implies either solving OBDH or breaking the symmetric encryption scheme. The later we assume to be unbreakable under chosen ciphertext attack. CCA security under the OBDH assumption is expressed in the following theorem.

THEOREM 5.4.1. *The modified IBE protocol is* (ϵ_{ibe}, t, q)-IND-ID-CCA *secure assuming that the* (t', ϵ_{obdh})*-OBDH assumption holds in* $\langle G_1, G_2, e\rangle$*, where* $\epsilon_{ibe} \leq 2\epsilon_{obdh}/\lambda$*;* $t' = t + O(\tau q) + \chi(\epsilon_{ibe})$*, where* λ*,* $\chi(\epsilon_{ibe})$ *and* τ *are as defined in Theorem 5.2.1.*

Proof : (*Brief sketch*) Given a tuple $\langle P, aP, bP, cP, \mathsf{str}\rangle$, the simulator $\mathcal{B}$ has to decide whether $\mathsf{str} = H(cP, e(cP, bP)^a)$ or str is random. The Setup and key-extraction queries of both Phase 1 and 2 are just the same as that in the simulation of Theorem 5.2.1. of Section 5.2.1. Whenever $\mathcal{A}$ places any decryption query $C = \langle rP, rV, y\rangle$, $\mathcal{B}$ queries the oracle $\mathcal{H}_a()$ with (rP, P_2) and decrypts y using whatever value the oracle returns.

In the Challenge phase, when $\mathcal{A}$ submits two messages M_0, M_1 and an identity v^*, $\mathcal{B}$ aborts if $\mathcal{S}$ would have aborted under v^* in the simulation part of Theorem 5.2.1.. Otherwise $\mathcal{B}$ gives $\mathcal{A}$ the tuple $C' = \langle cP, J(I^*)cP, y\rangle$, where y is the encryption of M_γ, $\gamma \in \{0, 1\}$ using str as the symmetric key. If str is random then C' gives no information about $\mathcal{B}$'s choice of γ. Otherwise C' is a valid encryption of M_γ.

The rest of the simulation exactly mimics that of Theorem 5.2.1. ∎

Use of OBDH assumption prevents the loss of one level in the conversion from CPA security to CCA security and does not require any one time signature or MAC as in the generic conversion discussed in Section 3.5. Using the endogenous technique of Boyen, Mei and Waters [23] we can avoid this MAC or signature. However, it still requires a 2 level HIBE. On the other hand, OBDH is a stronger assumption than DBDH.

5.5 Signature

It is an observation of Naor that any identity-based encryption scheme can be converted to a signature scheme. Waters in his paper [89] has given a construction of a signature scheme based on his IBE scheme. A similar construction is possible for the generalised scheme IBE-SPP(ℓ) which we detail here. The sketch of the security reduction is provided next.

Let $G_1 = \langle P\rangle$, G_2 and $e()$ be as defined in Section 2.1. Messages are assumed to be elements of $\mathbb{Z}_N$ where $N = 2^n$. Alternatively, if messages are assumed to be bit strings of arbitrary length, then we use a collision resistant hash function to map the messages into $\mathbb{Z}_N$.

Setup: Choose a random x in $\mathbb{Z}_p$ and compute $P_1 = xP$. Next, choose from G_1 random points $P_2, U', U_1, \ldots, U_\ell$. The public key is $\langle P, P_1, P_2, U', U_1, \ldots, U_\ell\rangle$ and the secret key is xP_2.

Signing: Let $M = (m_1, m_2, \ldots, m_\ell)$ is the message to be signed, where each $m_i, 1 \leq i \leq \ell$ is a bit string of length n/ℓ. To generate a signature on M, first choose a random $r \in \mathbb{Z}_p^*$. Then the signature is

$$\sigma_M = (xP_2 + rV, rP),$$

where $V = U' + \sum_{i=1}^{\ell} m_i U_i$

Verification: Given a message $M = (m_1, m_2, \ldots, m_\ell)$ and a signature $\sigma = (\sigma_1, \sigma_2)$ on M, one accepts σ as a valid signature on M if

$$e(\sigma_1, P) = e(P_1, P_2)e(\sigma_2, V)$$

where $V = U' + \sum_{i=1}^{\ell} m_i U_i$.

5.5.1 Security

The security of the above signature scheme can be reduced from the hardness of the DBDH problem. In fact, using the same argument, we can show that the reduction in Theorem 5.2.1. also holds for this signature scheme. Moreover, the forged signature that the adversary returns can be used to break the computational Diffie-Hellman problem (CDH) in G_1. The CDH problem in G_1 is: given a tuple $\langle P, aP, bP \rangle$, compute abP. The success probability of an adversary $\mathcal{B}$ in solving the CDH problem in G_1 is defined as

$$\mathsf{Succ}_{\mathcal{B}}^{\mathsf{CDH}} = \mathsf{Pr}[\mathcal{B}(P, aP, bP) = abP]$$

where the probability is calculated over the random choice of $a, b \in \mathbb{Z}_p$ as well as the random bits used by $\mathcal{B}$. Let $(\mathsf{Adv}^{\mathsf{SIG}}(t, q)$ in this context denote the maximum advantage where the maximum is taken over all adversaries running in time t and making at most q queries.

THEOREM 5.5.1. *For $t \geq 1$, $q \geq 1$; $\mathsf{Adv}^{\mathsf{SIG}}(t, q) \leq (2/\lambda)\mathsf{Succ}^{\mathsf{CDH}}(t + O(\tau q))$, where messages are chosen from Z_N and $1 < \ell \leq \lg N$ is a size parameter.*

Proof : *Brief sketch:* This proof also is a reduction. Suppose $\mathcal{A}$ is a CPA adversary for the signature scheme. Then we construct an algorithm $\mathcal{S}$ for Computational Diffie-Hellman problem (CDH). $\mathcal{S}$ will take as input a 3-tuple $\langle P, aP, bP \rangle$ where P is a generator of G_1 and $aP, bP \in G_1$. We define the following game between $\mathcal{S}$ and $\mathcal{A}$.

The Setup and Signature Generation steps of this game is exactly same as the Setup and Phase 1 in the simulation part of Theorem 5.2.1.

Forge: At this stage the adversary $\mathcal{A}$ submits a message $M^* \in Z_N$ and a signature $\sigma^* = (\sigma_1^*, \sigma_2^*)$ with the constraint that it has not asked for the signature of M^* in the Signature Generation phase. $\mathcal{A}$ wins if σ^* is a valid signature on M^*.

If $\mathcal{A}$ is successful in forging the signature, $\mathcal{S}$ first checks whether $F(M^*) \neq 0$ and aborts in that situation. Otherwise, $\mathcal{S}$ computes $J(M^*)\sigma_2^*$ and then adds the inverse of this product with σ_1^*. It returns the end result as the value of abP.

Since $F(M^*) = 0$, then as in the Challenge part of the simulation in Theorem 5.2.1. we have

$$J(M^*)\sigma_2^* = rV.$$

$$\begin{aligned} J(M^*)\sigma_2^* &= J(M^*)rP \\ &= r(y + \sum_{i=1}^{l} y_i m_i^*)P \\ &= r((p - mk + x + \sum_{i=1}^{l} x_i m_i^*)P_2 + (y + \sum_{i=1}^{l} y_i m_i^*)P) \\ &= r((p - mk + x)P_2 + yP + \sum_{i=1}^{l} m_i(x_i P_2 + y_i P)) \\ &= r(u' + \sum_{i=1}^{l} m_i u_i) \\ &= rV \end{aligned}$$

Note that, this condition is satisfied as long as $F(M^*) \equiv 0 \bmod p$, which holds if $x + \sum_{j=1}^{\ell} x_j m_j^* = km$.

Now, $\sigma_1^* = abP + rV$ and hence $abP = \sigma_1^* - rV$.

Note that, the conditions under which $\mathcal{S}$ aborts this game is exactly the same under which $\mathcal{S}$ aborts the game in Theorem 5.2.1. So the lower bound on the probability of not aborting remains exactly the same.

5.6 Conclusion

At Eurocrypt 2005, Brent Waters proposed an efficient IBE scheme which is secure in the standard model. One drawback of this scheme is that the number of elements in the public parameter is rather large. In this chapter, we have proposed a generalisation of Waters scheme. In particular, we have shown that there is an interesting trade-off between the tightness of the security reduction and smallness of the public parameter. For a given security level, this implies that if one reduces the number of elements in public parameter then there is a corresponding increase in the computational cost due to the increase in group size. This introduces a flexibility in choosing the public parameter size without compromising in security. In concrete terms, to achieve 80-bit security for 160-bit identities we show that compared to Waters protocol the public parameter size can be reduced by almost 90% while increasing the computation cost by 30%. Our construction is proven secure in the standard

model without random oracles. Additionally, we show that CCA security can also be achieved through the reduction to oracle decision bilinear Diffie-Hellman problem (OBDH).

We note that Naccache [77] has independently obtained a similar construction as of ours in Section 5.2. Though the construction is similar, the work by Naccache does not perform any concrete security analysis. In fact, the Naccache work asserts that the loss of security due to the generalisation is "insignificant". As discussed in Section 5.3, this is not correct. In fact, the conversion of security degradation into a trade-off between time and space is original to our work and is the most important feature of the generalisation of Waters scheme. On the other hand, we would like to mention that Naccache's work presents a clearer probability analysis than that of Waters.

Chapter 6

Extending IBE to HIBE with Short Public Parameters

6.1 Introduction

In this chapter, we present a hierarchical identity-based encryption scheme which can be proved to be secure in the full model without random oracle assuming the hardness of decisional bilinear Diffie-Hellman problem. The construction extends the IBE of the previous chapter to a HIBE. A suggestion for extending the IBE of Waters [89] to a HIBE was provided in [89] itself.

Recall from Section 3.3.1 that Waters' IBE uses U', $U_1, \ldots, U_n$ (and P, P_1, P_2) as public parameters. His suggestion to extend this to a HIBE is to have new public parameters for each level. For an h-level HIBE, the public parameters will be of the form $U'_1, U_{1,1}, \ldots, U_{1,n}$, $U'_2, U_{2,1}, \ldots, U_{2,n}$, $\ldots$, $U'_h, U_{h,1}, \ldots, U_{h,n}$. The parameters P, P_1, P_2 are still required giving rise to $3 + (n+1)h$ many parameters.

The HIBE construction in this chapter uses public parameters of the form $U'_1, \ldots, U'_h$, $U_1, \ldots, U_l$ for $1 \leq l \leq n$. In other words, the parameters $U'_1, \ldots, U'_h$ correspond to the different levels of the HIBE, whereas the parameters $U_1, \ldots, U_l$ are the same for all the levels. These parameters $U_1, \ldots, U_l$ are reused in the key generation procedure. For $l = n$, we require $3 + n + h$ parameters compared to $3 + (n+1)h$ parameters in Waters' suggestion.

Thus, our work provides two things. First, by reusing public parameters it reduces the size of the public parameters. Second, it extends the flexibility in the IBE protocol of Chapter 5 to the HIBE setting. The reuse of public parameters over the different levels of the HIBE complicates the security proof. A straightforward extension of the independence results and lower bound proofs from [89] is not possible. We provide complete proofs of the required results. The constructed HIBE is proved to be secure under chosen plaintext attack (CPA-secure). Standard techniques as discussed in Section 3.5 can convert such a HIBE into one which is secure against chosen ciphertext attack (CCA-secure).

6.2 Construction

HIBE-spp

The identities are of the type $(\mathsf{v}_1, \ldots, \mathsf{v}_j)$, for $\leq j \leq h$ where each $\mathsf{v}_k = (\mathsf{v}_{k,1}, \ldots, \mathsf{v}_{k,l})$ and $\mathsf{v}_{k,i}$ is an (n/l)-bit string which will also be considered to be an integer in the set $\{0, \ldots, 2^{n/l} - 1\}$. Choosing $l = n$ gives v_k to be an n-bit string as considered by Waters [89].

Set-Up: The public parameters are the following elements: P, $P_1 = \alpha P$, P_2, $U'_1, \ldots, U'_h$, $U_1, \ldots, U_\ell$, where $G_1 = \langle P \rangle$, α is chosen randomly from $\mathbb{Z}_p$ and the other quantities are chosen randomly from G_1.

The master secret is αP_2. (The quantities P_1 and P_2 are not directly required; instead $e(P_1, P_2)$ is required. Hence one may store $e(P_1, P_2)$ as part of the public parameters instead of P_1 and P_2.)

A Useful Notation: Let $v = (v_1, \ldots, v_l)$, where each v_i is an (n/l)-bit string and is considered to be an element of $\mathbb{Z}_{2^{n/l}}$. For $1 \leq k \leq h$ we define,

$$V_k(v) \quad = \quad U'_k + \sum_{i=1}^{l} v_i U_i. \tag{6.2.1}$$

When v is clear from the context we will write V_k instead of $V_k(v)$. The modularity introduced by this notation allows an easier understanding of the protocol.

Note that for the jth level of the HIBE, we add a single element, i.e., U'_j in the public parameter while the elements $U_1, \ldots, U_l$ are re-used for each level. This way we are able to shorten the public parameter size. Later in the security reduction we show that the simulator forms U'_js, $1 \leq j \leq h$ in such a way that it is able to answer the adversarial queries.

Key-Gen: Let $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$, $j \leq h$, be the identity for which the private key is required. Choose $r_1, \ldots, r_j$ randomly from $\mathbb{Z}_p$ and define $d_{\mathsf{v}} = (d_0, d_1, \ldots, d_j)$ where

$$d_0 = \alpha P_2 + \sum_{k=1}^{j} r_k V_k(\mathsf{v}_k)$$

and $d_k = r_k P$ for $1 \leq k \leq j$.

Key delegation can be done in the manner shown in [17]. Suppose $(d'_0, d'_1, \ldots, d'_{j-1})$ is a private key for the identity $(\mathsf{v}_1, \ldots, \mathsf{v}_{j-1})$. To generate a private key for v, r_j randomly from $\mathbb{Z}_p$ and compute d_{v} as follows.

$$\begin{aligned} d_0 &= d'_0 + r_j V_j(\mathsf{v}_j); \\ d_i &= d'_i \qquad 1 \leq i \leq j-1; \\ d_j &= r_j P. \end{aligned}$$

Encrypt: Let $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ be the identity under which a message $M \in G_2$ is to be encrypted. Choose s to be a random element of $\mathbb{Z}_p$. The ciphertext is

$$(C_0 = M \times e(P_1, P_2)^s, C_1 = sP, B_1 = sV_1(\mathsf{v}_1), \ldots, B_j = sV_j(\mathsf{v}_j)).$$

Decrypt: Let $C = (C_0, C_1, B_1, \ldots, B_j)$ be a ciphertext and the corresponding identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$. Let $(d_0, d_1, \ldots, d_j)$ be the decryption key corresponding to the identity v. The decryption steps are as follows.

Verify whether C_0 is in G_2, C_1 and the B_i's are in G_1. If any of these verifications fail, then return bad, else proceed with further decryption as follows. Return

$$C_0 \times \frac{\prod_{k=1}^{j} e(B_i, d_i)}{e(d_0, C_1)}.$$

It is standard to verify the consistency of decryption.

6.3 Security

We first state the result on security and discuss its implications.

THEOREM 6.3.1. *The protocol* HIBE-spp *described in Section 6.2 is* (ϵ_{hibe}, t, q)-IND-ID-CPA *secure assuming that the* (t', ϵ_{dbdh})*-DBDH assumption holds in* $\langle G_1, G_2, e\rangle$*, where* $\epsilon_{hibe} \leq 2\epsilon_{dbdh}/\lambda$*;* $t' = t + \chi(\epsilon_{hibe})$ *and*

$\chi(\epsilon) = O(\tau q + O(\epsilon^{-2}\ln(\epsilon^{-1})\lambda^{-1}\ln(\lambda^{-1}))$*;*
τ *is the time required for one scalar multiplication in* G_1*;*
$\lambda = 1/(2(2\sigma(\mu_l + 1))^h)$ *with* $\mu_l = l(N^{1/l} - 1)$*,* $N = 2^n$ *and* $\sigma = \max(2q, 2^{n/l})$.

We further assume $2\sigma(1 + \mu_l) < p$.

Before proceeding to the proof, we discuss the above result. The main point of the theorem is the bound on ϵ_{hibe}. This is given in terms of λ and in turn in terms of μ_l. We simplify this bound.

Since $l \geq 1$, we have $1 + \mu_l = 1 + l(N^{1/l} - 1) \leq lN^{1/l} = l2^{n/l}$. Consequently,

$$\begin{aligned} \epsilon_{hibe} &\leq \frac{2\epsilon_{dbdh}}{\lambda} = 4(2\sigma(\mu_l + 1))^h \epsilon_{dbdh} \\ &\leq 4(2\sigma l 2^{n/l})^h \epsilon_{dbdh} \\ &= 4(2l2^{n/l})^h \sigma^h \epsilon_{dbdh} \end{aligned} \tag{6.3.2}$$

The reduction is not tight; security degrades by a factor of $4(2l2^{n/l})^h\sigma^h$. We now consider several cases. The actual value of degradation depends on the value of q, the number of key extraction queries made by the adversary. A value of q used in the previous chapter and also in earlier analysis is $q = 2^{30}$ [47]. We will use this value of q in the subsequent analysis.

Table 6.1: Comparison of HIBE Protocols.

HIBE Protocol	Hardness Assumption	Random Oracle	Security Degradation	PP size (elts. of G_1)	PK/CT size (elts. of G_1)	Pairing	
						Enc.	Dec.
GS [49]	BDH	Yes	$q_H q^h$	2	j	1	j
Waters [89]	DBDH	No	$(32nq)^h$	$(n+1)h+3$	$j+1$	None	$j+1$
HIBE-spp	DBDH	No	$4(2l2^{n/l}\sigma)^h$	$h+l+3$	$j+1$	None	$j+1$

The maximum height of the HIBEs is h; PP, PK and CT respectively stand for public parameter, private key and ciphertext. We compare the PK/CT sizes and number of pairings during decryption for an identity tuple at level j.

$h = 1$ and $l = n$: The value of $h = 1$ implies that the HIBE is actually an IBE and $l = n$ implies that each identity is a bit vector of length n. This is the situation originally considered by Waters [89]. In this case, $2q = \max(2q, 2^{n/l})$ and Equation (6.3.2) reduces to $\epsilon_{hibe} \leq 32nq\epsilon_{dbdh}$. For $n = 160$, the degradation is by a factor of 10×2^{39}.

$h = 1$: This correspond to the IBE considered in the previous chapter. The value of σ depends on the value of l. For example, if $l = 1$, then $\sigma = 2^{n/l} = 2^n$ and hence $\epsilon_{hibe} \leq 2^{2(n+1)}\epsilon_{dbdh}$. This makes the security degradation unacceptably bad. One would obtain a similar security degradation when converting a selective-ID secure IBE to an IBE secure in the full model. The situation for other values of l has already been discussed in the concrete security analysis of Section 5.3 and hence not repeated here.

$h > 1$: This corresponds to a proper HIBE. If $l = n$, then we obtain $\epsilon_{hibe} \leq 4(8nq)^h\epsilon_{dbdh}$. For $n = 160$ (and $q = 2^{30}$), this amounts to $\epsilon_{hibe} \leq 4(10 \times 2^{37})^h$. We consider a few other values of l. If $l = 10$, then $\epsilon_{hibe} \leq 4(10 \times 2^{48})^h\epsilon_{dbdh}$ and if $l = 32$, then $\epsilon_{hibe} \leq 2^{42h+2}\epsilon_{dbdh}$.

In Table 6.1, we compare the known HIBE protocols which are secure in the full model. We note that HIBE protocols which are secure in the selective-ID model are also secure in the full model with a security degradation of $\approx 2^{nh}$, where h is the number of levels in the HIBE and n is number of bits in the identity. This degradation is far worse than the protocols in Table 6.1. For the GS-HIBE [49], the parameter q_H stands for the total number of random oracle queries and in general $q_H \approx 2^{60} \gg q$ [47]. The parameter j in the private key size, ciphertext size and the encryption and decryption columns of Table 6.1 represents the number of levels of the identity on which the operations are performed. The parameter h is the maximum number of levels in the HIBE. For $\ell = n$, HIBE-spp requires $(h + n + 3)$ many elements of G_1 as public parameters whereas Waters suggestion requires $(n+1)h+3$ many elements. The security degradation remains the same in both cases. For $\ell < n$, the new construction extends the IBE protocol of Chapter 5. In this setting, no previous HIBE protocols were known.

6.3.1 Security Reduction

The security reduction follows along standard lines and develops on the proof given in previous chapter and in [89, 77]. We need to lower bound the probability of the simulator aborting on certain queries and in the challenge stage. The details of obtaining this lower bound is given in Section 6.3.2. In the following proof, we simply use the lower bound. We want to show that the HIBE is (ϵ_{hibe}, t, q)-CPA secure. In the game sequence style of proofs, we start with the adversarial game defining the CPA-security of the protocol against an adversary $\mathcal{A}$ and then obtain a sequence of games as usual. In each of the games, the simulator chooses a bit b and the adversary makes a guess b'. By X_i we will denote the event that the bit b is equal to the bit b' in the ith game.

Game 0: This is the usual adversarial game used in defining CPA-secure HIBE. We assume that the adversary's runtime is t and it makes q key extraction queries. Also, we assume that the adversary maximizes the advantage among all adversaries with similar resources. Thus, we have $\epsilon_{hibe} = \left|\mathsf{Pr}[X_0] - \frac{1}{2}\right|$.

Game 1: In this game, we setup the protocol from a tuple $\langle P, P_1 = aP, P_2 = bP, P_3 = cP, Z = e(P_1, P_2)^{abc}\rangle$ and answer key extraction queries and generate the challenge. The simulator is assumed to know the values a, b and c. However, the simulator can setup the protocol as well as answer certain private key queries without the knowledge of these values. Also, for certain challenge identities it can generate the challenge ciphertext without the knowledge of a, b and c. In the following, we show how this can be done. If the simulator cannot answer a key extraction query or generate a challenge without using the knowledge of a, b and c, it sets a flag flg to one. The value of flg is initially set to zero.

Note that the simulator is always able to answer the adversary (with or without using a, b and c). The adversary is provided with proper replies to all its queries and is also provided the proper challenge ciphertext. Thus, irrespective of whether flg is set to one, the adversary's view in Game 1 is same as that in Game 0. Hence, we have $\mathsf{Pr}[X_0] = \mathsf{Pr}[X_1]$.

We next show how to setup the protocol and answer the queries based on the tuple $\langle P, P_1 = aP, P_2 = bP, P_3 = cP, Z = e(P_1, P_2)^{abc}\rangle$.

Set-Up: Recall that $\sigma = \max(2q, 2^{n/l})$. Let m be a prime such that $\sigma < m < 2\sigma$. Our choice of m is different from that of previous works [89, 77] where m was chosen to be equal to $4q$ and $2q$.

Choose $x'_1, \ldots, x'_h$ and $x_1, \ldots, x_l$ randomly from $\mathbb{Z}_m$; $y'_1, \ldots, y'_h$ and $y_1, \ldots, y_l$ randomly from $\mathbb{Z}_p$. Choose $k_1, \ldots, k_h$ randomly from $\{0, \ldots, \mu_l\}$.

For $1 \leq j \leq h$, define $U'_j = (p - mk_j + x'_j)P_2 + y'_j P$ and for $1 \leq i \leq l$ define $U_i = x_i P_2 + y_i P$. Set the public parameters of HIBE to be $(P, P_1, P_2, U'_1, \ldots, U'_h, U_1, \ldots, U_l)$. The master secret is $aP_2 = abP$. The distribution of the public parameters is as expected by $\mathcal{A}$. In its attack, $\mathcal{A}$ will make some queries, which have to be properly answered by the simulator.

For $1 \leq j \leq h$, we define several functions. Let $v = (v_1, \ldots, v_l)$ where each v_i is an n/l-bit string considered to be an integer from the set $\{0, \ldots, 2^{n/l} - 1\}$. We define

$$\left.\begin{array}{rcl} F_j(v) & = & p - mk_j + x'_j + \sum_{i=1}^{l} x_i v_i \\ J_j(v) & = & y'_j + \sum_{i=1}^{l} y_i v_i \\ L_j(v) & = & x'_j + \sum_{i=1}^{l} x_i v_i \pmod m \\ K_j(v) & = & \begin{cases} 0 & \text{if } L_j(v) = 0 \\ 1 & \text{otherwise.} \end{cases} \end{array}\right\} \tag{6.3.3}$$

Recall that we have assumed $2\sigma(1 + \mu_l) < p$. Let $F_{\min}$ and $F_{\max}$ be the minimum and maximum values of $F_j(v)$. $F_{\min}$ is achieved when k_j is maximum and x'_j and the x_i's are all zero. Thus, $F_{\min} = p - m\mu_l$. We have $m\mu_l < 2\sigma(1 + \mu_l)$ and by assumption $2\sigma(1 + \mu_l) < p$. Hence, $F_{\min} > 0$. Again $F_{\max}$ is achieved when $k_j = 0$ and x'_j and the x_i's and v_i's are equal to their respective maximum values. We get $F_{\max} < p + m(1 + l(2^{n/l} - 1)) = p + m(1 + \mu_l) < p + 2\sigma(1 + \mu_l) < 2p$. Thus, we have $0 < F_{\min} \leq F_j(v) \leq F_{\max} < 2p$. Consequently, $F_j(v) \equiv 0 \bmod p$ if and only if $F_j(v) = p$ which holds if and only if $-mk_j + x'_j + \sum_{i=1}^{l} x_i v_i = 0$.

Now we describe how the queries made by $\mathcal{A}$ are answered by $\mathcal{B}$. The queries can be made in both Phases 1 and 2 of the adversarial game (subject to the usual restrictions). The manner in which they are answered by the simulator is the same in both the phases.

Key Extraction Query: Suppose $\mathcal{A}$ makes a key extraction query on the identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$. Suppose there is a u with $1 \leq u \leq j$ such that $K_u(\mathsf{v}_u) = 1$. Otherwise set flg to one. In the second case, the simulator uses the value of a to return the proper decryption key $d_{\mathsf{v}} = (aP_2 + \sum_{i=1}^{j} r_i V_i, r_1 V_1, \ldots, r_j V_j)$. In the first case, the simulator constructs a decryption key in the following manner.

Choose random $r_1, \ldots, r_j$ from $\mathbb{Z}_p$ and define

$$\left.\begin{array}{rcl} d_{0|u} & = & -\frac{J_u(\mathsf{v}_u)}{F_u(\mathsf{v}_u)} P_1 + r_u (F_u(\mathsf{v}_u) P_2 + J_u(\mathsf{v}_u) P) \\ d_u & = & \frac{-1}{F_u(\mathsf{v}_u)} P_1 + r_u P \\ d_k & = & r_k P \text{ for } k \neq u \\ d_{\mathsf{v}} & = & (d_{0|u} + \sum_{k \in \{1,\ldots,j\} \setminus \{u\}} r_k V_k, d_1, \ldots, d_j) \end{array}\right\} \tag{6.3.4}$$

The quantity d_{v} is a proper private key corresponding to the identity v. The can be verified using an algebraic technique similar to that in the simulation of Theorem 5.2.1. of Chapter 5. This is provided to $\mathcal{A}$.

Challenge: Let the challenge identity be $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_{h^*}^*)$, $1 \leq h^* \leq h$ and the messages be M_0 and M_1. Choose a random bit b. We need to have $F_k(\mathsf{v}_k^*) \equiv 0 \bmod p$ for all $1 \leq k \leq h^*$. If this condition does not hold, then set flg to one. In the second case, the simulator uses the value of c to provide a proper encryption of M_b to $\mathcal{A}$ by computing $(M_b \times e(P_1, P_2)^c, cP, cV_1, \ldots, cV_{h^*})$. In the first case, it constructs a proper encryption of M_b in the following manner.

$$(M^b \times Z, C_1 = P_3, B_1 = J_1(\mathsf{v}_1^*) P_3, \ldots, B_{h^*} = J_{h^*}(\mathsf{v}_{h^*}^*) P_3).$$

We require B_j to be equal to $cV_j(\mathsf{v}_j^*)$ for $1 \leq j \leq h^*$. Recall that the definition of $V_j(v)$ is $V_j(v) = U_j' + \sum_{k=1}^{l} v_k U_k$. Using the definition of U_j' and the U_k's as defined in the setup by the simulator, we obtain, $cV_i = c(F_i(\mathsf{v}_i^*)P_2 + J_i(\mathsf{v}_i^*)P) = J_i(\mathsf{v}_i^*)cP = J_i(\mathsf{v}_i^*)P_3$. Here we use the fact, $F_i(\mathsf{v}_i^*) \equiv 0 \bmod p$. Hence, the quantities $B_1, \ldots, B_{h^*}$ are properly formed.

Guess: The adversary outputs a guess b' of b.

Game 2: This is a modification of Game 1 whereby the Z in Game 1 is now chosen to be a random element of G_2. This Z is used to mask the message M_b in the challenge ciphertext. Since Z is random, the first component of the challenge ciphertext is a random element of G_2 and provides no information to the adversary about b. Thus, $\mathsf{Pr}[X_2] = \frac{1}{2}$.

We have the following claim.

Claim:

$$|\mathsf{Pr}[X_1] - \mathsf{Pr}[X_2]| \leq \frac{\epsilon_{dbdh}}{\lambda} + \frac{\epsilon_{hibe}}{2}.$$

Proof : The change from Game 1 to Game 2 corresponds to an "indistinguishability" step in Shoup's tutorial [85] on such games. Usually, it is easy to bound the probability difference. In this case, the situation is complicated by the fact that there is a need to abort.

We show that it is possible to obtain an algorithm $\mathcal{B}$ for DBDH by extending Games 1 and 2. The extension of both the games is same and is described as follows. $\mathcal{B}$ takes as input a tuple (P, aP, bP, cP, Z) and sets up the HIBE protocol as in Game 1 (The setup of Games 1 and 2 are the same). The key extraction queries are answered and the challenge ciphertext is generated as in Game 1. If at any point of time flg is set to one by the game, then $\mathcal{B}$ outputs a random bit and aborts. This is because the query cannot be answered or the challenge ciphertext cannot be generated using the input tuple. At the end of the game, the adversary outputs the guess b'. $\mathcal{B}$ now goes through a separate abort stage as follows.

"Artificial Abort": The probability that $\mathcal{B}$ aborts in the query or challenge phases depends on the adversary's input. The goal of the artificial abort step is to make the probability of abort independent of the adversary's queries by ensuring that in all cases its probability of abort is the maximum possible. This is done by sampling the transcript of adversary's query and in certain cases aborting. The sampling procedure introduces the extra component $O(\epsilon_{hibe}^{-2} \ln(\epsilon_{hibe}^{-1})\lambda^{-1} \ln(\lambda^{-1}))$ into the simulator's runtime. (For details see [89, 77].) Here λ is a lower bound on the probability that $\mathcal{B}$ does not abort before entering the artificial abort stage. The expression for λ is obtained in Proposition 6.3.3. of Section 6.3.2.

Output: If $\mathcal{B}$ has not aborted up to this stage, then it outputs 1 if $b = b'$; else 0.

Note that if Z is real, then the adversary is playing Game 1 and if Z is random, then the adversary is playing Game 2. The time taken by the simulator in either Game 1 or 2 is clearly $t + \chi(\epsilon_{hibe})$. From this point, standard inequalities and probability calculations

establish the claim. We provide the details. Let Y_i be the event that the simulator outputs 1 in Game i, $i = 1, 2$. Then, we have

$$|\mathsf{Pr}[Y_1] - \mathsf{Pr}[Y_2]| \leq \epsilon_{dbdh}.$$

Let ab_i be the event that the simulator aborts in Game i, $i = 1, 2$. This includes both protocol and artificial abort. Following the analysis of [89] and [77], we have

$$\lambda - \frac{\lambda\epsilon}{2} \leq \mathsf{Pr}[\overline{\mathsf{ab}_i}|X_i], \mathsf{Pr}[\overline{\mathsf{ab}_i}|\overline{X_i}] \leq \lambda + \frac{\lambda\epsilon}{2}. \tag{6.3.5}$$

Here $\epsilon = \epsilon_{hibe}$ and λ is the lower bound on the probability of not abort up to the artificial abort stage (see Section 6.3.2).

$$\begin{aligned}
\mathsf{Pr}[Y_i] &= \mathsf{Pr}[Y_i \wedge (\mathsf{ab}_i \vee \overline{\mathsf{ab}_i})] \\
&= \mathsf{Pr}[(Y_i \wedge \mathsf{ab}_i) \vee (Y_i \wedge \overline{\mathsf{ab}_i})] \\
&= \mathsf{Pr}[Y_i \wedge \mathsf{ab}_i] + \mathsf{Pr}[Y_i \wedge \overline{\mathsf{ab}_i}] \\
&= \mathsf{Pr}[Y_i \mid \mathsf{ab}_i]\mathsf{Pr}[\mathsf{ab}_i] + \mathsf{Pr}[Y_i \mid \overline{\mathsf{ab}_i}]\mathsf{Pr}[\overline{\mathsf{ab}_i}] \\
&= \frac{1}{2}(1 - \mathsf{Pr}[\overline{\mathsf{ab}_i}]) + \mathsf{Pr}[X_i \mid \overline{\mathsf{ab}_i}]\mathsf{Pr}[\overline{\mathsf{ab}_i}] \\
&= \frac{1}{2}(1 - \mathsf{Pr}[\overline{\mathsf{ab}_i} \wedge (X_i \vee \overline{X_i})]) + \mathsf{Pr}[X_i \wedge \overline{\mathsf{ab}_i}] \\
&= \frac{1}{2} + \frac{1}{2}\left(\mathsf{Pr}[\overline{\mathsf{ab}_i}|X_i]\mathsf{Pr}[X_i] - \mathsf{Pr}[\overline{\mathsf{ab}_i}|\overline{X_i}]\mathsf{Pr}[\overline{X_i}]\right)
\end{aligned}$$

Now we need to do some manipulations with inequalities and for convenience we set $A_i = \mathsf{Pr}[\overline{\mathsf{ab}_i}|X_i]$, $B_i = \mathsf{Pr}[X_i]$ and $C_i = \mathsf{Pr}[\overline{\mathsf{ab}_i}|\overline{X_i}]$ and $D = \mathsf{Pr}[Y_1] - \mathsf{Pr}[Y_2]$. We have from (6.3.5)

$$\lambda - \frac{\lambda\epsilon}{2} \leq A_i, C_i \leq \lambda + \frac{\lambda\epsilon}{2}.$$

Also

$$2D = (A_1B_1 - C_1(1 - B_1)) - (A_2B_2 - C_2(1 - B_2)). \tag{6.3.6}$$

Since both B_1 and $(1 - B_1)$ are non-negative, we have

$$\begin{array}{rcccl}
B_i(\lambda - \frac{\lambda\epsilon}{2}) & \leq & A_iB_i & \leq & B_i(\lambda + \frac{\lambda\epsilon}{2}) \\
(1 - B_i)(-\lambda - \frac{\lambda\epsilon}{2}) & \leq & -C_i(1 - B_i) & \leq & (1 - B_i)(-\lambda + \frac{\lambda\epsilon}{2}).
\end{array}$$

Hence,

$$\lambda(2B_i - 1) - \frac{\lambda\epsilon}{2} \leq A_iB_i - C_i(1 - B_i) \leq \lambda(2B_i - 1) + \frac{\lambda\epsilon}{2}. \tag{6.3.7}$$

Putting $i = 1$ in (6.3.7), we obtain

$$\lambda(2B_1 - 1) - \frac{\lambda\epsilon}{2} \leq A_1B_1 - C_1(1 - B_1) \leq \lambda(2B_1 - 1) + \frac{\lambda\epsilon}{2}. \tag{6.3.8}$$

Multiplying (6.3.7) by -1 and putting $i = 2$ we obtain

$$-\lambda(2B_2 - 1) - \frac{\lambda\epsilon}{2} \leq -(A_2B_2 - C_2(1 - B_2)) \leq -\lambda(2B_2 - 1) + \frac{\lambda\epsilon}{2}. \tag{6.3.9}$$

Combining (6.3.6), (6.3.8) and (6.3.9) we get

$$2\lambda(B_1 - B_2) - \lambda\epsilon \leq 2D \leq 2\lambda(B_1 - B_2) + \lambda\epsilon. \tag{6.3.10}$$

This shows that $|\lambda(B_1 - B_2) - D| \leq \frac{\lambda\epsilon}{2}$. Now $|\lambda(B_1 - B_2)| - |D| \leq |\lambda(B_1 - B_2) - D| \leq \frac{\lambda\epsilon}{2}$. Note that $|D| = |\mathsf{Pr}[Y_1] - \mathsf{Pr}[Y_2]| \leq \epsilon_{dbdh}$ and recalling the values of B_1 and B_2, we have

$$|\mathsf{Pr}[X_1] - \mathsf{Pr}[X_2]| \leq \frac{\epsilon_{dbdh}}{\lambda} + \frac{\epsilon_{hibe}}{2}.$$

This completes the proof of the claim. ∎

Now we can complete the proof in the following manner.

$$\begin{aligned}
\epsilon_{hibe} &= \left|\mathsf{Pr}[X_0] - \frac{1}{2}\right| \\
&\leq |\mathsf{Pr}[X_0] - \mathsf{Pr}[X_2]| \\
&\leq |\mathsf{Pr}[X_0] - \mathsf{Pr}[X_1]| + |\mathsf{Pr}[X_1] - \mathsf{Pr}[X_2]| \\
&\leq \frac{\epsilon_{hibe}}{2} + \frac{\epsilon_{dbdh}}{\lambda}.
\end{aligned}$$

Rearranging the inequality gives the desired result. This completes the proof. ∎

6.3.2 Lower Bound on Not Abort

We require the following two independence results in obtaining the required lower bound. Similar independence results have been used earlier in [89, 77] in connection with IBE protocols. The situation for HIBE is more complicated than IBE and especially so since we reuse some of the public parameters over different levels of the HIBE. This makes the proofs more difficult. Our independence results are given in Proposition 6.3.1. and 6.3.2. and these subsume the results for the IBE of previous chapter. We provide complete proofs for these two propositions as well as a complete proof for the lower bound. The probability calculation for the lower bound is also more complicated compared to the IBE case.

PROPOSITION 6.3.1. *Let m be a prime and $L(\cdot)$ be as defined in (6.3.3). Let $\mathsf{v}_1, \ldots, \mathsf{v}_j$ be identities, i.e., each $\mathsf{v}_i = (\mathsf{v}_{i,1}, \ldots, \mathsf{v}_{i,l})$, with $\mathsf{v}_{i,k}$ to be an n/l-bit string (and hence $0 \leq \mathsf{v}_{i,k} \leq 2^{n/l} - 1$). Then*

$$\mathsf{Pr}\left[\bigwedge_{k=1}^{j} (L_k(\mathsf{v}_k) = 0)\right] = \frac{1}{m^j}.$$

The probability is over the independent and uniform random choices of $x'_1, \ldots, x'_j, x_1, \ldots, x_l$ *from* $\mathbb{Z}_m$*. Consequently, for any* $\theta \in \{1, \ldots, j\}$*, we have*

$$\mathsf{Pr}\left[L_\theta(\mathsf{v}_\theta) = 0 \,\middle|\, \bigwedge_{k=1, k\neq\theta}^{j} (L_k(\mathsf{v}_k) = 0)\right] = \frac{1}{m}.$$

Proof : Since $\mathbb{Z}_m$ forms a field, we can do linear algebra with vector spaces over $\mathbb{Z}_m$. The condition $\bigwedge_{k=1}^{j} (L_j(\mathsf{v}_j) = 0)$ is equivalent to the following system of equations over $\mathbb{Z}_m$.

$$\begin{array}{ccccccccccc} x'_1 & + & \mathsf{v}_{1,1}x_1 & + & \cdots & + & \mathsf{v}_{1,l}x_l & = & 0 \\ x'_2 & + & \mathsf{v}_{2,1}x_1 & + & \cdots & + & \mathsf{v}_{2,l}x_l & = & 0 \\ \cdots & \cdot & \cdots & \cdot & \cdots & \cdot & \cdots & \cdot & \cdot \\ x'_j & + & \mathsf{v}_{j,1}x_1 & + & \cdots & + & \mathsf{v}_{j,l}x_l & = & 0 \end{array}$$

This can be rewritten as

$$(x'_1, \ldots, x'_j, x_1, \ldots, x_l)A_{(j+l)\times(j+l)} \;=\; (0, \ldots, 0)_{1\times(j+l)}$$

where

$$A = \begin{bmatrix} I_j & O_{j\times l} \\ \mathsf{V}_{l\times j} & O_{l\times l} \end{bmatrix} \text{ and } \mathsf{V}_{l\times j} = \begin{bmatrix} \mathsf{v}_{1,1} & \cdots & \mathsf{v}_{j,1} \\ \cdots & \cdots & \cdots \\ \mathsf{v}_{1,l} & \cdots & \mathsf{v}_{j,l} \end{bmatrix};$$

I_j is the identity matrix of order j; O is the all zero matrix of the specified order. The rank of A is clearly j and hence the dimension of the solution space is l. Hence, there are m^l solutions in $(x'_1, \ldots, x'_j, x_1, \ldots, x_l)$ to the above system of linear equations. Since the variables $x'_1, \ldots, x'_j, x_1, \ldots, x_l$ are chosen independently and uniformly at random, the probability that the system of linear equations is satisfied for a particular choice of these variables is $m^l/m^{l+j} = 1/m^j$. This proves the first part of the result.

For the second part, note that we may assume $\theta = j$ by renaming the x''s if required. Then

$$\mathsf{Pr}\left[L_j(\mathsf{v}_j) = 0 \,\middle|\, \bigwedge_{k=1}^{j-1} (L_k(\mathsf{v}_k) = 0)\right] = \frac{\mathsf{Pr}\left[\bigwedge_{k=1}^{j} (L_k(\mathsf{v}_k) = 0)\right]}{\mathsf{Pr}\left[\bigwedge_{k=1}^{j-1} (L_k(\mathsf{v}_k) = 0)\right]} = \frac{m^{j-1}}{m^j} = \frac{1}{m}.$$

■

PROPOSITION 6.3.2. *Let* m *be a prime and* $L(\cdot)$ *be as defined in (6.3.3). Let* $\mathsf{v}_1, \ldots, \mathsf{v}_j$ *be identities, i.e., each* $\mathsf{v}_i = (\mathsf{v}_{i,1}, \ldots, \mathsf{v}_{i,l})$*, with* $\mathsf{v}_{i,k}$ *to be an* n/l*-bit string. Let* $\theta \in \{1, \ldots, j\}$ *and let* v'_θ *be an identity such that* $\mathsf{v}'_\theta \neq \mathsf{v}_\theta$*. Then*

$$\mathsf{Pr}\left[(L_\theta(\mathsf{v}'_\theta) = 0) \wedge \bigwedge_{k=1}^{j} (L_k(\mathsf{v}_k) = 0)\right] = \frac{1}{m^{j+1}}.$$

The probability is over the independent and uniform random choices of $x'_1, \ldots, x'_j, x_1, \ldots, x_l$ *from* $\mathbb{Z}_m$*. Consequently, we have*

$$\mathsf{Pr}\left[L_\theta(\mathsf{v}'_\theta) = 0 \,\middle|\, \bigwedge_{k=1}^{j} (L_k(\mathsf{v}_k) = 0)\right] = \frac{1}{m}.$$

Proof : The proof is similar to the proof of Proposition 6.3.1. Without loss of generality, we may assume that $\theta = j$, since otherwise we may rename variables to achieve this. The condition $(L_\theta(\mathsf{v}'_\theta) = 0) \wedge \bigwedge_{k=1}^{j} (L_k(\mathsf{v}_k) = 0)$ is equivalent to a system of linear equations $xA = 0$ over $\mathbb{Z}_m$. In this case, the form of A is the following.

$$A = \begin{bmatrix} I_j & c^T & O_{j\times l} \\ V_{l\times j} & (\mathsf{v}'_j)^T & O_{l\times l} \end{bmatrix}$$

where $c = (0, \ldots, 0, 1)$; c^T denotes the transpose of c and $(\mathsf{v}'_j)^T$ is the transpose of v'_j. The first j columns of A are linearly independent. The $(j+1)$th column of A is clearly linearly independent of the first $(j-1)$ columns. We have $\mathsf{v}_j \neq \mathsf{v}'_j$. Since each component of both v_j and v'_j is less than $2^{n/l}$ and $m > 2^{n/l}$, we have $\mathsf{v}_j \not\equiv \mathsf{v}'_j \bmod m$. Using this, it is not difficult to see that the first $(j+1)$ columns of A are linearly independent and hence the rank of A is $(j+1)$. (Note that if $m \leq 2^{n/l}$, then it is possible to have $\mathsf{v}_j \neq \mathsf{v}'_j$ but $\mathsf{v}_j \equiv \mathsf{v}'_j \bmod m$. Then the jth and $(j+1)$th columns of A are equal and the rank of A is j.) Consequently, the dimension of the solution space is $l-1$ and there are m^{l-1} solutions in $(x'_1, \ldots, x'_j, x_1, \ldots, x_l)$ to the system of linear equations. Since the x''s and the x's are chosen independently and uniformly at random from $\mathbb{Z}_m$, the probability of getting a solution is $m^{l-1}/m^{l+j} = 1/m^{j+1}$. This proves the first part of the result. The proof of the second part is similar to that of Proposition 6.3.1.. ∎

PROPOSITION 6.3.3. *The probability that the simulator in the proof of Theorem 6.3.1. does not abort before the artificial abort stage is at least* $\frac{1}{2(2\sigma(\mu_l+1))^h}$.

Proof : We consider the simulator in the proof of Theorem 6.3.1. Up to the artificial abort stage, the simulator could abort on either a key extraction query or in the challenge stage. Let abort be the event that the simulator aborts before the artificial abort stage. For $1 \leq i \leq q$, let E_i denote the event that the simulator does not abort on the ith key extraction query and let C be the event that the simulator does not abort in the challenge stage. We have

$$\begin{aligned} \mathsf{Pr}[\overline{\mathsf{abort}}] &= \mathsf{Pr}\left[\left(\bigwedge_{i=1}^{q} E_i\right) \wedge C\right] \\ &= \mathsf{Pr}\left[\left(\bigwedge_{i=1}^{q} E_i\right) | C\right] \mathsf{Pr}[C] \end{aligned}$$

$$
\begin{aligned}
&= \left(1 - \mathsf{Pr}\left[\left(\bigvee_{i=1}^{q} \neg E_i\right) | C\right]\right) \mathsf{Pr}[C] \\
&\geq \left(1 - \sum_{i=1}^{q} \mathsf{Pr}\left[\neg E_i \,|C\right]\right) \mathsf{Pr}[C].
\end{aligned}
$$

We first consider the event C. Suppose the challenge identity is $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_{h^*}^*)$. Event C holds if and only if $F_j(\mathsf{v}_j^*) \equiv 0 \bmod p$ for $1 \leq j \leq h^*$. Recall that by choice of p, we can assume $F_j(\mathsf{v}_j^*) \equiv 0 \bmod p$ if and only if $x'_j + \sum_{k=1}^{l} x_k \mathsf{v}_{j,k} = mk_j$. Hence,

$$
\mathsf{Pr}[C] \;=\; \mathsf{Pr}\left[\bigwedge_{j=1}^{h^*} \left(x'_j + \sum_{k=1}^{l} x_k \mathsf{v}_{j,k} = mk_j\right)\right]. \tag{6.3.11}
$$

For $1 \leq j \leq h^*$ and $0 \leq i \leq \mu_l$, denote the event $x'_j + \sum_{k=1}^{l} x_k \mathsf{v}_{j,k} = mi$ by $A_{j,i}$ and the event $k_j = i$ by $B_{j,i}$. Also, let $C_{j,i}$ be the event $A_{j,i} \wedge B_{j,i}$.

Note that the event $\bigvee_{i=0}^{\mu_l} A_{j,i}$ is equivalent to the condition $x'_j + \sum_{k=1}^{l} x_k \mathsf{v}_{j,k} \equiv 0 \bmod m$ and hence equivalent to the condition $L_j(\mathsf{v}_j) = 0$. Since k_j is chosen uniformly at random from the set $\{0, \ldots, \mu_l\}$, we have $\mathsf{Pr}[B_{j,i}] = 1/(1+\mu_l)$ for all j and i. The events $B_{j,i}$'s are independent of each other and also independent of the $A_{j,i}$'s. We have

$$
\begin{aligned}
&\mathsf{Pr}\left[\bigwedge_{j=1}^{h^*} \left(x'_j + \sum_{k=1}^{l} x_k \mathsf{v}_{j,k} = mk_j\right)\right] \\
=\;& \mathsf{Pr}\left[\bigwedge_{j=1}^{h^*} \left(\bigvee_{i=0}^{\mu_l} C_{j,i}\right)\right] \\
=\;& \mathsf{Pr}\left[\bigvee_{i_1,\ldots,i_{h^*} \in \{0,\ldots,\mu_l\}} (C_{1,i_1} \wedge \cdots \wedge C_{h^*,i_{h^*}})\right] \\
=\;& \mathsf{Pr}\left[\bigvee_{i_1,\ldots,i_{h^*} \in \{0,\ldots,\mu_l\}} (A_{1,i_1} \wedge B_{1,i_1} \wedge \cdots \wedge A_{h^*,i_{h^*}} \wedge B_{h^*,i_{h^*}})\right] \\
=\;& \sum_{i_1,\ldots,i_{h^*} \in \{0,\ldots,\mu_l\}} \mathsf{Pr}\left[A_{1,i_1} \wedge B_{1,i_1} \wedge \cdots \wedge A_{h^*,i_{h^*}} \wedge B_{h^*,i_{h^*}}\right] \\
=\;& \sum_{i_1,\ldots,i_{h^*} \in \{0,\ldots,\mu_l\}} \mathsf{Pr}\left[A_{1,i_1} \wedge \cdots \wedge A_{h^*,i_{h^*}}\right] \times \mathsf{Pr}\left[B_{1,i_1} \wedge \cdots \wedge B_{h^*,i_{h^*}}\right] \\
=\;& \frac{1}{(1+\mu_l)^{h^*}} \sum_{i_1,\ldots,i_{h^*} \in \{0,\ldots,\mu_l\}} \mathsf{Pr}\left[A_{1,i_1} \wedge \cdots \wedge A_{h^*,i_{h^*}}\right] \\
=\;& \frac{1}{(1+\mu_l)^{h^*}} \mathsf{Pr}\left[\bigvee_{i_1,\ldots,i_{h^*} \in \{0,\ldots,\mu_l\}} (A_{1,i_1} \wedge \cdots \wedge A_{h^*,i_{h^*}})\right]
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{(1+\mu_l)^{h^*}} \mathsf{Pr}\left[\bigwedge_{j=1}^{h^*}\left(\bigvee_{i=0}^{\mu_l} A_{j,i}\right)\right] \\
&= \frac{1}{(1+\mu_l)^{h^*}} \mathsf{Pr}\left[\bigwedge_{j=1}^{h^*}\left(L_j(\mathsf{v}_j)=0\right)\right] \\
&= \frac{1}{(m(1+\mu_l))^{h^*}}
\end{aligned}
$$

The last equality follows from Proposition 6.3.1.

Now we turn to bounding $\mathsf{Pr}[\neg E_i|C]$. For simplicity of notation, we will drop the subscript i from E_i and consider the event E that the simulator does not abort on a particular key extraction query on an identity $(\mathsf{v}_1, \ldots, \mathsf{v}_j)$. By the simulation, the event $\neg E$ implies that $L_i(\mathsf{v}_i) = 0$ for all $1 \leq i \leq j$. This holds even when the event is conditioned under C. Thus, we have $\mathsf{Pr}[\neg E|C] \leq \mathsf{Pr}[\wedge_{i=1}^{j} L_i(\mathsf{v}_i) = 0|C]$. The number of components in the challenge identity is h^* and now two cases can happen:
$j \leq h^*$: By the protocol constraint (a prefix of the challenge identity cannot be queried to the key extraction oracle), we must have a θ with $1 \leq \theta \leq j$ such that $\mathsf{v}_\theta \neq \mathsf{v}^*_\theta$.
$j > h^*$: In this case, we choose $\theta = h^* + 1$.

Now we have

$$
\mathsf{Pr}[\neg E|C] \leq \mathsf{Pr}\left[\bigwedge_{i=1}^{j} L_i(\mathsf{v}_i) = 0|C\right] \leq \mathsf{Pr}[L_\theta(\mathsf{v}_\theta) = 0|C] = \mathsf{Pr}\left[L_\theta(\mathsf{v}_\theta) = 0|\bigwedge_{i=1}^{h^*} L_i(\mathsf{v}^*_i) = 0\right] = 1/m.
$$

The last equality follows from an application of either Proposition 6.3.1. or Proposition 6.3.2. according as whether $j > h^*$ or $j \leq h^*$. Substituting this in the bound for $\mathsf{Pr}[\overline{\mathsf{abort}}]$ we obtain

$$
\begin{aligned}
\mathsf{Pr}[\overline{\mathsf{abort}}] &\geq \left(1 - \sum_{i=1}^{q} \mathsf{Pr}\left[\neg E_i\,|C\right]\right) \mathsf{Pr}[C]. \\
&\geq \left(1 - \frac{q}{m}\right) \frac{1}{(m(\mu_l+1))^{h^*}} \\
&\geq \left(1 - \frac{q}{m}\right) \frac{1}{(m(\mu_l+1))^{h}} \\
&\geq \frac{1}{2} \times \frac{1}{(2\sigma(\mu_l+1))^{h}}.
\end{aligned}
$$

We use $h \geq h^*$ and $2q \leq \sigma < m < 2\sigma$ to obtain the inequalities. This completes the proof. ■

6.4 Conclusion

In this chapter, we have presented a construction of a HIBE which builds upon the previous IBE protocols. The HIBE is secure in the full model without random oracle. The number

of public parameters is significantly less compared to previous suggestion. The main open problem in the construction of HIBE protocols secure in the full model is to avoid or control the security degradation which is exponential in the number of levels of the HIBE.

Chapter 7

Generalization of the Selective-ID Security Model

7.1 Introduction

In this chapter, we generalize the selective-ID model and introduce two new models of security for HIBE protocols. The basic idea is to modify the security game so as to allow the adversary to commit to a set of identities (instead of one identity in the sID model) before set-up. During the game, the adversary can execute key extraction queries on any identity not in the committed set. In the challenge stage, the challenge identity is chosen by the adversary from among the set that it has previously committed to.

For IBE, this is a strict generalization of the sID model, since we can get the sID model by enforcing the size of the committed set of identities to be one. On the other hand, for HIBE, there are two ways to view this generalization leading to two different security models $\boldsymbol{\mathcal{M}}_1$ and $\boldsymbol{\mathcal{M}}_2$.

In $\boldsymbol{\mathcal{M}}_1$, the adversary commits to a set $\mathcal{I}^*$. It can then ask for the private key of any identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ as long as all the v_is are not in $\mathcal{I}^*$. Further, during the challenge stage, it has to submit an identity all of whose components are in $\mathcal{I}^*$. If we restrict the adversary to only single component identities (i.e., we are considering only the IBE protocols), then this is a clear generalization of the sID model for IBE. On the other hand, in the case of HIBE, we cannot fix the parameters of this model to obtain the sID model for HIBE.

The second model, $\boldsymbol{\mathcal{M}}_2$, is an obvious generalization of the sID model for HIBE. In this case, the adversary specifies j sets $\mathcal{I}_1^*, \ldots, \mathcal{I}_j^*$. Then it can ask for private key of any identity v as long as there is an i such that the ith component of v is not in $\mathcal{I}_i^*$. In the challenge stage, the adversary has to submit an identity such that for all i, the ith component of the identity is in $\mathcal{I}_i^*$.

Even though $\boldsymbol{\mathcal{M}}_2$ generalizes the sID model for HIBE, we think $\boldsymbol{\mathcal{M}}_1$ is also an appropriate model for a HIBE protocol. The adversary would be specifying a set of "sensitive" keywords

to be $\mathcal{I}^*$. It can then ask for the private key of any identity as long as one component of the identity is not sensitive and in the challenge stage has to submit an identity all of whose components are sensitive. The added flexibility in $\boldsymbol{\mathcal{M}}_2$ is that the adversary can specify different sets of sensitive keywords for the different levels of HIBE. In practice, this flexibility might not be required since keywords like root, admin, dba, etcetera will be sensitive for all levels.

We present two constructions of HIBE denoted by $\mathcal{H}_1$ and $\mathcal{H}_2$. $\mathcal{H}_1$ is proved to be secure in the model $\boldsymbol{\mathcal{M}}_1$ under the DBDH assumption while $\mathcal{H}_2$ is proved to be secure in the model $\boldsymbol{\mathcal{M}}_2$ also under the DBDH assumption. Our constructions and proofs of security are very similar to that of the Boneh-Boyen HIBE (BB-HIBE) discussed in Section 3.2.1. The actual technical novelty in the proofs is the use of a polynomial, which in the case of the BB-HIBE is of degree one. The use of an appropriate polynomial of degree greater than one allows us to prove security in the more general models $\boldsymbol{\mathcal{M}}_1$ and $\boldsymbol{\mathcal{M}}_2$. However, this flexibility comes at a cost. In both $\mathcal{H}_1$ and $\mathcal{H}_2$, the number of required scalar multiplications increases linearly with the size of the committed set of identities.

Multiple receiver IBE (MR-IBE) is an interesting concept which was introduced by Baek, Safavi-Naini and Susilo [4]. In an MR-IBE, an encryptor can encrypt a message in such a way that any one of a set of identities can decrypt the message. A trivial way to achieve this is to separately encrypt the message several times. It turns out that the efficiency can be improved. A more efficient construction of MR-IBE was presented in [4]. The proof of security was in the sID model *using* random oracle.

We show that the HIBE $\mathcal{H}_1$ or $\mathcal{H}_2$ when restricted to IBE can be easily modified to obtain an efficient MR-IBE. Our MR-IBE is proved to be secure in the sID model *without* random oracle and to the best of our knowledge this is the first of such kind.

7.2 Another Look at Security Model

We have seen in Chapter 2 that the security model for HIBE is defined as an interactive game between an adversary and a simulator. Both the adversary and the simulator are modeled as probabilistic algorithms. Currently, there are two security models for HIBE – the selective-ID (sID) model and the full model. We will be interested in defining two new security models. We present the description of the interactive game in a manner which will help in obtaining a unified view of the sID, full and the new security models that we define.

In the game, the adversary is allowed to query two oracles – a decryption oracle $\mathcal{O}_d$ and a key-extraction oracle $\mathcal{O}_k$. The game has several stages.

Adversary's Commitment: In this stage, the adversary commits to two sets $\mathcal{S}_1$ and $\mathcal{S}_2$ of identities. The commitment has the following two consequences.

1. The adversary is not allowed to query $\mathcal{O}_k$ on any identity in $\mathcal{S}_1$ or on a prefix of any identity in $\mathcal{S}_1$.

2. In the challenge stage, the adversary has to choose one of the identities from the set $\mathcal{S}_2$.

There is a technical difficulty here. Note that the adversary has to commit to a set of identities even before the HIBE protocol has been set-up. On the other hand, the identity space is specified by the set-up algorithm of the HIBE protocol. In effect, this means that the adversary has to commit to identities even before it knows the set of identities. Clearly, this is not possible.

One possible way out is to allow the adversary to commit to binary strings and later when the set-up program has been executed, these binary strings are mapped to identities using a collision resistant hash functions. Another solution is to run the set-up program in two phases. In the first phase, the identity space is specified and is made available to the adversary; then the adversary commits to $\mathcal{S}_1$ and $\mathcal{S}_2$; and after obtaining $\mathcal{S}_1$ and $\mathcal{S}_2$ the rest of the set-up program is executed.

The above two approaches are not necessarily equivalent and may have different security consequences. On the other hand, note that if $\mathcal{S}_1 = \emptyset$ and $\mathcal{S}_2$ is the set of all identities (as is true in the full model), then this technical difficulty does not arise.

Set-Up: The simulator sets up the HIBE protocol and provides the public parameters to the adversary and keeps the master key to itself. Note that at this stage, the simulator knows $\mathcal{S}_1, \mathcal{S}_2$ and could possibly set-up the HIBE based on this knowledge. However, while doing this, the simulator must ensure that the probability distribution of the public parameters remains the same as in the specification of the actual HIBE protocol.

Phase 1: The adversary makes a finite number of queries where each query is addressed either to $\mathcal{O}_d$ or to $\mathcal{O}_k$. In a query to $\mathcal{O}_d$, it provides the ciphertext as well as the identity under which it wants the decryption. The simulator returns either the corresponding message or bad if the ciphertext is malformed. Similarly, in a query to $\mathcal{O}_k$, it asks for the private key of the identity it provides. This identity cannot be an element of $\mathcal{S}_1$ and neither can it be a prefix of any element in $\mathcal{S}_1$. Further, the adversary is allowed to make these queries adaptively, i.e., any query may depend on the previous queries as well as their answers.

Certain queries are useless and we will assume that the adversary does not make such queries. For example, if an adversary has queried $\mathcal{O}_k$ on any identity, then it is not allowed to present the same identity to $\mathcal{O}_d$ as part of a decryption query. The rationale is that since the adversary already has the private key, it can itself decrypt the required ciphertext.

Challenge: The adversary provides the simulator with an identity $\mathsf{v}^* \in \mathcal{S}_2$ and two messages M_0 and M_1. There is a restriction that the simulator should not have queried $\mathcal{O}_k$ for the private key of v^* or for the private key of any prefix of v^* in Phase 1. The simulator randomly chooses a $\gamma \in \{0, 1\}$ and returns the encryption of M_γ under v^* to the adversary.

Phase 2: The adversary issues additional queries just as in Phase 1 with the following restrictions. It cannot ask $\mathcal{O}_d$ for the decryption of C^* under v^*; cannot ask $\mathcal{O}_k$ for the private key of any prefix of an identity in $\mathcal{S}_1$; and cannot make any useless query.

Guess: The adversary outputs a guess γ' of γ.

The adversary's success is measured as defined in Section 2.5

7.2.1 Full Model

Suppose $\mathcal{S}_1 = \emptyset$ and $\mathcal{S}_2$ is the set of all identities. By the rules of the game, the adversary is not allowed to query $\mathcal{O}_k$ on any identity in $\mathcal{S}_1$. Since $\mathcal{S}_1$ is empty, this means that the adversary is actually allowed to query $\mathcal{O}_k$ on any identity. Further, since $\mathcal{S}_2$ is the set of all identities, in the challenge stage, the adversary is allowed to choose any identity. In effect, this means that the adversary does not really commit to anything before set-up and hence, in this case, the commitment stage can be done away with. This particular choice of $\mathcal{S}_1$ and $\mathcal{S}_2$ is called the full model (defined in Section 2.5 and is currently believed to be the most general notion of security for HIBE.

Note that the challenge stage restrictions as well as the restrictions in Phase 2 still apply.

7.2.2 Selective-ID Model

Let $\mathcal{S}_1 = \mathcal{S}_2$ be a singleton set. This means that the adversary commits to one particular identity; does not ask for a private key of any of its prefixes; and in the challenge phase is given the encryption of M_γ under this particular identity. This model is significantly weaker than the full model and is called the selective-ID model (defined in Section 2.5.3.

7.2.3 New Security Models

We introduce two new security models by suitably defining the sets $\mathcal{S}_1$ and $\mathcal{S}_2$. In our new models, (as well as the sID model), we have $\mathcal{S}_1 = \mathcal{S}_2$. (Note that in the full model, $\mathcal{S}_1 = \overline{\mathcal{S}_2}$.)

Model $\boldsymbol{\mathcal{M}_1}$: Let $\mathcal{I}^*$ be a set. Define $\mathcal{S}_1 = \mathcal{S}_2$ to be the set of all tuples $(\mathsf{v}_1, \ldots, \mathsf{v}_j)$, such that each $\mathsf{v}_i \in \mathcal{I}^*$. If the HIBE is of maximum depth h, then $1 \leq j \leq h$. The length of the target identity tuple is *not* fixed by the adversary in the commit phase.

Let us now see what this means. In the commit phase, the adversary commits to a set $\mathcal{I}^*$; never asks for a private key of an identity all of whose components are in $\mathcal{I}^*$; and during the challenge phase presents an identity all of whose components are in $\mathcal{I}^*$.

Consider the case of IBE, i.e., $h = 1$, which means that only single component identities are allowed. Then, we have $\mathcal{S}_1 = \mathcal{S}_2 = \mathcal{I}^*$. Let $|\mathcal{I}^*| = n$. If we put $n = 1$, then we obtain the sID model for IBE as discussed in Section 7.2.2. In other words, for IBE protocol, $\boldsymbol{\mathcal{M}_1}$ is a strict generalization of sID model.

If $h > 1$, then we have proper HIBE. In this case, $\boldsymbol{\mathcal{M}}_1$ differs fundamentally from the sID model.

1. In the sID model, the adversary is allowed to query $\mathcal{O}_k$ on a permutation of the challenge identity. This is not allowed in $\boldsymbol{\mathcal{M}}_1$.

2. In the sID model, the length of the challenge identity is fixed by the adversary in the commit phase. On the other hand, in $\boldsymbol{\mathcal{M}}_1$, the adversary is free to choose this length (to be between 1 and h) in the challenge stage itself.

In the case of HIBE, model $\boldsymbol{\mathcal{M}}_1$ is no longer a strict generalization of the usual sID model for HIBE. We cannot restrict the parameters of the model $\boldsymbol{\mathcal{M}}_1$ in any manner and obtain the sID model for HIBE. Thus, in this case, $\boldsymbol{\mathcal{M}}_1$ must be considered to be a new model.

Model $\boldsymbol{\mathcal{M}}_2$: Let $\mathcal{I}_1^*, \ldots, \mathcal{I}_u^*$ be sets and $|\mathcal{I}_j^*| = n_j$ for $1 \leq j \leq u$. We set

$$\mathcal{S}_1 = \mathcal{S}_2 = \mathcal{I}_1^* \times \cdots \times \mathcal{I}_u^*.$$

If the maximum depth of the HIBE is h, then $1 \leq u \leq h$.

In this model, for $1 \leq j \leq u$, the adversary is not allowed to obtain a private key for an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ such that $\mathsf{v}_i \in \mathcal{I}_i^*$ for all $1 \leq i \leq j$. Further, the challenge identity is a tuple $(\mathsf{v}_1^*, \ldots, \mathsf{v}_u^*)$, with $\mathsf{v}_i \in \mathcal{I}_i^*$ for all $1 \leq i \leq u$. Like the sID model, the length of the challenge identity is fixed by the adversary in the commit phase.

This model is a strict generalization of the sID model for HIBE. This can be seen by setting $n_1 = \cdots = n_h = 1$, i.e., setting $\mathcal{I}_1^*, \ldots, \mathcal{I}_h^*$ to be singleton sets. On the other hand, $\boldsymbol{\mathcal{M}}_2$ and $\boldsymbol{\mathcal{M}}_1$ are not comparable due to at least two reasons.

1. In $\boldsymbol{\mathcal{M}}_1$, the length of the challenge identity can vary, while in $\boldsymbol{\mathcal{M}}_2$, the length is fixed in the commit phase.

2. In $\boldsymbol{\mathcal{M}}_2$, it may be possible for the adversary to obtain the private key for a permutation of the challenge identity, which is not allowed in $\boldsymbol{\mathcal{M}}_1$.

These two reasons are similar to the reasons for the difference between sID and $\boldsymbol{\mathcal{M}}_1$.

Parametrizing the models: A HIBE may have a bound on the maximum number of levels that can be supported. The corresponding security model also has the same restriction. For example, a HIBE of at most h levels will be called h-sID secure if it is secure in the sID model. The models $\boldsymbol{\mathcal{M}}_1$ and $\boldsymbol{\mathcal{M}}_2$ have additional parameters.

For the case of $\boldsymbol{\mathcal{M}}_1$, there is a single parameter n, which specifies the size of $\mathcal{I}^*$, the set which the adversary specifies in the commit phase. In this case, we will talk of (h, n)-$\boldsymbol{\mathcal{M}}_1$ security for an HIBE. Similarly, for $\boldsymbol{\mathcal{M}}_2$, we will talk of $(h, n_1, \ldots, n_h)$-$\boldsymbol{\mathcal{M}}_2$ security. Note that $(h, 1 \ldots, 1)$-$\boldsymbol{\mathcal{M}}_2$ model is same as the h-sID model.

7.3 Interpreting Security Models

The full security model is currently believed to provide the most general security model for HIBE. In other words, it provides any entity (having any particular identity) in the HIBE with the most satisfactory security assurance that the entity can hope for. The notion of security based on an appropriate adversarial game is adapted from the corresponding notion for public key encryption and the security assurance provided in that setting also applies to the HIBE setting. The additional consideration is that of identity and the key extraction queries to $\mathcal{O}_k$. We may consider the identity present during the challenge stage to be a target identity. In other words, the adversary wishes to break the security of the corresponding entity. In the full model, the target identity can be any identity, with the usual restriction that the adversary does not know the private key corresponding to this identity or one of its prefixes.

From the viewpoint of an individual entity $\mathbf{e}$ in the HIBE structure, the adversary's behavior appears to be the following. The adversary can possibly corrupt any entity in the structure, but as long as it is not able to corrupt that particular entity $\mathbf{e}$ or one of its ancestors, then it will not be able to succeed in an attack where the target identity is that of $\mathbf{e}$. In other words, obtaining the private keys corresponding to the other identities does not help the adversary. Intuitively, that is the maximum protection that any entity $\mathbf{e}$ can expect from the system.

Let's reflect on the sID model. In this model, the adversary commits to an identity even before the set-up of the HIBE is done. The actual set-up can depend on the identity in question. Now consider the security assurance obtained by an individual entity $\mathbf{e}$. Entity $\mathbf{e}$ can be convinced that if the adversary had targeted its identity and then the HIBE structure was set-up, in that case the adversary will not be successful in attacking it. Alternatively, $\mathbf{e}$ can be convinced that the HIBE structure can be set-up so as to protect it. Inherently, the sID model assures that the HIBE structure can be set-up to protect any identity, but only one.

Suppose that a HIBE structure which is secure in the sID model has already been set-up. It has possibly been set-up to protect one particular identity. The question now is what protection does it offer to entities with other identities? The model does not assure that other identities will be protected. Of course, this does not mean that other identities are vulnerable. The model simply does not say anything about these identities.

The system designer's point of view also needs to be considered. While setting up the HIBE structure, the designer needs to ensure security. The HIBE is known to be secure in the sID model and hence has a proof of security. The designer will play the role of the simulator in the security game. In the game, the adversary commits to an identity and then the HIBE is set-up so as to protect this identity. However, since the actual set-up has not been done, there is no real adversary and hence no real target identity. Thus, the designer has to assume that the adversary will probably be targeting some sensitive identity like root. The designer can then set-up the HIBE so as to protect this identity. However, once the HIBE has been set-up, the designer cannot say anything about the security of other possible

sensitive identities like sysadmin. This is a limitation of the sID model.

It has been observed in [17] that a generic conversion from an IBE protocol secure in the selective-ID model to a protocol secure in the full model suffers from a security degradation by a factor of 2^ℓ, where identities are ℓ-bit strings. This also indicates the inadequacy of the selective-ID model.

This brings us to the generalization of the sID model that we have introduced. First consider the model $\boldsymbol{\mathcal{M}}_1$ as it applies to IBE. In this model, the designer can assume that the adversary will possibly attack one out of a set of sensitive identities like {root, admin, dba, sysadmin}. It can then set-up the IBE so as to protect this set of identities. This offers better security than the sID model.

Now consider the model $\boldsymbol{\mathcal{M}}_1$ as it applies to HIBE. In this case, the set $\mathcal{I}^*$ can be taken to be a set of sensitive keywords such as {root, admin, dba, sysadmin}. The adversary is not allowed to obtain private keys corresponding to identities all of whose components lie in $\mathcal{I}^*$. For the above example, the adversary cannot obtain the private key of (root, root), or (admin, root, dba). On the other hand, it is allowed to obtain keys corresponding to identities like (root, abracadabra). Thus, some of the components of the identities (on which key extraction query is made) may be in $\mathcal{I}^*$; as long as all of them are not in $\mathcal{I}^*$, the adversary can obtain the private key. On the other hand, all the components of the target identity have to be sensitive keywords, i.e., elements of $\mathcal{I}^*$. Clearly, model $\boldsymbol{\mathcal{M}}_1$ provides an acceptable security notion for HIBE.

As mentioned earlier, a major difference of $\boldsymbol{\mathcal{M}}_1$ with sID is that in sID the adversary is allowed to obtain a private key for a permutation of the challenge identity, whereas this is not allowed in $\boldsymbol{\mathcal{M}}_1$. We point out that it is possible for a particular HIBE to be secure in both sID and $\boldsymbol{\mathcal{M}}_1$. An example will be provided later. Thus, one may choose to obtain the good features of both sID and $\boldsymbol{\mathcal{M}}_1$.

The model $\boldsymbol{\mathcal{M}}_2$ is a clear generalization of the usual sID model for HIBE. The adversary fixes the sensitive keywords for each level of the HIBE up to the level it wishes to attack. It cannot make a key extraction query on an identity of depth j, such that for $1 \leq i \leq j$, the ith component of the identity is among the pre-specified sensitive keywords for the ith level of the HIBE. Further, the target identity must be such that each of its component is a sensitive keyword for the corresponding HIBE level. As mentioned earlier, by fixing exactly one keyword for each level of the HIBE, we obtain the sID model.

The protocols like Waters IBE, IBE-SPP(ℓ) of Chapter 5 and HIBE-spp of Chapter 6 which offer full model security suffer from security degradation. On the other hand, protocols such as BB-HIBE and BBG-HIBE discussed in Chapter 3 which are secure in the selective-ID model have no security degradation. Thus, one can work with significantly smaller size groups while implementing the later protocols compared to the former protocols. The protocols that are described in this chapter have no security degradation. Hence, the group sizes used for implementing selective-ID protocols can be used for implementing the protocols secure in $\boldsymbol{\mathcal{M}}_1$ and $\boldsymbol{\mathcal{M}}_2$.

7.4 Constructions

We present several HIBE protocols which are proved to be secure in different models. In this section, we provide only the constructions. The security proofs are provided later.

The underlying groups G_1, G_2 and the pairing map $e(,)$ will be required by all the HIBE protocols. The set-up procedure of each HIBE will generate these groups based on the security parameter. The maximum depth of a HIBE will be denoted by h. In each of the HIBEs below, we will have P_1 and P_2 as public parameters which are not directly required. Instead, one may keep $e(P_1, P_2)$ in the public parameter which will save the pairing computation during encryption.

The components of identities are elements of $\mathbb{Z}_p$. Alternatively, if these are bit strings, then (as is standard) they will be hashed using a collision resistant hash function into $\mathbb{Z}_p$.

7.4.1 HIBE $\mathcal{H}_1$

Set-Up: The identity space consists of all tuples $(\mathsf{v}_1, \ldots, \mathsf{v}_j)$, $j \leq h$, where each $\mathsf{v}_i \in \mathbb{Z}_p$. The message space is G_2. The ciphertext corresponding to an identity $(\mathsf{v}_1, \ldots, \mathsf{v}_j)$ is a tuple $(A, B, C_1, \ldots, C_j)$, where $A \in G_2$ and $B, C_1, \ldots, C_j \in G_1$.

Randomly choose $\alpha \in \mathbb{Z}_p$ and set $P_1 = \alpha P$. Randomly choose P_2, $P_{3,1}, \ldots, P_{3,h}$, $Q_1, \ldots, Q_n$ from G_1 where n is a parameter. The public parameters are

$$(P, P_1, P_2, P_{3,1}, \ldots, P_{3,h}, Q_1, \ldots, Q_n)$$

and the master secret key is αP_2.

Notation: For any $y \in \mathbb{Z}_p$ define

$$V_i(y) = y^n Q_n + \cdots + yQ_1 + P_{3,i}.$$

Let $(\mathsf{v}_1, \ldots, \mathsf{v}_j)$ be an identity. We write V_i for $V_i(\mathsf{v}_i)$.

Key-Gen: The private key $d_{\mathsf{v}} = (d_0, d_1, \ldots, d_j)$ corresponding to an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ is defined to be

$$(d_0, d_1, \ldots, d_j) = (\alpha P_2 + r_1 V_1 + \ldots + r_j V_j, r_1 P, \ldots, r_j P)$$

where $r_1, \ldots, r_j$ are random elements of $\mathbb{Z}_p$. Key delegation can be done in the following manner. Let $(d'_0, d'_1, \ldots, d'_{j-1})$ be the private key corresponding to the identity $(\mathsf{v}_1, \ldots, \mathsf{v}_{j-1})$. Then $(d_0, d_1, \ldots, d_j)$ is obtained as follows. Choose a random r_j from $\mathbb{Z}_p$ and define

$$\begin{array}{lcll} d_0 & = & d'_0 + r_j V_j; & \\ d_i & = & d'_i & \text{for } 1 \leq i \leq j-1; \\ d_j & = & r_j P. & \end{array}$$

This provides a proper private key corresponding to the identity $(\mathsf{v}_1, \ldots, \mathsf{v}_j)$.

Encrypt: Suppose a message M is to be encrypted under the identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$. Choose a random $t \in \mathbb{Z}_p$. The ciphertext is $(A, B, C_1, \ldots, C_j)$, where

$$A = M \times e(P_1, P_2)^s; \quad B = sP; \quad C_i = sV_i, \text{ for } 1 \leq i \leq j.$$

Decrypt: Suppose $(A, B, C_1, \ldots, C_j)$ is to be decrypted using the private key $(d_0, d_1, \ldots, d_j)$ corresponding to the identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$. Compute

$$\begin{aligned} A \times \frac{\prod_{i=1}^{j} e(d_i, C_i)}{e(d_0, B)} &= M \times e(P_1, P_2)^s \frac{\prod_{i=1}^{j} e(r_i P, sV_i)}{e(\alpha P_2 + \sum_{i=1}^{j} r_i V_i, sP)} \\ &= M \times e(P_1, P_2)^s \times \frac{1}{e(P_1, P_2)^s} \times \frac{\prod_{i=1}^{j} e(r_i P, sV_i)}{e(\sum_{i=1}^{j} r_i V_i, sP)} \\ &= M. \end{aligned}$$

Unbounded Depth HIBE:

It is possible to modify $\mathcal{H}_1$ to obtain a HIBE which is secure in model $\boldsymbol{\mathcal{M}}_1$ and which supports key delegation over any number of levels. The required modifications are as follows.

- The public parameters are $(P, P_1, P_2, P_3, Q_1, \ldots, Q_n)$.
- $V(y) = y^n Q_n + \cdots + yQ_1 + P_3$ and $V_i = V(\mathsf{v}_i)$ as in the case of $\mathcal{H}_1$.

With the above two changes, the rest of key generation, encryption and decryption are as in $\mathcal{H}_1$.

More specifically, let us look at key generation. The private key d_{v} corresponding to $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ is defined to be

$$(xP_2 + r_1 V_1 + \ldots + r_j V_j, r_1 P, \ldots, r_j P) = (d_0, d_1, \ldots, d_j)$$

where $r_1, \ldots, r_j$ are random elements of $\mathbb{Z}_p$.

Since the maximum number of levels is not fixed in the set-up phase, the HIBE supports unbounded key delegation. This HIBE can be proved to be secure in model $\boldsymbol{\mathcal{M}}_1$. However, it is not secure in the sID-model, the reason being the following. Note that the first component d_0 of the secret key does not depend upon the ordering of the components of v. Hence, for any permutation of the components of v, the first component remains the same and thus, one can obtain a valid private key for any permutation of the components of v. In the sID model, the adversary can commit to an identity v^* and then ask the key extraction oracle for a private key of v' which is a permutation of v^*. Using the obtained private key of v', the adversary can easily obtain a private key for v^* and hence decrypt the challenge ciphertext. This shows insecurity in the sID model. Since, sID model is an accepted notion of security, insecurity in this model makes the unbounded depth HIBE less interesting and hence we will not consider this HIBE any further in this work.

7.4.2 HIBE $\mathcal{H}_2$

The description of $\mathcal{H}_2$ is similar to that of $\mathcal{H}_1$. The differences are in the specification of the public parameters and the definition of the V_i's.

1. Let $(n_1, \ldots, n_h)$ be a tuple of positive integers.
2. The new public parameters are $(P, P_1, P_2, \overrightarrow{P}_3, \overrightarrow{Q}_1, \ldots, \overrightarrow{Q}_h)$ where $\overrightarrow{P}_3 = (P_{3,1}, \ldots, P_{3,h})$ and $\overrightarrow{Q}_i = (Q_{i,1}, \ldots, Q_{i,n_i})$. The master secret is αP_2.
3. Define
$$V_i(y) = y^{n_i} Q_{i,n_i} + y^{n_i - 1} Q_{i,n_i - 1} + \ldots + y Q_{i,1} + P_{3,i}.$$
As before V_i is used to denote $V_i(\mathsf{v}_i)$.

With these differences, the rest of set-up, key generation, encryption and decryption algorithms remain the same.

Note: The HIBE $\mathcal{H}_1$ has the parameters h and n and we will write (h, n)-$\mathcal{H}_1$ to denote this explicit parametrization. The HIBE $\mathcal{H}_2$ is parametrized by the tuple $(n_1, \ldots, n_h)$ and we will write $(h, n_1, \ldots, n_h)$-$\mathcal{H}_2$ to denote this parametrization.

7.5 Security Reduction

In this section, we show security reductions for the HIBE protocols.

Recall that the security model $\boldsymbol{\mathcal{M}}_1$ is parametrized as (h, n)-$\boldsymbol{\mathcal{M}}_1$ while model $\boldsymbol{\mathcal{M}}_2$ is parametrized as $(h, n_1, \ldots, n_h)$-$\boldsymbol{\mathcal{M}}_2$. The advantage of an adversary in the security game is denoted by Adv. A subscript to this will denote the model and a superscript will denote the HIBE for which the result is being stated. For example, $\mathsf{Adv}^{(h,n)\text{-}\mathcal{H}_1}_{(h,n)\text{-}\boldsymbol{\mathcal{M}}_1}(t, q)$ denotes the maximum advantage of any adversary running in time t and making q queries to $\mathcal{O}_k$ in winning the security game defined by (h, n)-$\boldsymbol{\mathcal{M}}_1$ for the HIBE (h, n)-$\mathcal{H}_1$. We will assume that one scalar multiplication in G_1 can be done in time $O(\tau)$.

7.5.1 Security Reduction for $\mathcal{H}_1$

THEOREM 7.5.1. *Let h, n, q be positive integers and n' be another positive integer with $n' \leq n$. Then*
$$\mathsf{Adv}^{(h,n)\text{-}\mathcal{H}_1}_{(h,n')\text{-}\boldsymbol{\mathcal{M}}_1}(t, q) \leq \mathsf{Adv}^{\mathsf{DBDH}}(t + O(\tau n q)).$$

Proof : The security reduction is to show that if there is an adversary which can break $\mathcal{H}_1$ then one obtains an algorithm to solve DBDH. The heart of such an algorithm is a simulator which is constructed as follows. Given an instance of DBDH as input, the simulator

plays the security game (h, n')-$\boldsymbol{\mathcal{M}}_1$ with an adversary for (h, n)-$\mathcal{H}_1$. The adversary executes the commitment stage; then the simulator sets up the HIBE based on the adversary's commitment as well as the DBDH instance. The simulator gives the public parameters to the adversary and continues the game by answering all queries made by the adversary. In the process, it randomly chooses a bit γ and encrypts M_γ using the DBDH instance provided as input. Finally, the adversary outputs γ'. Based on the value of γ and γ', the simulator decides whether the instance it received is real or random. Intuitively, if the adversary has an advantage in breaking the HIBE protocol, the simulator also has an advantage in distinguishing between real and random instances. This leads to an upper bound on the advantage of the adversary in terms of the advantage of the simulator in solving DBDH.

We want to prove (h, n)-$\mathcal{H}_1$ secure in model (h, n')-$\boldsymbol{\mathcal{M}}_1$, where $1 \leq n' \leq n$. This means that the public parameters of the HIBE depend on n, while the adversary commits to a set $\mathcal{I}^*$ of size n' in the commit phase.

DBDH Instance: The simulator receives an instance $(P, P_1 = aP, P_2 = bP, Q = cP, Z)$ of DBDH.

The simulator now starts the security game for model $\boldsymbol{\mathcal{M}}_1$. This consists of several stages which we describe below. We will consider security against chosen plaintext attacks and hence the adversary will only have access to the key extraction oracle $\mathcal{O}_k$.

Adversary's Commitment: The adversary commits to a set $\mathcal{I}^*$ of size n'. The elements of $\mathcal{I}^*$ are from $\mathbb{Z}_p$. We write $\mathcal{I}^* = \{\mathsf{v}_1^*, \ldots, \mathsf{v}_{n'}^*\}$.

Set-Up: Define a polynomial $F(x)$ in $\mathbb{Z}_p[x]$ as follows.

$$\begin{aligned} F(x) &= (x - \mathsf{v}_1^*) \cdots (x - \mathsf{v}_{n'}^*) && (7.5.1) \\ &= x^{n'} + a_{n'-1}x^{n'-1} + \cdots + a_1 x + a_0 && (7.5.2) \end{aligned}$$

where the coefficients a_i's are in $\mathbb{Z}_p$ and are obtained from the values $\{\mathsf{v}_1^*, \ldots, \mathsf{v}_{n'}^*\}$. Since $F(x)$ is a polynomial of degree n' over $\mathbb{Z}_p$ and $\mathsf{v}_1^*, \ldots, \mathsf{v}_{n'}^*$ are its n distinct roots, we have $F(y) \neq 0$ for any $y \in \mathbb{Z}_p \setminus \{\mathsf{v}_1^*, \ldots, \mathsf{v}_{n'}^*\}$. The coefficients of $F(x)$ depend on the adversary's input and one cannot assume any distribution on these values. Define $a_{n'} = 1$ and $a_n = a_{n-1} = \cdots = a_{n'+1} = 0$.

For $1 \leq i \leq h$, define another set of polynomials $J_i(x)$ each of degree n in the following manner. Randomly choose $b_{0,1}, \ldots, b_{0,h}, b_1, \ldots, b_n$ from $\mathbb{Z}_p$. Define

$$J_i(x) = b_n x^n + b_{n-1}x^{n-1} + \cdots + b_1 x + b_{0,i} \quad (7.5.3)$$

The public parameters $P_{3,i}$s and Q_js are defined in the following manner.

- For $1 \leq i \leq h$, define $P_{3,i} = a_0 P_2 + b_{0,i} P$.
- For $1 \leq j \leq n$, define $Q_j = a_j P_2 + b_j P$.

Since $b_{0,1}, \ldots, b_{0,h}, b_1, \ldots, b_n$ are chosen randomly from $\mathbb{Z}_p$, the $P_{3,i}$s and the Q_js are random elements of G_1. The public parameters are given to the adversary. The master secret is aP_2, which is not known to the simulator.

Now comes the most crucial part of the proof. For $y \in \mathbb{Z}_p$,

$$\begin{aligned} V_i(y) &= P_{3,i} + yQ_1 + y^2Q_2 + \cdots + y^nQ_n \\ &= a_0P_2 + b_{0,i}P + y(a_1P_2 + b_1P) + y^2(a_2P_2 + b_2P) + \cdots + y^n(a_nP_2 + b_nP) \\ &= (a_0 + a_1y + a_2y^2 + \cdots + a_ny^n)P_2 + (b_{0,i} + b_1y + b_2y^2 + \cdots + b_ny^n)P \\ &= F(y)P_2 + J_i(y)P. \end{aligned}$$

This decomposes $V_i(y)$ into two parts – one depends on P_2 and the other depends on P. The part which depends on P_2 vanishes if and only if y is equal to some element of $\mathcal{I}^*$. The ability of the simulator to properly answer key extraction queries and generate a proper challenge ciphertext depends crucially on this fact.

Phase 1: In this stage, the adversary can make queries to $\mathcal{O}_k$, all of which have to be answered by the simulator. Suppose the adversary queries $\mathcal{O}_k$ on an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$, with $1 \leq j \leq h$. By the constraint of model $\boldsymbol{\mathcal{M}}_1$ all the v_i's cannot be in $\mathcal{I}^*$. Suppose $\imath$ is such that $\mathsf{v}_\imath$ is not in $\mathcal{I}^*$. Then $F(\mathsf{v}_\imath) \not\equiv 0 \bmod p$.

As in the protocol, define V_i to be $V_i(\mathsf{v}_i)$. Choose $r_1, \ldots, r_{\imath-1}, r'_\imath, r_{\imath+1}, \ldots, r_j$ randomly from $\mathbb{Z}_p$. Define $W = \sum_{i=1, i\neq\imath}^{j} r_iV_i$. The first component d_0 of the secret key for $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ is computed in the following manner.

$$d_0 = -\frac{J_\imath(v_\imath)}{F(v_\imath)}P_1 + r'_\imath(F(v_\imath)P_2 + J_\imath(v_\imath)P) + W.$$

The following computation shows that d_0 is properly formed.

$$\begin{aligned} d_0 &= \pm aP_2 - \frac{J_\imath(v_\imath)}{F(v_\imath)}P_1 + r'_\imath(F(v_\imath)P_2 + J_\imath(v_\imath)P) + W \\ &= aP_2 + \left(r'_\imath - \frac{a}{F(v_\imath)}\right)(F(v_\imath)P_2 + J_\imath(v_\imath)P) + W \\ &= aP_2 + \sum_{i=1}^{j} r_iV_i \end{aligned}$$

where $r_\imath = r'_\imath - a/F(v_\imath)$. Since $r'_\imath$ is random, so is $r_\imath$. The quantities $d_1, \ldots, d_j$ are computed in the following manner.

$$\begin{aligned} d_i &= r_iP && 1 \leq i \leq j,\ i \neq \imath; \\ &= r'_\imath P - \tfrac{1}{F(v_\imath)}P_1 = r_\imath P && \text{for } i = \imath. \end{aligned}$$

This technique is based on the algebraic techniques introduced by Boneh and Boyen [17] as discussed in Section 3.2.1. The generalization is in the definition of $F()$ and $J_i()$s. Here we take these to be polynomials, which allows us to tackle the case of adversary committing to more than one identity.

Challenge Generation: The adversary submits messages M_0, M_1 and an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ with $1 \leq j \leq h$. By the rules of model $\boldsymbol{\mathcal{M}}_1$, each $\mathsf{v}_i \in \mathcal{I}^*$ and so $F(\mathsf{v}_i) \equiv 0 \bmod p$ for $1 \leq i \leq j$. Consequently,

$$V_i = V_i(\mathsf{v}_i) = F(\mathsf{v}_i)P_2 + J_i(\mathsf{v}_i)P = J_i(\mathsf{v}_i)P$$

and hence

$$cV_i = cJ_i(\mathsf{v}_i)P = J_i(\mathsf{v}_i)(cP) = J_i(\mathsf{v}_i)Q$$

where $Q = cP$ was supplied as part of the DBDH instance. Note that it is possible to compute $W_i = cV_i$ even without knowing c. The simulator now randomly chooses a bit γ and returns

$$(M_\gamma \times Z, Q, W_1, \ldots, W_j)$$

to the adversary. If Z is real, then this is a proper encryption of M_γ under the identity v.

Phase 2: The key extraction queries in this stage are handled as in Phase 1.

Guess: The adversary outputs a guess γ'. The simulator outputs 1 if $\gamma = \gamma'$, else it outputs 0.

If $Z = e(P, P)^{abc}$, then the simulator provides a perfect simulation of the (h, n')-$\boldsymbol{\mathcal{M}}_1$ game. On the other hand, if Z is random, the adversary receives no information about the message M_γ from the challenge ciphertext.

The above shows that an adversary's ability to attack (h, n)-$\mathcal{H}_1$ HIBE in model (h, n')-$\boldsymbol{\mathcal{M}}_1$ can be converted into an algorithm for solving DBDH. The bound on the advantage follows from this fact. ■

Theorem 7.5.1. shows that an (h, n)-$\mathcal{H}_1$ HIBE is CPA-secure in model (h, n')-$\boldsymbol{\mathcal{M}}_1$ for $n' \leq n$. The next result shows that (h, n)-$\mathcal{H}_1$ is also secure in the h-sID model.

THEOREM 7.5.2. *Let h, n, q be positive integers. Then*

$$\mathsf{Adv}^{(h,n)\text{-}\mathcal{H}_1}_{h\text{-}sID}(t, q) \leq \frac{q}{p} + \mathsf{Adv}^{\mathsf{DBDH}}(t + O(\tau n q)).$$

Proof : The proof is similar to the proof of Theorem 7.5.1.. In the h-sID model, the adversary commits to an identity $(\mathsf{v}_1^*, \ldots, \mathsf{v}_j^*)$ where $1 \leq j \leq h$ and $\mathsf{v}_i \in \mathbb{Z}_p$. Randomly choose $\mathsf{v}_{j+1}^*, \ldots, \mathsf{v}_h^*$ from $\mathbb{Z}_p$. Randomly choose $b_1, \ldots, b_n, b_{0,1}, \ldots, b_{0,h}$ from $\mathbb{Z}_p$. For $1 \leq i \leq h$, define

$$\begin{aligned} F_i(x) &= x - \mathsf{v}_i^*; \\ J_i(x) &= b_n x^n + b_{n-1}x^{n-1} + \cdots + b_1 x + b_{0,i}. \end{aligned}$$

The protocol is set-up in the following manner. For $2 \leq j \leq n$, define $Q_j = b_j P$, $Q_1 = P_2 + b_1 P$ and for $1 \leq i \leq h$, define $P_{3,i} = -v_i^* P_2 + b_{0,i} P$. This defines all the public parameters.

For $1 \leq i \leq h$, we have

$$\begin{aligned} V_i(y) &= P_{3,i} + yQ_1 + y^2Q_2 + \cdots + y^nQ_n \\ &= (-v_i^*P_2 + b_{0,i}P) + y(P_2 + b_1P) + y^2b_2P + \cdots + y^nb_nP \\ &= (y - v_i^*)P_2 + (b_{0,i} + b_1y + \cdots + b_ny^n)P \\ &= F_i(y)P_2 + J_i(y)P \end{aligned}$$

The rest of the simulation is similar to the proof of Theorem 7.5.1. with one difference. If the adversary ever submits a key extraction query of the form $(\mathsf{v}_1, \ldots, \mathsf{v}_k)$, with $k > j$ and $\mathsf{v}_i = \mathsf{v}_i^*$ for $1 \leq i \leq j$, then the simulator aborts and outputs a random bit. Note that since the length of the identity is longer than the committed identity, the adversary is allowed to make such queries. The probability that $\mathsf{v}_i = \mathsf{v}_i^*$ for $j < i \leq k$ is $1/p^{k-j} \leq 1/p$. Since this can be repeated for each of the q key extraction queries, we have the additive degradation by the factor q/p. ■

7.5.2 Security Reduction for $\mathcal{H}_2$

THEOREM 7.5.3. *Let $h, n_1, \ldots, n_h, q$ be positive integers and $n'_1, \ldots, n'_h$ be another set of positive integers with $n'_i \leq n_i$ for $1 \leq i \leq h$. Then*

$$\mathsf{Adv}^{(h,n_1,\ldots,n_h)\text{-}\mathcal{H}_2}_{(h,n'_1,\ldots,n'_h)\text{-}\boldsymbol{\mathcal{M}}_2}(t,q) \leq \mathsf{Adv}^{\mathsf{DBDH}}(t + O(\tau nq))$$

where $n = \sum_{i=1}^{h} n_i$.

Proof : The proof is similar to the proof of Theorem 7.5.1.. The difference is in the definition of $F(x)$ and $J_i(x)$. In this case, for $1 \leq i \leq h$, we require $F_i(x)$. After the appropriate definition of $F_i(x)$ and $J_i(x)$, we show that $V_i(y)$ can be written as $V_i(y) = F_i(y)P_2 + J_i(y)P$. As mentioned in the proof of Theorem 7.5.1., the simulator's ability for answering key extraction queries and challenge generation depends upon this decomposition. The actual procedure for key extraction and challenge generation is very similar to that in the proof of Theorem 7.5.1. and these details are not provided. Thus, we provide the details of only two stages of the game – adversary's commitment and set-up.

The simulator is given an instance $(P, P_1 = aP, P_2 = bP, Q = cP, Z)$ of DBDH.

Adversary's Commitment: Following model $(h, n'_1, \ldots, n'_h)$-$\boldsymbol{\mathcal{M}}_2$, the adversary commits to sets $\mathcal{I}_1^*, \ldots, \mathcal{I}_u^*$, where $|\mathcal{I}_i^*| = n'_i$ and $1 \leq u \leq h$.

Set-Up: The simulator defines polynomials $F_1(x), \ldots, F_h(x)$, and $J_1(x), \ldots, J_h(x)$ in the following manner. For $1 \leq i \leq j$, define

$$\begin{aligned} F_i(x) &= \prod_{\mathsf{v} \in \mathcal{I}_i} (x - \mathsf{v}) \\ &= x^{n'_i} + a_{i,n'_i-1}x^{n'_i-1} + \cdots + a_{i,1}x + a_{i,0}; \end{aligned}$$

For $u+1 \leq i \leq h$ choose a random non-zero element $a_{i,0}$ from $\mathbb{Z}_p$ and define $F_i(x) = a_{i,0}$. Note that $F_i(x)$ is a non-zero constant polynomial. For $1 \leq i \leq u$, define $a_{i,n'_i} = 1$ and $a_{i,n_i} = \cdots = a_{i,n'_i+1} = 0$; for $u+1 \leq i \leq h$, set $n'_i = 0$ and $a_{i,1} = \cdots = a_{i,n_i} = 0$.

For $1 \leq i \leq h$ and $1 \leq j \leq n_i$ choose random elements $b_{i,j}$ from $\mathbb{Z}_p$. Define

$$J_i(x) \;=\; b_{i,n_i}x^{n_i} + b_{i,n_i-1}x^{n_i-1} + \cdots + b_{i,1}x + b_{i,0}.$$

Note that $F_i(x)$ is of degree n'_i while $J_i(x)$ is of degree n_i.

The public parameters are defined as follows.

- For $1 \leq i \leq h$, define $P_{3,i} = a_{i,0}P_2 + b_{i,0}P$.
- For $1 \leq i \leq h$ and $1 \leq j \leq n_i$ define $Q_{i,j} = a_{i,j}P_2 + b_{i,j}P$.

Since the $b_{i,j}$s are chosen randomly, the distribution of the public parameters is random. We now show the decomposition of $V_i(y)$.

$$\begin{aligned} V_i(y) &= P_{3,i} + yQ_{i,1} + y^2Q_{i,2} + \cdots + y^{n_i}Q_{i,n_i} \\ &= (a_{i,0}P_2 + b_{i,0}P) + y(a_{i,1}P_2 + b_{i,1}P) + y^2(a_{i,2}P_2 + b_{i,2}P) + \cdots + y^{n_i}(a_{i,n_i}P_2 + b_{i,n_i}P) \\ &= (a_{i,0} + a_{i,1}y + a_{i,2}y^2 + \cdots + a_{i,n_i}y^{n_i})P_2 + (b_{i,0} + b_{i,1}y + b_{i,2}y^2 + \cdots + b_{i,n_i}y^{n_i})P \\ &= F_i(y)P_2 + J_i(y)P. \end{aligned}$$

The rest of the simulation is very similar to that in the proof of Theorem 7.5.1.. Also, the bound on the advantage follows as in the above mentioned proof. ∎

The proof shows that $(h, n_1, \ldots, n_h)$-$\mathcal{H}_2$ is secure in model $(h, n'_1, \ldots, n'_h)$-$\boldsymbol{\mathcal{M}}_2$ with $n'_i \leq n_i$. Recall that $(h, 1, \ldots, 1)$-$\boldsymbol{\mathcal{M}}_2$ is same as the h-sID model and hence $(h, n_1, \ldots, n_h)$-$\mathcal{H}_2$ is secure in the h-sID model.

7.6 Multi-Receiver IBE

A multi-receiver IBE (MR-IBE) is an extension of the IBE, which allows a sender to encrypt a message in such a way that it can be decrypted by any one of a particular set of identities. In other words, there is one encryptor but more than one valid receivers. In IBE, the number of valid receivers is one. One trivial way to realize an MR-IBE from an IBE is to encrypt the same message several times. A non-trivial construction attempts to reduce the cost of encryption.

This notion was introduced in [4] and a non-trivial construction based on the Boneh-Franklin IBE (BF-IBE) was provided. The construction was proved to be secure in the *selective*-ID model using *random oracle* assumption. Note that the BF-IBE is secure in the full model using random oracle.

We show that $\mathcal{H}_1$ restricted to IBE can be modified to obtain an MR-IBE. The situation for $\mathcal{H}_2$ is almost identical. The required modifications to the protocol are as follows.

1. The encryption is converted into a hybrid scheme. Instead of multiplying the message with the "mask" $Z = e(P_1, P_2)^s$, the value Z is provided as input to a pseudorandom generator and the message (considered to be a bit string) is XORed with the resulting keystream.

2. The private key corresponding to an identity v is $d_{\mathsf{v}} = (xP_2 + rV_{\mathsf{v}}, rP)$, where $V_{\mathsf{v}} = P_{3,1} + V(\mathsf{v})$ as defined in Section 7.4.1.

3. Suppose the intended set of receivers is $\{\mathsf{v}_1, \ldots, \mathsf{v}_j\}$. Then the ciphertext consists of the encryption of the message plus a header of the form $(sP, sV_1, \ldots, sV_j)$, where V_i is as defined in the construction of $\mathcal{H}_1$ in Section 7.4.1 and s is a random element of $\mathbb{Z}_p$.

4. The receiver possessing the secret key d_{v_i} $(1 \leq i \leq j)$ can compute $e(P_1, P_2)^s$ in the standard manner and hence obtain the input to the pseudorandom generator. Thus it can decrypt the message.

The MR-IBE described above can be proved to be secure in the selective-ID model *without* random oracle. The security model for MR-IBE is the following [4]. In the commitment stage, the adversary commits to a set of identities; does not ask for the private key of these identities in the key extraction queries and finally asks for the encryption under this set of identities. Note that this is very similar to the model $\boldsymbol{\mathcal{M}}_1$ restricted to IBE. The only difference is that during the generation of the challenge ciphertext, in $\boldsymbol{\mathcal{M}}_1$, the adversary supplies only one identity out of the set of identities it had previously committed to, whereas in the model for MR-IBE, the adversary asks for the encryption under the whole set of these identities.

This difference is easily tackled in our proof in Section 7.5.1 which shows that $\mathcal{H}_1$ is secure in model $\boldsymbol{\mathcal{M}}_1$. Recall that the construction of the polynomial $F(x)$ is such that $F(\mathsf{v}) = 0$ for all $\mathsf{v} \in \mathcal{I}^*$, where $\mathcal{I}^*$ is the set of committed identities. In the challenge stage of the security proof for $\mathcal{H}_1$ as an IBE, we use this fact for only one identity (the identity given by the adversary). In the proof for MR-IBE, we will need to generate cV_i for all $\mathsf{v} \in \mathcal{I}^*$. Since $F(\mathsf{v}) = 0$ for any such v, this can be done in the standard fashion.

The above argument does not provide any security degradation. Hence, we obtain an MR-IBE which can be proved to be secure in the selective-ID model *without* random oracle.

7.7 Conclusion

In this chapter, we have generalized the notion of *selective*-ID secure HIBE. Two new security models $\boldsymbol{\mathcal{M}}_1$ and $\boldsymbol{\mathcal{M}}_2$ have been introduced. In the security game, both these models allow an adversary to commit to a set of identities (as opposed to a single identity in the sID model) before the set-up. During the challenge stage, the adversary can choose any one of the previously committed identities as a challenge identity. We provide two HIBE constructions $\mathcal{H}_1$ and $\mathcal{H}_2$ which are secure in the models $\boldsymbol{\mathcal{M}}_1$ and $\boldsymbol{\mathcal{M}}_2$ respectively. The public parameter size is smaller in case of $\mathcal{H}_1$. Further, we also show that $\mathcal{H}_1$ and $\mathcal{H}_2$ can be modified to

obtain an MR-IBE protocol which is secure in the sID model *without* random oracles. The only previous construction of MR-IBE is secure in the sID model using random oracle.

Chapter 8

Constant Size Ciphertext HIBE

8.1 Introduction

In this chapter, we present two variants of the constant size ciphertext HIBE of Boneh-Boyen-Goh discussed in Section 3.4.1. The aim of these variants is to be able to prove security in models which are stronger than the sID-model. The first variant (called ccHIBE) is secure in the generalized model $\boldsymbol{\mathcal{M}}_2$ introduced in the previous chapter. The second variant (called FullccHIBE) is secure in the full model. Security in the stronger models is attained at the cost of increasing the size of the public parameters and an increase in the computation time of the key generation and encryption. The ciphertext expansion as well as the decryption efficiency remains the same as that in the BBG-HIBE. Another point to note is that the reduction for the full model security is not tight. The security degradation is along the lines of HIBE of Chapter 6.

Constant size ciphertext HIBE is an important cryptographic primitive. To some extent, this justifies the variants that we present. On the other hand, from a technical point of view, our variants are not really straightforward extensions of the BBG-HIBE. There are certain technical subtleties which need to be taken care of for the proof to go through. Here we provide an overview of some of these aspects.

The BBG-HIBE has been proved to be secure in the sID-model. We augment this HIBE to attain security in model $\boldsymbol{\mathcal{M}}_2$. A similar construction has been done for the BB-HIBE in the previous chapter. The main technical novelty in the proof is the use of a polynomial which in [19] is of degree one. The other problem is that the security of the BBG-HIBE is based on the wDBDHI* problem. An instance of this problem consists of a tuple $(P, Q, aP, a^2P, \ldots, a^hP, Z)$, where P, Q are points elements of $G_1$and Z is an element of G_2. This instance is more complicated than an instance (P, aP, bP, cP, Z) of DBDH. Properly combining the polynomial-based security proof from the previous chapter with the wDBDHI* problem is the main technical difficulty in the proof of security in model $\boldsymbol{\mathcal{M}}_2$.

The public parameters in the BBG-HIBE consists of $(P, P_1, P_2, P_3, Q_1, \ldots, Q_h)$. In extending the BBG-HIBE to attain security in model $\boldsymbol{\mathcal{M}}_2$, we have to replace each Q_i by a

tuple $(Q_{i,1}, \ldots, Q_{i,n_i})$. The parameter P_3 does not change. The public parameters of the new HIBE are $(P, P_1, P_2, P_3, \overrightarrow{Q}_1, \ldots, \overrightarrow{Q}_h)$, where $\overrightarrow{Q}_i = (Q_{i,1}, \ldots, Q_{i,n_i})$. The Q-parameters capture the dependence on the level in both the BBG-HIBE and in its extension to $\boldsymbol{\mathcal{M}}_2$. The parameters P_1, P_2, P_3 do not depend on the number of levels of the HIBE.

On the other hand, when we modify the BBG-HIBE to attain security in the full model, we need to change the parameter P_3 to a tuple $\overrightarrow{P}_3 = (P_{3,1}, \ldots, P_{3,h})$. The new $\overrightarrow{P}_3$ depends on the number of levels of the HIBE. The public parameters in this case are of the form $(P, P_1, P_2, \overrightarrow{P}_3, \overrightarrow{Q}_1, \ldots, \overrightarrow{Q}_h)$, where $\overrightarrow{Q}_i = (Q_{i,1}, \ldots, Q_{i,l})$.

It has been mentioned in [19] that the BBG-HIBE protocol can be modified using Waters technique [89] to attain security in the full model. The change to P_3 forms a part of this modification, which was perhaps not anticipated in [19]. Adapting the techniques of Chapter 5 and Chapter 6 to the more complicated wDBDHI* assumption is the main technical difficulty in the proof of security in the full model.

8.2 ccHIBE

8.2.1 Construction

Setup: The maximum depth of the HIBE is a prespecified value denoted by h. Identities are of the form $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_u)$ with $1 \leq u \leq h$ and each $\mathsf{v}_i \in \mathbb{Z}_p^*$. Messages are elements of G_2.

Let $(n_1, \ldots, n_h)$ be a tuple of integers. Choose a random $\alpha \in \mathbb{Z}_p$ and set $P_1 = \alpha P$. Choose random points P_2, P_3 from G_1. The public parameters are $(P, P_1, P_2, P_3, \overrightarrow{Q}_1, \ldots, \overrightarrow{Q}_h)$ where $\overrightarrow{Q}_i = (Q_{i,1}, \ldots, Q_{i,n_i})$. Each $Q_{i,j}$ is chosen randomly from G_1. The master secret is αP_2.

Notation: For ease of description, we define a notation.

$$V_i(y) = y^{n_i} Q_{i,n_i} + \cdots + y Q_{i,1}.$$

Let $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ be an identity. By V_i we will denote $V_i(\mathsf{v}_i)$.

Key-Gen: Given an identity $(\mathsf{v}_1, \ldots, \mathsf{v}_u)$, pick a random $r \in \mathbb{Z}_p$ and output

$$d_{\mathsf{v}} = \left(\alpha P_2 + r \left(P_3 + \sum_{j=1}^{u} V_j \right),\ rP,\ r\overrightarrow{Q}_{u+1}, \ldots, r\overrightarrow{Q}_h \right)$$

where $r\overrightarrow{Q}_i = (rQ_{i,1}, \ldots, rQ_{i,n_i})$. A private key at level u consists of $(2 + \sum_{i=u+1}^{h} n_i)$ elements of G_1. Among these, only the first two are required in decryption which we denote as decryption sub-key. The rest are used to generate a private key for the next level as follows:

Let a private key for $(\mathsf{v}_1, \ldots, \mathsf{v}_{u-1})$ be $(A'_0, A'_1, \overrightarrow{B}'_u, \ldots, \overrightarrow{B}'_h)$, where

$$A'_0 = \alpha P_2 + r' \left(\sum_{j=1}^{u-1} V_j + P_3 \right),$$

$A'_1 = r'P$, and for $u \leq j \leq h$, $\overrightarrow{B}'_j = (r'Q_{j,1}, \ldots, r'Q_{j,n_j})$. Let $B'_{j,k} = r'Q_{j,k}$. Pick a random $r^* \in \mathbb{Z}_p$ and compute $d_{\mathsf{v}} = (A_0, A_1, \overrightarrow{B}_{u+1}, \ldots, \overrightarrow{B}_h)$ where

$$\begin{array}{lcl} A_0 & = & A'_0 + \sum_{i=1}^{n_u} \mathsf{v}_u^i B'_{u,i} + r^* \left(\sum_{j=1}^{u} V_j + P_3 \right), \\ A_1 & = & A'_1 + r^* P, \\ B_{u+1} & = & \overrightarrow{B}'_{u+1} + r^* \overrightarrow{Q}_{u+1}, \\ & & \ldots, \\ B_h & = & \overrightarrow{B}'_h + r^* \overrightarrow{Q}_h. \end{array}$$

If we put $r = r' + r^*$, then d_{v} is a proper private key for $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k)$.

Encrypt: To encrypt $M \in G_2$ under the identity $(\mathsf{v}_1, \ldots, \mathsf{v}_u) \in (\mathbb{Z}_p^*)^k$, pick a random $s \in \mathbb{Z}_p$ and output

$$\left(e(P_1, P_2)^s \times M, sP, s \left(P_3 + \sum_{j=1}^{u} V_j \right) \right).$$

Decrypt: To decrypt (A, B, C) using the private key $d_{\mathsf{v}} = (d_0, d_1, \ldots)$, compute

$$A \times \frac{e(d_1, C)}{e(B, d_0)} = e(P_1, P_2)^s \times M \times \frac{e\left(rP, s\left(P_3 + \sum_{j=1}^{u} V_j\right)\right)}{e\left(sP, \alpha P_2 + r\left(P_3 + \sum_{j=1}^{k} V_j\right)\right)} = M.$$

Note: ccHIBE is parametrized by $(n_1, \ldots, n_h)$ and we will write $(h, n_1, \ldots, n_h)$-ccHIBE to explicitly denote this parametrization.

8.2.2 Security Reduction

We wish to show that ccHIBE is secure in model $\boldsymbol{\mathcal{M}}_2$. Recall that Adv is used to denote the advantage of an adversary in attacking a HIBE. By the notation $\mathsf{Adv}^{(h,n_1,\ldots,n_h)\text{-ccHIBE}}_{(h,n'_1,\ldots,n'_h)\text{-}\boldsymbol{\mathcal{M}}_2}(t, q)$ we will denote the maximum advantage of an adversary which runs in time t and makes q key-extraction queries in attacking $(h, n_1, \ldots, n_h)$-ccHIBE in the model $(h, n'_1, \ldots, n'_h)$-$\boldsymbol{\mathcal{M}}_2$.

THEOREM 8.2.1. *Let $h, n_1, \ldots, n_h, q$ be positive integers and $n'_1, \ldots, n'_h$ be another set of positive integers with $n'_i \leq n_i$ for $1 \leq i \leq h$. Then*

$$\mathsf{Adv}^{(h,n_1,\ldots,n_h)\text{-ccHIBE}}_{(h,n'_1,\ldots,n'_h)\text{-}\boldsymbol{\mathcal{M}}_2}(t, q) \leq \mathsf{Adv}^{h\text{-wDBDHI}^*}(t + O(\tau n q))$$

where $n = \sum_{i=1}^{h} n_i$.

Proof : We show that an adversary playing the $(h, n'_1, \ldots, n'_h)$-$\boldsymbol{\mathcal{M}}_2$ game against a simulator for the HIBE $(h, n_1, \ldots, n_h)$-ccHIBE can be converted into an algorithm for the h-wDBDHI* problem.

An instance of the h-wDBDHI* problem is a tuple $\langle P, Q, Y_1, \ldots, Y_h, T\rangle$ where $Y_i = \alpha^i P$ for some random $\alpha \in \mathbb{Z}_p^*$ and T is either equal to $e(P,Q)^{\alpha^{h+1}}$ or a random element of G_2.

Adversary's commitment: The adversary outputs $\mathcal{I}_1^*, \ldots, \mathcal{I}_u^*$ where $1 \leq u \leq h$ and each set $\mathcal{I}_i^*$ is a set of cardinality n'_i.

Setup: Define polynomials $F_1(x), \ldots, F_h(x)$ as follows. For $1 \leq i \leq u$, define

$$\begin{aligned} F_i(x) &= \prod_{\mathsf{v} \in \mathcal{I}_i^*} (x - \mathsf{v}) \\ &= x^{n'_i} + a_{i,n'_i-1} x^{n'_i-1} + \ldots + a_{i,1} x + a_{i,0}. \end{aligned}$$

For $u+1 \leq i \leq h$, define $F_i(x) = x$ (and so $F_i(x) \not\equiv 0 \bmod p$ for any $x \in \mathbb{Z}_p^*$). For $1 \leq i \leq u$, define $a_{i,n'_i} = 1$ and $a_{i,n_i} = \cdots = a_{i,n'_i+1} = 0$; for $u+1 \leq i \leq h$, set $n'_i = 1$, $a_{i,0} = 0$, $a_{i,1} = 1$ and $a_{i,2} = \cdots = a_{i,n_i} = 0$.

For $1 \leq i \leq h$, define $J_1(x), \ldots, J_h(x)$ in the following manner.

$$J_i(x) = b_{i,n_i} x^{n_i} + b_{i,n_i-1} x^{n_i-1} + \ldots + b_{i,1} x + b_{i,0}$$

where $b_{i,j}$ are random elements of $\mathbb{Z}_p$. Note that $F_i(x)$ is of degree n'_i while $J_i(x)$ is of degree n_i.

The public parameters are defined as follows. Choose a random $\beta \in \mathbb{Z}_p$.

1. $P_1 = Y_1 = \alpha P$;
2. $P_2 = Y_h + \beta P = (\alpha^h + \beta) P$;
3. $P_3 = \sum_{i=1}^{h} (b_{i,0} P + a_{i,0} Y_{h-i+1})$; and
4. for $1 \leq i \leq h$, $1 \leq j \leq n_i$,
   $Q_{i,j} = b_{i,j} P + a_{i,j} Y_{h-i+1}$;

The public parameters are $(P, P_1, P_2, P_3, \overrightarrow{Q}_1, \ldots, \overrightarrow{Q}_h)$ where $\overrightarrow{Q}_i = (Q_{i,1}, \ldots, Q_{i,n_i})$. The distribution of the public parameters is as expected by the adversary in the original protocol. The corresponding master key $\alpha P_2 = Y_{h+1} + \beta Y_1$ is unknown to $\mathcal{B}$.

Phase 1: Suppose a key extraction query is made on $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_j)$ for $j \leq h$. (Note that j may be less than, equal to, or greater than u.)

If $j \leq u$, then there must be a $k \leq j$ such that $F_k(\mathsf{v}_k) \not\equiv 0 \bmod p$, as otherwise $\mathsf{v}_i \in \mathcal{I}_i^*$ for each $i \in \{1, \ldots, j\}$ – which is not allowed by the rules of $\boldsymbol{\mathcal{M}}_2$. In case $j > u$, it is possible that $F_1(\mathsf{v}_1) = \cdots = F_u(\mathsf{v}_u) = 0$. Then, since $\mathsf{v}_{u+1} \in \mathbb{Z}_p^*$ and $F_{u+1}(x) = x$, we have $F_{u+1}(\mathsf{v}_{u+1}) \not\equiv 0 \bmod p$.

Thus, in all cases, there is a k such that $F_k(\mathsf{v}_k) \not\equiv 0 \bmod p$. We choose k to be the first such value in the range $\{1, \ldots, j\}$ and so for $i < k$, we have $F_i(\mathsf{v}_i) \equiv 0 \bmod p$. We next show that it is possible to construct a valid private key for v from what is known to the adversary.

Recall that $Y_i = \alpha^i P$ and hence $Y_{i_1+i_2} = \alpha^{i_1} Y_{i_2}$. Choose a random r in $\mathbb{Z}_p$ and define

$$\begin{aligned}
A_1 &= \beta Y_1 - \frac{1}{F_k(\mathsf{v}_k)} \left(\sum_{i=1}^{j} J_i(\mathsf{v}_i) Y_k \right) + r \left(\sum_{i=1}^{j} (F_i(\mathsf{v}_i) Y_{h-i+1} + J_i(\mathsf{v}_i) P) \right); \\
A_2 &= -\frac{1}{F_k(\mathsf{v}_k)} \sum_{i \in \{1,\ldots,j\} \setminus \{k\}} F_i(\mathsf{v}_i) Y_{h+k-i+1}; \\
A_3 &= \sum_{i=j+1}^{h} \left(r(b_{i,0} P + a_{i,0} Y_{h-i+1}) - \frac{1}{F_k(\mathsf{v}_k)} (b_{i,0} Y_k + a_{i,0} Y_{h+k-i+1}) \right).
\end{aligned}$$

It is possible to compute A_1, A_2 and A_3 from what is known to the simulator. First note that $F_k(\mathsf{v}_k) \not\equiv 0 \bmod p$ and hence $1/F_k(\mathsf{v}_k)$ is well defined. The values $F_i(\mathsf{v}_i)$, $J_i(\mathsf{v}_i)$ and $P, Y_1, \ldots, Y_h$ are known to the simulator. Hence, A_1 and A_3 can be computed directly. In A_2, the values $Y_{h+2}, \ldots, Y_{h+k}$ are involved. However, the corresponding coefficients are $F_{k-1}(\mathsf{v}_{k-1}), \ldots, F_1(\mathsf{v}_1)$. By definition, k is the first integer in the set $\{1, \ldots, j\}$ such that $F_k(\mathsf{v}_k) \not\equiv 0 \bmod p$. Hence, $F_{k-1}(\mathsf{v}_{k-1}) \equiv \cdots \equiv F_1(\mathsf{v}_1) \equiv 0 \bmod p$ and consequently, the values $Y_{h+2}, \ldots, Y_{h+k}$ are not required by the simulator in computing A_2.

The first component d_0 of the private key d_{v} for v is obtained as $d_0 = A_1 + A_2 + A_3$. The following computation shows that this is proper.

$$\begin{aligned}
d_0 &= A_1 + A_2 + A_3 \\
&= \pm Y_{h+1} + A_1 + A_2 + A_3 \\
&= Y_{h+1} + \beta Y_1 - \alpha^k \frac{F_k(\mathsf{v}_k)}{F_k(\mathsf{v}_k)} Y_{h-k+1} + (A_1 - \beta Y_1) + A_2 + A_3 \\
&= \alpha P_2 + \left(r - \frac{\alpha^k}{F_k(\mathsf{v}_k)} \right) A
\end{aligned}$$

where

$$A = \sum_{i=1}^{j} (J_i(\mathsf{v}_i) P + F_i(\mathsf{v}_i) Y_{h-i+1}) + \sum_{i=j+1}^{h} (b_{i,0} P + a_{i,0} Y_{h-i+1}).$$

Now

$$J_i(\mathsf{v}_i) P + F_i(\mathsf{v}_i) Y_{h-i+1} = \sum_{l=1}^{n_i} b_{i,l} \mathsf{v}_i^l P + \sum_{l=1}^{n_i} a_{i,l} \mathsf{v}_i^l Y_{h-i+1} + b_{i,0} P + a_{i,0} Y_{h-i+1}$$

$$= \sum_{l=1}^{n_i} \mathsf{v}_i^l Q_{i,l} + b_{i,0}P + a_{i,0}Y_{h-i+1}$$
$$= V_i + b_{i,0}P + a_{i,0}Y_{h-i+1}.$$

Hence,

$$A = \sum_{i=1}^{j} V_i + \sum_{i=1}^{h}(b_{i,0}P + a_{i,0}Y_{h-i+1})$$
$$= P_3 + \sum_{i=1}^{j} V_i.$$

This shows

$$d_0 = \alpha P_2 + r'\left(P_3 + \sum_{i=1}^{j} V_i\right)$$

where $\tilde{r} = r - (\alpha^k / F_k(\mathsf{v}_k))$. Since r is random, so is $\tilde{r}$ and hence d_0 is properly formed. Also,

$$d_1 = -\frac{1}{F_k(\mathsf{v}_k)}Y_k + rP = -\frac{\alpha^k}{F_k(\mathsf{v}_k)}P + rP = \tilde{r}P$$

which is as required. To form a valid private key $\tilde{r}\overrightarrow{Q}_i$ has to be computed for $j < i \leq h$. This is done as follows.

$$\tilde{r}Q_{i,l} = \left(r - \frac{\alpha^k}{F_k(\mathsf{v}_k)}\right)(b_{i,l}P + a_{i,l}Y_{h-i+1})$$
$$= r(b_{i,l}P + a_{i,l}Y_{h-i+1}) - \frac{1}{F_k(\mathsf{v}_k)}\left(b_{i,l}Y_k + a_{i,l}Y_{h+k-i+1}\right).$$

Thus, we get

$$d_{\mathsf{v}} = \left(d_0, d_1, \tilde{r}\overrightarrow{Q}_{j+1}, \ldots, \tilde{r}\overrightarrow{Q}_h\right).$$

Challenge: After completion of Phase 1, the adversary outputs two messages $M_0, M_1 \in G_2$ together with a target identity $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_u^*)$ on which it wishes to be challenged. The constraint is each $\mathsf{v}_i^* \in \mathcal{I}_i^*$ and hence $F_i(\mathsf{v}_i^*) \equiv 0 \bmod p$ for $1 \leq i \leq u$. If $u \leq h$, then $a_{j,0} = 0$ for $u \leq j \leq h$. The simulator picks a random $b \in \{0,1\}$ and constructs the challenge ciphertext

$$\left(M_b \times T \times e(Y_1, \beta Q),\ Q,\ \left(\sum_{i=1}^{u} J_i(\mathsf{v}_i^*) + \sum_{i=u+1}^{h} b_{i,0}\right)Q\right).$$

Suppose, $Q = \gamma P$ for some unknown $\gamma \in \mathbb{Z}_p$. Using the fact $F_i(\mathsf{v}_i^*) \equiv 0 \bmod p$ for $1 \leq i \leq u$ and $a_{i,0} = 0$ for $u+1 \leq i \leq h$, we have

$$\left(\sum_{i=1}^{u} J_i(\mathsf{v}_i^*) + \sum_{i=u+1}^{h} b_{i,0}\right)Q = \gamma\left(\sum_{i=1}^{u}(J_i(\mathsf{v}_i^*)P + F_i(\mathsf{v}_i^*)Y_{h-i+1}) + \sum_{i=u+1}^{h}(a_{i,0}Y_{h-i+1} + b_{i,0}P)\right)$$

$$= \gamma \left(P_3 + \sum_{i=1}^{u} V_i \right).$$

If the input provided to the simulator is a true h-wDBDHI* tuple, i.e., $T = e(P,Q)^{(\alpha^{h+1})}$, then

$$\begin{aligned} T \times e(Y_1, \beta Q) &= e(P,Q)^{(\alpha^{h+1})} \times e(Y_1, \beta Q) \\ &= e(Y_h, Q)^{\alpha} \times e(\beta P, Q)^{\alpha} \\ &= e(Y_h + \beta P, Q)^{\alpha} \\ &= e(P_2, \gamma P)^{\alpha} \\ &= e(P_1, P_2)^{\gamma}. \end{aligned}$$

So, the challenge ciphertext

$$\left(M_b \times e(P_1, P_2)^{\gamma}, \gamma P, \gamma \left(\sum_{j=1}^{u} V_j + P_3 \right) \right)$$

is a valid encryption of M_b under $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_u^*)$. On the other hand, when T is random, the first component of the challenge ciphertext is a random element of G_2 and provides no information to the adversary.

Phase 2: This is similar to Phase 1.

Guess: Finally, the adversary outputs its guess $b' \in \{0,1\}$. The simulator outputs $1 \oplus b \oplus b'$.

This gives us the required bound on the advantage of the adversary in breaking the HIBE protocol. ∎

8.3 FullccHIBE

In this section, we consider the problem of constructing a constant size ciphertext HIBE which is secure in the full model. Our construction is based on the IBE scheme given by Waters [89] and its generalization given in Chapter 5.We note that the possibility of obtaining such a constant size ciphertext HIBE based on the work in [89] was mentioned as a passing remark in [19] though no details were provided.

Let the maximum height of the HIBE be h. Any identity v at height $k \leq h$ is represented by a k-tuple, $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k)$ where each v_i is an n-bit string. Each v_i is represented as $\mathsf{v}_i = (\mathsf{v}_{i,1}, \ldots, \mathsf{v}_{i,l})$, where $\mathsf{v}_{i,j}$ is a bit string of length n/l and $N = 2^n$. In other words, an n-bit identity at each level is represented as l blocks each of length n/l bits. This manner of representing identities is already used in Chapter 5 and Chapter 6.

In our construction, the public parameter size depends on both the size parameter l and the height h of the HIBE. If we decrease the value of l, the public parameter size also

decreases. However, the security of the HIBE degrades as l decreases. Hence, the decrease in the size of the public parameters comes at an increase in the security degradation. This trade-off can be converted into a trade-off between the memory required to store the public parameters and the time required for the different operations. This space/time trade-off has been studied in details in Chapter 5. A similar space/time trade-off also holds for the present case.

8.3.1 Construction

Setup: Choose a random secret $\alpha \in \mathbb{Z}_p$ and set $P_1 = \alpha P$. Randomly choose P_2; an h-length vector $\overrightarrow{P_3} = (P_{3,1}, \ldots, P_{3,h})$; and h many l-length vectors $\overrightarrow{U_1}, \ldots, \overrightarrow{U_h}$ from G_1, where each $\overrightarrow{U_j} = (U_{j,1}, \ldots, U_{j,l})$. The public parameters consist of the elements

$$\langle P, P_1, P_2, \overrightarrow{P_3}, \overrightarrow{U_1}, \ldots, \overrightarrow{U_h} \rangle$$

while the master secret is αP_2. Note that, for each level i of the HIBE we have $l+1$ elements i.e., $P_{3,i}$ and $\overrightarrow{U_i}$.

Notation: Let v_j be an n-bit string written as $\mathsf{v}_j = (\mathsf{v}_{j,1}, \ldots, \mathsf{v}_{j,l})$, where each $\mathsf{v}_{j,i}$ is an (n/l)-bit string. Define

$$V_j = P_{3,j} + \sum_{i=1}^{l} \mathsf{v}_{j,i} U_{j,i}.$$

The modularity introduced by this notation is useful in describing the protocol.

Key-Gen: Given an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k)$ for $k \leq h$, this algorithm generates the private key d_{v} of v as follows. Choose a random element $r \in \mathbb{Z}_p$ and output

$$d_{\mathsf{v}} = \left(xP_2 + r\left(\sum_{j=1}^{k} V_j\right),\ rP,\ rP_{3,k+1}, \ldots, rP_{3,h},\ r\overrightarrow{U}_{k+1}, \ldots, r\overrightarrow{U}_h \right)$$

where $r\overrightarrow{U_j} = (rU_{j,1}, \ldots, rU_{j,l})$.

Note that, the private key for v can also be generated from the private key of $(\mathsf{v}_1, \ldots, \mathsf{v}_{k-1})$ as is the general requirement for a HIBE scheme. Suppose the private key for $(\mathsf{v}_1, \ldots, \mathsf{v}_{k-1})$ is $(d'_0, d'_1, a'_k, \ldots, a'_h, \overrightarrow{b'_k}, \ldots, \overrightarrow{b'_h})$, where $\overrightarrow{b'_i} = \langle b'_{i,1}, \ldots, b'_{i,\ell} \rangle$. Pick a random $r' \in \mathbb{Z}_p$ and then $d_{\mathsf{v}} = (d_0, d_1, a_{k+1}, \ldots, a_h, \overrightarrow{b}_{k+1}, \ldots, \overrightarrow{b}_h)$, where

$$\begin{aligned} d_0 &= d'_0 + a'_k + \textstyle\sum_{i=1}^{l} \mathsf{v}_{k,i} b'_{k,i} + r' \sum_{j=1}^{k} V_j; \\ d_1 &= d'_1 + r'P; \end{aligned}$$

and for $k+1 \leq j \leq h$.

$$\begin{aligned} a_j &= a'_j + r' P_{3,j}; \\ \overrightarrow{b_j} &= \overrightarrow{b'_j} + r' \overrightarrow{U_j}, \end{aligned}$$

Encrypt: To encrypt a message $M \in G_2$ under the public key $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k)$ choose a random $s \in \mathbb{Z}_p$ and then the cipher text is

$$C = \left(e(P_1, P_2)^s \times M, sP, s\sum_{j=1}^{k} V_j \right)$$

where V_j is as defined in Key Generation part.

Decrypt: Let (A, B, C) be a ciphertext and $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k)$ be the corresponding identity. Then we decrypt using $d_{\mathsf{v}} = (d_0, d_1, \ldots)$ as

$$A \times \frac{e(d_1, C)}{e(B, d_0)} = M.$$

Note that, only the first two components of the private key are required for the decryption.

8.3.2 Security Reduction

Security of the FullccHIBE scheme described above can be reduced from the hardness of the h-wDBDHI* problem. The reduction combines ideas from the proof in Section 8.2.2 with ideas from the proofs in Chapter 5. In particular, the general idea of tackling adaptive adversaries including an "artificial abort" stage is from Waters [89], the modification for the case of $1 < l \leq n$ is from Chapter 5 whereas the idea of the simulation of the key-extraction queries is from the proof in Section 8.2.2 and is based on algebraic techniques originally used by Boneh and Boyen [17]. To explain this idea further, the simulator in the proof will abort on certain queries made by the adversary and also on certain challenge identities. The idea of controlling this abort strategy is based on the technique from [89]. On the other hand, if on a certain query, the simulator does not abort, then the technique for the actual simulation of the key-extraction oracle is very similar to the technique in Section 8.2.2.

The challenge generation is a bit different due to the fact that in FullccHIBE level j of the HIBE has a parameter $P_{3,j}$, whereas in ccHIBE, there is one parameter P_3 for all levels of the HIBE. In case of BBG-HIBE or its augmented version ccHIBE, the height of the target identity is fixed in the commitment stage itself. Based on this information the simulator sets up the HIBE and the effect of the committed identity tuple for BBG-HIBE or the sets of committed identities in ccHIBE is assimilated in P_3. In case of FullccHIBE there is no prior commitment stage in the reduction and the number of levels in the target identity may vary between 1 and h. This is the intuitive reason of why we need different $P_{3,i}$ for each level of the HIBE.

THEOREM 8.3.1. *The* FullccHIBE *protocol is* (ϵ, t, q)-IND-ID-CPA *secure assuming that the* (t', ϵ', h)-*wDBDHI** *assumption holds, where*

$\epsilon \leq 2\epsilon'/\lambda$;
$t' = t + O(uq) + O(\epsilon^{-2}\ln(\epsilon^{-1})\lambda^{-1}\ln(\lambda^{-1}))$; *and*
$\lambda = 1/(2(4lq2^{n/l})^h)$.

We assume $2q > 2^{n/l}$.

Proof : Suppose $\mathcal{A}$ is a (t, q)-CPA adversary for the h-HIBE, then we construct an algorithm $\mathcal{B}$ that solves the h-wDBDHI* problem. $\mathcal{B}$ takes as input a tuple $\langle P, Q, Y_1, \ldots, Y_h, T\rangle$ where $Y_i = \alpha^i P$ for some random $\alpha \in \mathbb{Z}_p^*$ and T is either equal to $e(P,Q)^{\alpha^{h+1}}$ or is a random element of G_2. We define the following game between $\mathcal{B}$ and $\mathcal{A}$.

Setup: $\mathcal{B}$ chooses random $u_1, \ldots, u_h \in \mathbb{Z}_m$ and l-length vectors $\overrightarrow{x_1}, \ldots, \overrightarrow{x_h}$ with entries from $\mathbb{Z}_m$. Here $m = 2\max(2q, 2^{n/l}) = 4q$. Similarly, it chooses random $v_1, \ldots, v_h \in \mathbb{Z}_p$ and l-length vectors $\overrightarrow{y_1}, \ldots, \overrightarrow{y_h}$ from $\mathbb{Z}_p$. It further chooses k_j for $1 \leq j \leq h$ randomly from $\{0, \ldots, \mu_l\}$, where $\mu_l = l(N^{1/l} - 1)\}$. Let, $\mathsf{v}_j = (\mathsf{v}_{j,1}, \ldots, \mathsf{v}_{j,l})$. For $1 \leq j \leq h$, it then defines the functions:

$$\begin{aligned}
F_j(\mathsf{v}_j) &= p + mk_j - u_j - \sum_{i=1}^{l} x_{j,i}\mathsf{v}_{j,i} \\
J_j(\mathsf{v}_j) &= v_j + \sum_{i=1}^{l} y_{j,i}\mathsf{v}_{j,i} \\
K_j(\mathsf{v}_j) &= \begin{cases} 0 & \text{if } u_j + \sum_{i=1}^{l} x_{j,i}\mathsf{v}_{j,i} \equiv 0 \bmod m \\ 1 & \text{otherwise} \end{cases}
\end{aligned}$$

These functions are used to control the abort strategy by the simulator.

Next, $\mathcal{B}$ assigns $P_1 = Y_1,\ P_2 = Y_h + yP$, $P_{3,j} = (p + mk_j - u_j)Y_{h-j+1} + v_j P$ for $1 \leq j \leq h$ and $U_{j,i} = -x_{j,i}Y_{h-j+1} + y_{j,i}P$ for $1 \leq j \leq h$ and $1 \leq i \leq l$. It provides $\mathcal{A}$ the public parameters $\langle P, P_1, P_2, \overrightarrow{P}_3, \overrightarrow{U}_1, \ldots, \overrightarrow{U}_h\rangle$. Everything else is internal to $\mathcal{B}$. Note that from $\mathcal{A}$'s point of view the distribution of the public parameters is identical to the distribution of the public parameters in an actual setup. The master secret αP_2 is unknown to $\mathcal{B}$.

Using the definition of the public parameters it is possible to show that

$$V_j = P_{3,j} + \sum_{i=1}^{l} \mathsf{v}_{j,i}U_{j,i} = F_j(\mathsf{v}_j)Y_{h-j+1} + J_j(\mathsf{v}_j)P.$$

As in the proof of Theorem 8.2.1., this fact is crucial to the answering key-extraction queries and challenge generation.

Phase 1: Suppose $\mathcal{A}$ asks for the private key corresponding to an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_u)$, for $u \leq h$. $\mathcal{B}$ first checks whether there exists a $j \in \{1, \ldots, u\}$ such that $K(\mathsf{v}_j) \neq 0$. It aborts

and outputs a random bit if there is no such j. Otherwise, it answers the query in a manner similar to that in the proof of Theorem 8.2.1..

$\mathcal{B}$ chooses r randomly from $\mathbb{Z}_p$ and computes

$$\begin{aligned} d_{0|j} &= -\frac{J(\mathsf{v}_j)}{F_j(\mathsf{v}_j)}Y_j + yY_1 + r(F_j(\mathsf{v}_j)Y_{h-j+1} + J(\mathsf{v}_j)P); \\ d_1 &= \frac{-1}{F_j(\mathsf{v}_j)}Y_j + rP. \end{aligned}$$

It is standard to show that $d_{0|j} = \alpha P_2 + \tilde{r}V_j$ and $d_1 = \tilde{r}P$, where $\tilde{r} = r - \frac{\alpha^j}{F_j(I_j)}$. As in the proof of Theorem 8.2.1., it is possible to show that $\mathcal{B}$ can compute $\tilde{r}V_i$ for any $i \in \{1, \ldots, u\} \setminus \{j\}$; and $\tilde{r}P_{3,k}$, $\tilde{r}\overrightarrow{U_k}$ for $u < k \leq h$. The simulator computes $d_0 = d_{0|j} + \sum_{i \in \{1,\ldots,u\}\setminus\{j\}} \tilde{r}V_i$. $\mathcal{A}$ is provided the private key corresponding to v as $d_{\mathsf{v}} = \left(d_0, d_1, \tilde{r}P_{3,u+1}, \ldots, \tilde{r}P_{3,h}, \tilde{r}\overrightarrow{U}_{u+1}, \ldots, \tilde{r}\overrightarrow{U}_h\right)$. Note that d_{v} is a valid private key for v following the proper distribution. $\mathcal{B}$ will be able to generate this d_{v} as long as there is a $j \in \{1, \ldots, u\}$ such that $F(\mathsf{v}_j, k_j) \not\equiv 0$ for which it suffices to have $K(\mathsf{v}_j) \neq 0$.

Challenge: $\mathcal{A}$ submits two messages $M_0, M_1 \in G_2$ and an identity $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_{h'}^*)$, $h' \leq h$ on which it wants to be challenged. $\mathcal{B}$ aborts and outputs a random bit, if $F_j(\mathsf{v}_j^*) \not\equiv 0$ for any $j \in \{1, \ldots, h'\}$. Otherwise, $\mathcal{B}$ chooses a random bit $\gamma \in \{0, 1\}$ and gives $\mathcal{A}$ the tuple $\mathsf{CT} = (T \times e(Y_1, yQ) \times M_\gamma, Q, \sum_{j=1}^{h'} J(\mathsf{v}_j^*)Q)$.

If $\langle P, Q, Y_1, \ldots, Y_h, T\rangle$ given to $\mathcal{B}$ is a valid h-wDBDHI* tuple, i.e., $T = e(P, Q)^{\alpha^{h+1}}$ then CT is a valid encryption for M_γ. Suppose $Q = cP$ for some unknown $c \in \mathbb{Z}_p$. Then the first component of CT can be seen to be $e(P_1, P_2)^c$. Further, using $F_j(\mathsf{v}_j^*) \equiv 0 \bmod p$ it can be shown that $J(\mathsf{v}_j^*)Q = cV_j$. The correctness of the third component of CT follows from this fact. If T is a random element of G_2, CT gives no information about $\mathcal{B}$'s choice of γ.

Phase 2: Similar to Phase 1, with the restriction that $\mathcal{A}$ cannot ask for the private key of ID^* or any of its ancestors.

Guess: $\mathcal{A}$ outputs a guess γ' of γ.

A lower bound λ on the probability of aborting up to this stage is the following.

$$\lambda = \frac{1}{2(4lq2^{n/l})^h}.$$

Waters [89] obtains a similar bound (for the case $l = n$) in the context of an IBE secure in the full model under the DBDH assumption. In the same paper, Waters had suggested a construction for HIBE where new public parameters are generated for each level of the HIBE. Generating new public parameters for each level of the HIBE simplifies the probability analysis for the lower bound on the probability of abort.

The security of FullccHIBE is based on the h-wDBDHI* problem which is obtained by modifying the BBG-HIBE. For each level of the HIBE we have separate public parameters $P_{3,i}$ and $(U_{i,1}, \ldots, U_{i,l})$. This makes it possible to easily apply the reasoning of Waters leading to the above mentioned lower bound.

At this point, we have to also use the technique of "artificial abort" employed by Waters [89]. The idea is that the probability of aborting up to this is not independent of the adversarial queries. The idea of the artificial abort technique is to allow the simulator to sample the transcript of queries it obtained from the adversary and on certain conditions abort and output a random bit. This increases the total probability of abort and makes it almost equal for all adversarial inputs. This helps in the probability analysis. The effect of sampling the transcript is to increase the runtime of the simulator. See [89] for the details. Incorporating the probability analysis of Waters into the present situation in a straightforward manner we obtain the required result. ■

8.4 Conclusion

In this chapter, we have presented two variants of the constant size ciphertext HIBE proposed by Boneh, Boyen and Goh [19]. Both the constructions have constant size ciphertext expansion. The first variant is proved to be secure in the generalized selective-ID model $\boldsymbol{\mathcal{M}}_2$ introduced in Chapter 7 while the second variant is secure in the full model. We combine techniques from several works along with the BBG-HIBE to obtain the new constructions and their proofs.

Chapter 9

Augmented *Selective*-ID Model: Case of Constant Size Ciphertext HIBE

9.1 Introduction

In this chapter, we augment the *selective*-ID security model by giving more power to the adversary. Recall that, for a HIBE of maximum height h, in the sID model the adversary commits to an identity tuple $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_j^*)$, $j \leq h$ and in the challenge phase obtains an encryption under v^*. In the augmented version, which we call *selective*$^+$-ID model, the adversary is allowed in the challenge phase to ask for an encryption under $\mathsf{v}^+ = (\mathsf{v}_1^*, \ldots, \mathsf{v}_{j'}^*)$, where $1 \leq j' \leq j$. This way we provide additional flexibility to the adversary in choosing the target identity.

In the sID model, the adversary is restricted from making private key query for *any* prefix of v^*. Consequently, a "natural" intuition is that the adversary be allowed to choose any prefix of v^* as a challenge identity. Unfortunately, the *selective*-ID model does not allow this flexibility to the adversary. In the s$^+$ID model, this flexibility is introduced and the challenge identity is allowed to be any prefix of v^*. Clearly, any protocol secure in the s$^+$ID model is also secure in the sID model, though the converse is not necessarily true.

We show that the security reduction for BB-HIBE [17] satisfies this notion of s$^+$ID security. On the other hand, the security proof of the BBG-HIBE does not go through in the s$^+$ID model. A simple modification of this proof gives a proof of security for the BBG-HIBE in the s$^+$-ID model. But this proof yields a multiplicative security degradation by a factor of h, where h is the maximum number of levels in the HIBE. Admittedly, a security degradation by a factor of h is not much. However, the sID and the s$^+$ID models are really restrictive models and one would like to obtain a protocol without any security degradation. We modify the BBG-HIBE to obtain a new constant size ciphertext HIBE, $\mathcal{G}_1$ which is secure in the s$^+$ID model without any security degradation.

Identities in the BBG-HIBE are tuples whose components are elements of $\mathbb{Z}_p^*$, where p is

a suitable large prime. It is an easy observation that if these components are allowed to be elements of $\mathbb{Z}_p$, then the BBG-HIBE is not secure (we provide the details in Section 9.2.2). In our construction, $\mathcal{G}_1$, we allow the components of the identity tuples to be elements of $\mathbb{Z}_p$.

We next modify this construction to obtain a constant size ciphertext HIBE, $\mathcal{G}_2$ which is proved to be secure in model $\boldsymbol{\mathcal{M}}_2$ augmented in the line of s$^+$ID model.

Our third construction is a product construction, in the sense that the constructed HIBE can be seen to be a "product" of two individual HIBEs. We have mentioned in Section 3.4 that a product construction combining the BB-HIBE and the BBG-HIBE has been presented in [19]. We consider the product of $\mathcal{H}_1$ of Chapter 7 with $\mathcal{G}_2$ to obtain a new HIBE $\mathcal{G}_3$. This HIBE is secure in model $\boldsymbol{\mathcal{M}}_1$ and reduces the size of the ciphertext in $\mathcal{H}_1$ by a factor of h, where h is the number of levels in $\mathcal{G}_2$.

The decryption subkey (i.e., the part of the private key required for decryption) for both $\mathcal{G}_1$ and $\mathcal{G}_2$ are equal to that of BBG-HIBE. While in $\mathcal{G}_3$ the size of the decryption subkey is reduced by a factor of h over the size of the decryption subkeys in $\mathcal{H}_1$.

9.2 From Selective-ID to Selective$^+$-ID Model

We modify the challenge phase of the selective-ID model to give more power to the adversary.

Challenge: $\mathcal{A}$ outputs two equal length messages M_0, M_1 and an identity v^+ where v^+ is either v^* or any of its prefixes. In response it receives an encryption of M_γ under v^+, where γ is chosen uniformly at random from $\{0, 1\}$.

We refer to this new model as selective$^+$-ID model (s$^+$ID model in short). This model is stronger than the original sID model because now the adversary is allowed to ask for a challenge ciphertext not only on v^* but also on any of its prefixes. In case of IBE both the models are same as we have only one level. For HIBE, a protocol secure in the selective$^+$-ID model is obviously secure in the selective-ID model, however the converse may not hold. We now analyse two well known HIBEs in the light of this new model.

9.2.1 Case of Boneh-Boyen HIBE

We show that the Boneh-Boyen HIBE discussed in Section 3.2.1 is secure in the s$^+$-ID model.

The original reduction as discussed in Section 3.2.1 goes through without almost any modification for the s$^+$-ID model. The only change is in challenge generation.

Initialization: $\mathcal{A}$ commits to a target identity $\mathsf{v}^* = \mathsf{v}_1^*, \ldots, \mathsf{v}_k^*$ of height $k \leq h$. If $k < h$, $\mathcal{B}$ adds extra random elements from $\mathbb{Z}_p$ to make v^* an identity of height h.

Setup: $\mathcal{B}$ picks random $\alpha_1, \ldots, \alpha_h \in \mathbb{Z}_p$ and defines $Q_j = \alpha_j P - \mathsf{v}_j^* P_1$ for $1 \leq j \leq h$. It gives $\mathcal{A}$ the public parameters $\mathsf{PP} = \langle P, P_1, P_2, Q_1, \ldots, Q_h \rangle$. Here the $\mathsf{msk} = aP_2 = abP$ is unknown to $\mathcal{B}$. Define the function $F_j(x) = xP_1 + Q_j = (x - \mathsf{v}_j^*)P_1 + \alpha_j P$ for $1 \leq j \leq h$.

Phase 1 and Phase 2: As discussed in Section 3.4.1.

Challenge: After completion of Phase 1, $\mathcal{A}$ outputs two messages $M_0, M_1 \in G_2$ and an identity tuple $\mathsf{v}^+ = \mathsf{v}_1^*, \ldots, \mathsf{v}_u^*$, $u \leq k$. $\mathcal{B}$ chooses a random bit γ and forms the ciphertext $C = \langle M_\gamma \cdot Z, cP, \alpha_1 cP, \ldots, \alpha_u cP \rangle$. Note that, $F_i(\mathsf{v}_i^*) = \alpha_i P$, so

$$C = \langle M_\gamma \cdot Z, cP, cF_1(\mathsf{v}_1^*), \ldots, cF_u(\mathsf{v}_u^*) \rangle.$$

If $Z = e(P,P)^{abc} = e(P_1, P_2)^c$ then C is a valid encryption of M_γ.

9.2.2 Case of Boneh-Boyen-Goh HIBE

The original BBG-HIBE protocol has been described in Section 3.4.1.

The BBG-HIBE is proved to be secure in the selective-ID model (Theorem 3.1 of [19]). We now show that the proof is not sufficient for the augmented s$^+$ID model and how can it be modified to achieve security in the s$^+$ID model.

In the original sID model, an adversary declares an identity v^* that it intends to attack before the system is set up. Suppose $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_m^*)$ where $m \leq h$. In the reduction given in [19], the following is done. If $m < h$ then the simulator appends $(h-m)$ zeros to v^* so that v^* is a vector of length h. Note that, in the protocol individual components of an identity are elements of $\mathbb{Z}_p^*$ so the adversary is restricted from making a query where one or more components of the identity is 0. (BB-HIBE does not have this restriction). The reduction in [19] crucially depends on this step.

In the protocol, a single element of G_1 (i.e. Q_i) is associated with the ith level of the HIBE and we have another element, namely P_3 which is required for the security reduction.

The simulator $\mathcal{B}$ is given as input a random tuple $(P, Q, Y_1, \ldots, Y_h, T)$ where $Y_i = \alpha^i P$s for $1 \leq i \leq h$ for some unknown α. The task of $\mathcal{B}$ is to decide whether $T = e(P,Q)^{\alpha^{h+1}}$ or T is a random element of G_2.

We now reproduce the relevant steps of the reduction in Theorem 3.1 in [19].

Setup: $\mathcal{B}$ picks a random $\gamma \in \mathbb{Z}_p$ and sets $P_1 = Y_1 = \alpha P$ and $P_2 = Y_h + \gamma P$. Next, $\mathcal{B}$ picks random $\gamma_1, \ldots, \gamma_h \in \mathbb{Z}_p$ and sets $Q_j = \gamma_j P - Y_{h-j+1}$ for $j = 1, \ldots, h$. $\mathcal{B}$ also picks a random $\delta \in \mathbb{Z}_p$ and sets $P_3 = \delta P + \sum_{j=1}^{h} \mathsf{v}_j^* Y_{h-j+1}$. $\mathcal{B}$ gives $\mathcal{A}$ the public parameters $\langle P, P_1, P_2, P_3, Q_1, \ldots, Q_h \rangle$.

Note that, the effect of $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_m^*)$ is assimilated in P_3. In case, m (the depth of the challenge identity tuple v^*) is less than h, we have $\mathsf{v}_{m+1}^* = \cdots = \mathsf{v}_h^* = 0$, so $\mathsf{v}_j^* Y_{h-j+1}$ for $m < j \leq h$ has no effect on P_3. The Q_js in the public parameter are independent of the target identity and depends only on the Y_{h-j+1}s after suitable randomization. In contrast, in case of the BB-HIBE, each Q_j depends on v_j^* i.e., the component corresponding to level j in target identity v^* and we have no term like P_3.

Given this setup, Boneh, Boyen and Goh show that all the private key queries of $\mathcal{A}$ can be answered (see Phase 1 in the proof of Theorem 3.1 in [19] for details).

Now, suppose in the challenge phase $\mathcal{A}$ asks the encryption under v^+ which is a prefix of v^*, i.e., $\mathsf{v}^+ = \mathsf{v}_1^*, \ldots, \mathsf{v}_\mu^*$, $\mu \leq m$. If $\mu = m$, then the original reduction goes through and we get a proper encryption of M_b provided the input instance is a true h-wDBDHI* instance. However, if $\mu < h$, then the original reduction in [19] *does not* give a proper encryption of M_b even if the input is a true h-wDBDHI* instance as we show below.

Let $Q = cP$ for some unknown $c \in \mathbb{Z}_p$, then third component of the challenge ciphertext is

$$\begin{aligned} C &= \left(\delta + \sum_{j=1}^{h} \mathsf{v}_j^* \gamma_j\right) Q \\ &= c\left(\sum_{j=1}^{h} \mathsf{v}_j^*(\gamma_j P - Y_{h-j+1}) + \delta P + \sum_{j=1}^{h} \mathsf{v}_j^* Y_{h-j+1}\right) \\ &= c(\mathsf{v}_1^* Q_1 + \ldots, \mathsf{v}_m^* Q_m + P_3) \;\text{ since }\; \mathsf{v}_{m+1}^* = \cdots = \mathsf{v}_h^* = 0 \end{aligned}$$

However, this corresponds to an encryption under v^* not v^+. To get a valid encryption under $\mathsf{v}^+ = \mathsf{v}_1^*, \ldots, \mathsf{v}_\mu^*$, the third component of the ciphertext should be of the form

$$\begin{aligned} C' &= c(\mathsf{v}_1^* Q_1 + \cdots + \mathsf{v}_\mu^* Q_\mu + P_3) \\ &= c\left(\sum_{j=1}^{\mu} \mathsf{v}_j^*(\gamma_j P - Y_{h-j+1}) + \delta P + \sum_{j=1}^{h} \mathsf{v}_j^* Y_{h-j+1}\right) \\ &= c\left(\sum_{j=1}^{\mu} \mathsf{v}_j^* \gamma_j P + \delta P + \sum_{j=\mu+1}^{m} \mathsf{v}_j^* Y_{h-j+1}\right) \\ &= \left(\delta + \sum_{j=1}^{\mu} \mathsf{v}_j^* \gamma_j\right) Q + c \sum_{j=\mu+1}^{m} \mathsf{v}_j^* Y_{h-j+1} \end{aligned}$$

This C' cannot be computed by $\mathcal{B}$ without the knowledge of c.

So we see that the BB-HIBE satisfies the security requirement of s$^+$-ID model but the BBG-HIBE does not. This difference is due to the fact that the simulator in the BBG-HIBE assimilates the effect of the challenge identity tuple in a single element in the public parameter (i.e., P_3) where as in case of BB-HIBE the public parameter corresponding to each level depends on the corresponding component in the target. So the reduction in [19] provides a proof of security against an adversary in the sID model. However, in the modified s$^+$-ID model the adversary is free to choose the length of the target identity to be any value between 1 and m. Because of the construction of P_3 in the reduction of [19], $\mathcal{B}$ cannot form a proper challenge unless $\mu = m$.

Another difference in the BB-HIBE and the BBG-HIBE is that in the former, components of identities are elements of $\mathbb{Z}_p$, whereas in the later the identity components are elements of $\mathbb{Z}_p^*$. It is an easy observation that if zero is allowed to be an identity component, then the BBG-HIBE is not secure. A sketch of the argument is as follows. In the sID game,

an adversary has to commit to an identity before the HIBE is set-up. Let adversary $\mathcal{A}$ commit to an identity $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_k^*)$ for some k with $1 \leq k < h$. In the query phase, $\mathcal{A}$ issues a private key query for the identity $\mathsf{v} = (\mathsf{v}_1^*, \ldots, \mathsf{v}_k^*, 0)$ which is a valid query *if* 0 is allowed. In return, $\mathcal{A}$ is provided the private key of $d_{\mathsf{v}} = (d_0, d_1, \ldots)$. Then $d_0 = \alpha P_2 + r(\mathsf{v}_1^* Q_1, \ldots, \mathsf{v}_k^* Q_k + 0 \cdot Q_{k+1} + P_3)$ and $d_1 = rP$ for some random $r \in \mathbb{Z}_p$. Using (d_0, d_1), $\mathcal{A}$ can decrypt any message encrypted for v^*. Removing 0 from the identity space avoids this situation and allows a proof of the BBG-HIBE in the sID model.

Modified Reduction

We modify the security reduction of BBG-HIBE in the following way. Suppose, as before the adversary committed to an identity tuple $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_{h'}^*)$ in the commitment stage. During setup, $\mathcal{B}$ choses a random u from $\{1, \ldots, h'\}$ and forms the public parameters as in the original reduction *assuming* that $\mathsf{v}^+ = (\mathsf{v}_1^*, \ldots \mathsf{v}_u^*)$ will be the target identity in challenge stage. This means that during setup, the simulator augments v^+ by appending zeros and forming a tuple of length h.

The above change does not affect the simulator's ability to answer key extraction queries. In the challenge phase, $\mathcal{B}$ can form a proper encryption *only if* the target identity tuple is v^+. The target identity submitted by the adversary should be a prefix of v^*. If this is not equal to v^+ then $\mathcal{B}$ aborts the game and outputs one with probability half. Otherwise, it returns a proper challenge as in the original reduction, provided the input instance is a true h-wDBDHI* tuple.

Since, $1 \leq u \leq h' \leq h$ and u is chosen uniformly at random, we have $\Pr[\overline{\mathsf{abort}}] \geq 1/h$. This leads to a multiplicative degradation by a factor of h, i.e., $\epsilon \leq h\epsilon'$, where ϵ is $\mathcal{A}$'s advantage against BBG-HIBE and ϵ' is the advantage of solving h-wDBDHI*.

9.3 Constant Size Ciphertext HIBE in Selective$^+$-ID Model

We augment the BBG-HIBE to obtain a new constant size ciphertext HIBE secure in the *selective*$^+$-ID model *without* any degradation. We call this new protocol $\mathcal{G}_1$. The basic intuition is to replace P_3 in BBG-HIBE by a vector $\overrightarrow{P}_3 = (P_{3,1}, \ldots, P_{3,h})$ where $P_{3,i}$ corresponds to the ith level of the HIBE.

9.3.1 Construction

Let the maximum height of the HIBE be h. The identities at a depth $u \leq h$ are of the form $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_u) \in \mathbb{Z}_p^u$. Note that, we also allow 0 as a valid identity component. Messages are elements of G_2.

Setup: Let $\langle P \rangle = G_1$. Choose a random $\alpha \in \mathbb{Z}_p$ and set $P_1 = \alpha P$. Choose a random element $P_2 \in G_1$ and two random h length vectors $\overrightarrow{P}_3, \overrightarrow{Q} \in G_1^h$ where $\overrightarrow{P}_3 = (P_{3,1}, \ldots, P_{3,h})$ and $\overrightarrow{Q} = (Q_1, \ldots, Q_h)$. Set the public parameter as $\mathsf{PP} = (P, P_1, P_2, \overrightarrow{P}_3, \overrightarrow{Q})$ while the master key is $P_4 = \alpha P_2$. Instead of P_1, P_2, $e(P_1, P_2)$ can also be kept as part of PP. This avoids the pairing computation during encryption.

Key-Gen: Given an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k) \in (\mathbb{Z}_p)^k$ of depth $k \leq h$, pick a random $r \in \mathbb{Z}_p$ and output

$$d_{\mathsf{v}} = \left(\alpha P_2 + r \sum_{j=1}^{k} V_j,\ rP,\ rP_{3,k+1}, \ldots, rP_{3,h},\ rQ_{k+1}, \ldots, rQ_h \right)$$

where $\boxed{V_j = P_{3,j} + \mathsf{v}_j Q_j}$. The private key at level k consists of $2(h-k+1)$ elements of G_1. Among these $2(h-k+1)$ elements only the first two are required in decryption, the rest are used to generate the private key for the next level as follows:
Let the secret key corresponding to the identity $\mathsf{v}_{|k-1} = (\mathsf{v}_1, \ldots, \mathsf{v}_{k-1})$ be

$$d_{\mathsf{v}_{|k-1}} = (A_0, A_1, B_k, \ldots, B_h, C_k, \ldots, C_h)$$

where $A_0 = \alpha P_2 + r' \sum_{j=1}^{k-1} V_j$, $A_1 = r'P$, and for $k \leq j \leq h$, $B_j = r'P_{3,j}$, $C_j = r'Q_j$. Pick a random $r^* \in \mathbb{Z}_p$ and compute

$$\begin{aligned} d_{\mathsf{v}} \ = \ & (A_0 + B_k + \mathsf{v}_k C_k + r^* \textstyle\sum_{j=1}^{k} V_k,\ A_1 + r^* P, \\ & B_{k+1} + r^* P_{3,k+1}, \ldots, B_h + r^* P_{3,h}, \\ & C_{k+1} + r^* Q_{k+1}, \ldots, C_h + r^* Q_h). \end{aligned}$$

If we put $r = r' + r^*$, then d_{v} is a proper private key for $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k)$.

Encrypt: To encrypt $M \in G_2$ under the identity $(\mathsf{v}_1, \ldots, \mathsf{v}_k) \in (\mathbb{Z}_p)^k$, pick a random $s \in \mathbb{Z}_p$ and output

$$\mathsf{CT} = \left(e(P_1, P_2)^s \times M, sP, s \left(\sum_{j=1}^{k} V_j \right) \right)$$

where V_j is as defined in Key Generation.

Decrypt: To decrypt $\mathsf{CT} = (A, B, C)$ using the private key $d_{\mathsf{v}} = (d_0, d_1, \ldots)$ of $\mathsf{v} = \mathsf{v}_1, \ldots, \mathsf{v}_k)$, compute

$$A \times \frac{e(d_1, C)}{e(B, d_0)} = e(P_1, P_2)^s \times M \times \frac{e\left(rP, s \sum_{j=1}^{k} V_j\right)}{e\left(sP, xP_2 + r \sum_{j=1}^{k} V_j\right)} = M.$$

Table 9.1: Comparison of BBG-HIBE and $\mathcal{G}_1$.

Protocol/ Sec. Model	id comp	PP size	max pvt key size	dec. subkey/ CT expansion	enc. eff.	dec. eff.	security degradation
$\mathcal{G}_1$ in s$^+$ID	$\mathbb{Z}_p$	$3+2h$	$2h$	2	$h+2$	2	none
BBG in s$^+$ID	$\mathbb{Z}_p^*$	$4+h$	$h+1$	2	$h+2$	2	h
BBG in sID	$\mathbb{Z}_p^*$	$4+h$	$h+1$	2	$h+2$	2	none

The columns PP *size, max pvt key size, dec. subkey/CT expansion stand for the number of elements of G_1 in public parameter, private key, decryption subkey and ciphertext expansion respectively. The column enc. eff stands for the number of scalar multiplications in G_1 during encryption while the column dec. eff denotes the number of pairing computations during decryption. All values are for a HIBE of maximum height h.*

Comparison to BBG protocol

The protocol $\mathcal{G}_1$ is a modification of the BBG-HIBE with a different $P_{3,i}$ for each level of the HIBE. This is required to get a proof of security in the augmented s$^+$-ID model without any security degradation. This reduction is provided in the next section. Additionally, it allows identity components to be elements of $\mathbb{Z}_p$, instead of $\mathbb{Z}_p^*$ as in BBG-HIBE. On the other hand, this modification only affects the efficiency of the BBG-HIBE in a small way. The first thing to note is the size of the ciphertext is still constant (three elements). Secondly, the size of the public parameter as well as private key is linear in the length of the HIBE and private key size decreases as we "go down" the HIBE. These two properties ensure that the applications mentioned in [19] also hold for the new HIBE described above. In particular, it is possible to combine the new HIBE with the BB-HIBE of [17] to get an intermediate HIBE with controllable trade-off between the size of the ciphertext and the size of the private key. Further, the application to the construction of forward secure encryption protocol mentioned in [19] can also be done with the new HIBE.

9.3.2 Security

Semantic security (i.e., (CPA-security) of the above scheme in the s$^+$-ID model is proved under the h-wDBDHI* assumption.

THEOREM 9.3.1. *Let (t', ϵ, h)-wDBDHI* assumption holds in $\langle G_1, G_2, e() \rangle$. Then the HIBE $\mathcal{G}_1$ is (t, q, ϵ)-*IND-sID-CPA *secure for $q \geq 1$, $t' \approx t + O(\tau q)$ where τ is the time for a scalar multiplication in G_1.*

Proof : Suppose $\mathcal{A}$ is a (t, q, ϵ)-CPA adversary for the $\mathcal{G}_1$, then we construct an algorithm $\mathcal{B}$ that solves the h-wDBDHI* problem. $\mathcal{B}$ takes as input a tuple $\langle P, Q, Y_1, \ldots, Y_h, T \rangle$ where

$Y_i = \alpha^i P$ for some random $\alpha \in \mathbb{Z}_p^*$ and T is either equal to $e(P,Q)^{\alpha^{h+1}}$ or a random element of G_2. We define the s$^+$ID game between $\mathcal{B}$ and $\mathcal{A}$ as follows.

Initialization: $\mathcal{A}$ outputs an identity tuple $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_u^*) \in (\mathbb{Z}_p)^u$ for any $u \leq h$. The restriction on $\mathcal{A}$ is that it cannot ask for the private key of v^* or any of its prefixes and in challenge stage it asks for an encryption under v^* or any of its prefixes. In case $u < h$, $\mathcal{B}$ chooses random $\mathsf{v}_{u+1}^*, \ldots, \mathsf{v}_h^*$ from $\mathbb{Z}_p$ and keeps these extra elements to itself. (Note that $\mathcal{B}$ is *not* augmenting the target identity to create a new target identity.)

Setup: $\mathcal{B}$ picks random β, $\beta_1, \ldots, \beta_h$ and $c_1, \ldots, c_h$ in $\mathbb{Z}_p$. It then sets

$$\begin{array}{ll} P_1 = Y_1 = \alpha P; & P_2 = Y_h + \beta P = (\alpha^h + \beta)P; \text{ and for } 1 \leq j \leq u, \\ Q_j = \beta_j P - Y_{h-j+1}; & P_{3,j} = c_j P + \mathsf{v}_j^* Y_{h-j+1}; \text{ and for } u < j \leq h, \\ Q_j = \beta_j P; & P_{3,j} = c_j P + \mathsf{v}_j^* Y_{h-j+1}. \end{array}$$

$\mathcal{B}$ declares the public parameter as $(P, P_1, P_2, \overrightarrow{P}_3, \overrightarrow{Q})$, where $\overrightarrow{Q} = (Q_1, \ldots, Q_h)$, $\overrightarrow{P}_3 = (P_{3,1}, \ldots, P_{3,h})$. The corresponding master key $\alpha P_2 = Y_{h+1} + \beta Y_1$ is unknown to $\mathcal{B}$. $\mathcal{B}$ defines the functions $F_j = \mathsf{v}_j^* - \mathsf{v}_j$ for $1 \leq j \leq u$ and $F_j = \mathsf{v}_j^*$ for $u < j \leq h$ and $J_j = c_j + \beta_j \mathsf{v}_j$ for $1 \leq j \leq h$.

Phase 1: Suppose $\mathcal{A}$ asks for the private key corresponding to an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_m)$ for $m \leq h$. Note that for any $j \leq u$,

$$\begin{aligned} V_j &= P_{3,j} + \mathsf{v}_j Q_j \\ &= c_j P + \mathsf{v}_j^* Y_{h-j+1} + \mathsf{v}_j(\beta_j P - Y_{h-j+1}) \\ &= (\mathsf{v}_j^* - \mathsf{v}_j) Y_{h-j+1} + (c_j + \beta_j \mathsf{v}_j) P \\ &= F_j Y_{h-j+1} + J_j P. \end{aligned}$$

Similarly, for $u < j \leq h$

$$V_j = P_{3,j} + \mathsf{v}_j Q_j = c_j P + \mathsf{v}_j^* Y_{h-j+1} + \mathsf{v}_j \beta_j P = F_j Y_{h-j+1} + J_j P.$$

Hence, V_j for $1 \leq j \leq h$ is computable from what is known to $\mathcal{B}$.

Recall that u is the length of v^* that the adversary committed to before the set-up phase. If $m \leq u$, then there must be a $k \leq \tau$ such that $F_k \neq 0$, as otherwise the queried identity is a prefix of the target identity. In case $m > u$, it is possible that $F_1 = \cdots = F_u = 0$. Then by construction, $F_{u+1} \neq 0$. Let k be the smallest in $\{1, \ldots, m\}$ such that $F_k \neq 0$. $\mathcal{B}$ picks a random $r \in \mathbb{Z}_p$ and assigns $d_{0|k} = (-J_k/F_k)Y_k + \beta Y_1 + rV_k$ and $d_1 = (-1/F_k)Y_k + rP$. Now,

$$d_{0|k} = -\frac{J_k}{F_k} Y_k + \beta Y_1 + \alpha^k Y_{h-k+1} - \alpha^k \frac{F_k}{F_k} Y_{h-k+1} + rV_k = \alpha P_2 + \tilde{r} V_k$$

where $\tilde{r} = (r - \frac{\alpha^k}{F_k})$. Also $d_1 = -\frac{1}{F_k} Y_k + rP = -\frac{\alpha^k}{F_k} P + rP = \tilde{r}P$. For any $j \in \{1, \ldots, m\} \setminus \{k\}$ we have

$$\begin{aligned} \tilde{r}V_j &= (r - \frac{\alpha^k}{F_k})(F_j Y_{h-j+1} + J_j P) \\ &= r(F_j Y_{h-j+1} + J_j P) - \frac{1}{F_k}(F_j Y_{h+k-j+1} + J_j Y_k). \end{aligned}$$

For $j < k$, $F_j = 0$, so $\mathcal{B}$ can compute all these $\tilde{r}V_j$s from what it has. It forms

$$d_0 = d_{0|k} + \sum_{j \in \{1,\ldots,m\} \setminus \{k\}} \tilde{r}V_j = \alpha P_2 + \tilde{r} \sum_{j=1}^{m} V_j.$$

To form a valid private key $\mathcal{B}$ also needs to compute $\tilde{r}P_{3,j}$ and $\tilde{r}Q_j$ for $m < j \leq h$. Now,

$$\begin{aligned} \tilde{r}P_{3,j} &= \left(r - \frac{\alpha^k}{F_k}\right)(c_j P + \mathsf{v}_j^* Y_{h-j+1}) \\ &= r(c_j P + \mathsf{v}_j^* Y_{h-j+1}) - \frac{1}{F_k}\left(c_j Y_k + \mathsf{v}_j^* Y_{h+k-j+1}\right); \end{aligned}$$

For $j \leq u$,

$$\tilde{r}Q_j = \left(r - \frac{\alpha^k}{F_k}\right)(\beta_j P - Y_{h-j+1}) = r(\beta_j P - Y_{h-j+1}) - \frac{1}{F_k}(\beta_j Y_k - Y_{h+k-j+1})$$

and for $u < j \leq h$,

$$\tilde{r}Q_j = \left(r - \frac{\alpha^k}{F_k}\right)\beta_j P = r\beta_j P - \frac{1}{F_k}\beta_j Y_k.$$

All these values are computable from what is known to $\mathcal{B}$. Hence, $\mathcal{B}$ forms the private key as:

$$d_{\mathsf{v}} = (d_0, d_1, \tilde{r}P_{3,m+1}, \ldots, \tilde{r}P_{3,h}, \tilde{r}Q_{m+1}, \ldots, \tilde{r}Q_h)$$

and provides it to $\mathcal{A}$.

Challenge: After completion of Phase 1, $\mathcal{A}$ outputs two messages $M_0, M_1 \in G_2$ on which it wishes to be challenged and $\mathsf{v}^+ = \mathsf{v}_1^*, \ldots, \mathsf{v}_{u'}^*$ where $u' \leq u \leq h$. $\mathcal{B}$ picks a random $b \in \{0, 1\}$ and provides $\mathcal{A}$ the challenge ciphertext

$$\mathsf{CT} = \left(M_b \times T \times e(Y_1, \beta Q),\ Q,\ \left(\sum_{j=1}^{u'}(c_j + \beta_j \mathsf{v}_j^*)\right) \times Q\right).$$

Suppose, $Q = \gamma P$ for some unknown $\gamma \in \mathbb{Z}_p$. Then

$$\left(\sum_{j=1}^{u'} c_j + \beta_j \mathsf{v}_j^*\right) \times Q = \gamma \left(\sum_{j=1}^{u'} c_j + \beta_j \mathsf{v}_j^*\right) P$$

$$
\begin{aligned}
&= \gamma \sum_{j=1}^{u'} \left(c_j P + \mathsf{v}_j^* Y_{h-j+1} + \mathsf{v}_j^* (\beta_j P - Y_{h-j+1}) \right) \\
&= \gamma \sum_{j=1}^{u'} \left(P_{3,j} + \mathsf{v}_j^* Q_j \right) \\
&= \gamma \left(\sum_{j=1}^{u'} V_j \right).
\end{aligned}
$$

If the input provided to $\mathcal{B}$ is a true h-wDBDHI* tuple, i.e., $T = e(P,Q)^{(\alpha^{h+1})}$, then

$$T \times e(Y_1, \beta Q) = e(P,Q)^{(\alpha^{h+1})} \times e(Y_1, \beta Q) = e(Y_h + \beta P, Q)^{\alpha} = e(P_1, P_2)^{\gamma}.$$

So, the challenge ciphertext is

$$\mathsf{CT} = \left(M_b \times e(P_1, P_2)^{\gamma}, \gamma P, \gamma \left(\sum_{j=1}^{u'} V_j \right) \right).$$

CT is a valid encryption of M_b under $\mathsf{v}^+ = (\mathsf{v}_1^*, \ldots, \mathsf{v}_{u'}^*)$. On the other hand, when T is random, CT is random from the view point of $\mathcal{A}$.

Phase 2: This is similar to Phase 1. Note that $\mathcal{A}$ places at most q queries in Phase 1 and 2 together.

Guess: Finally, $\mathcal{A}$ outputs its guess $b' \in \{0,1\}$. $\mathcal{B}$ outputs $1 \oplus b \oplus b'$.

$\mathcal{A}$'s view in the above simulation is identical to that in a real attack. This gives us the required bound on the advantage of the adversary in breaking the HIBE protocol. ■

9.4 Augmenting to $\boldsymbol{\mathcal{M}}_2^+$

Like the augmentation of the *selective*-ID model to selective$^+$-ID model, we can augment $\boldsymbol{\mathcal{M}}_2$ proposed in Chapter 7 in an obvious way to $\boldsymbol{\mathcal{M}}_2^+$. Suppose the adversary of an h-HIBE has committed to a set of target identities, $\mathcal{I}_1^*, \ldots, \mathcal{I}_u^*$ where $u \leq h$. Then in the challenge phase it outputs a target identity $\mathsf{v}_1^*, \ldots, \mathsf{v}_{u'}^*$ where $1 \leq u' \leq u$ and each $\mathsf{v}_j^* \in \mathcal{I}_j^*$.

The HIBE $\mathcal{H}_2$ proposed in Chapter 7 is also secure in $\boldsymbol{\mathcal{M}}_2^+$. ccHIBE of Chapter 8 secure in $\boldsymbol{\mathcal{M}}_2$ can be proved to be secure in $\boldsymbol{\mathcal{M}}_2^+$ with a multiplicative security degradation of h. Here, we show how $\mathcal{G}_1$ can be augmented to $\boldsymbol{\mathcal{M}}_2^+$.

9.4.1 Construction

We augment $\mathcal{G}_1$ to obtain security in model $\boldsymbol{\mathcal{M}}_2^+$ and call this new protocol $(h, n_1, \ldots, n_h)$-$\mathcal{G}_2$ or simply $\mathcal{G}_2$.

The maximum height of the HIBE be h. The identities at a depth $u \leq h$ are of the form $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_u) \in (\mathbb{Z}_p)^u$. Messages are elements of G_2.

Setup: Let $\langle P \rangle = G_1$. Choose a random $\alpha \in \mathbb{Z}_p$ and set $P_1 = \alpha P$. Choose a random element $P_2 \in G_1$ and a random h length vector $\overrightarrow{P}_3 = (P_{3,1}, \ldots, P_{3,h})$, where each $P_{3,i} \in G_1$. Also choose random vectors $\overrightarrow{Q}_1, \ldots, \overrightarrow{Q}_h$ where each $\overrightarrow{Q}_i$ consists of n_i elements of G_1. Set the public parameter as $\mathsf{PP} = (P, P_1, P_2, \overrightarrow{P}_3, \overrightarrow{Q}_1, \ldots, \overrightarrow{Q}_h)$ while the master key is $P_4 = \alpha P_2$. Instead of P_1, P_2, $e(P_1, P_2)$ can also be kept as part of PP. This avoids the pairing computation during encryption.

Note that, while the original BBG scheme and ccHIBE of Chapter 8 had a single element P_3 in the public parameter, we have a vector $\overrightarrow{P}_3$ of length h.

Key-Gen: Let, $V(i, y) = y^{n_i} Q_{i,n_i} + \cdots + y Q_{i,1}$ for any $y \in \mathbb{Z}_p$. Given an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k) \in \mathbb{Z}_p^k$ of depth $k \leq h$, pick a random $r \in \mathbb{Z}_p$ and output

$$d_{\mathsf{v}} = \left(\alpha P_2 + r \sum_{j=1}^{k} V_j,\ rP,\ rP_{3,k+1}, \ldots, rP_{3,h},\ r\overrightarrow{Q}_{k+1}, \ldots, r\overrightarrow{Q}_h \right)$$

where $\boxed{V_j = P_{3,j} + V(j, \mathsf{v}_j)}$. The private key at level k consists of $(2 + h - k + \sum_{i=k+1}^{h} n_i)$ elements of G_1. Among these, only the first two are required in decryption, the rest are used to generate the private key for the next level as follows:
Let the secret key corresponding to the identity $\mathsf{v}_{|k-1} = (\mathsf{v}_1, \ldots, \mathsf{v}_{k-1})$ be

$$d_{\mathsf{v}_{|k-1}} = (A_0, A_1, B_k, \ldots, B_h, \overrightarrow{C}_k, \ldots, \overrightarrow{C}_h)$$

where $A_0 = \alpha P_2 + r' \sum_{j=1}^{k-1} V_j$, $A_1 = r'P$, and for $k \leq j \leq h$, $B_j = r'P_{3,j}$, $\overrightarrow{C}_j = r'Q_{j,1}, \ldots, r'Q_{j,n_j} = \langle C_{j,n_j} \rangle$ Pick a random $r^* \in \mathbb{Z}_p$ and compute

$$\begin{aligned} d_{\mathsf{v}} \ = \ & (A_0 + B_k + \textstyle\sum_{i=1}^{n_k} \mathsf{v}_k^i C_{k,i} + r^* \sum_{j=1}^{k} V_j,\ A_1 + r^* P, \\ & B_{k+1} + r^* P_{3,k+1}, \ldots, B_h + r^* P_{3,h}, \\ & \overrightarrow{C}_{k+1} + r^* \overrightarrow{Q}_{k+1}, \ldots, \overrightarrow{C}_h + r^* \overrightarrow{Q}_h). \end{aligned}$$

If we put $r = r' + r^*$, then d_{v} is a proper private key for $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k)$.

Encrypt: To encrypt $M \in G_2$ under the identity $(\mathsf{v}_1, \ldots, \mathsf{v}_k) \in (\mathbb{Z}_p)^k$, pick a random $s \in \mathbb{Z}_p$ and output

$$\mathsf{CT} = \left(e(P_1, P_2)^s \times M, sP, s \left(\sum_{j=1}^{k} V_j \right) \right)$$

where V_j is as defined in Key Generation.

Decrypt: To decrypt $\mathsf{CT} = (A, B, C)$ using the private key $d_{\mathsf{v}} = (d_0, d_1, \ldots)$ of $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_k)$, compute

$$A \times \frac{e(d_1, C)}{e(B, d_0)} = e(P_1, P_2)^s \times M \times \frac{e\left(rP, s\sum_{j=1}^{k} V_j\right)}{e\left(sP, \alpha P_2 + r\sum_{j=1}^{k} V_j\right)} = M.$$

9.4.2 Security

Semantic security (i.e., CPA-security) of the above scheme in model $\boldsymbol{\mathcal{M}}_2^+$ is proved under the h-wDBDHI* assumption. Note that, the additional flexibility in terms of choosing the target identity that we allowed to the adversary in the s$^+$ID model is also applicable here.

THEOREM 9.4.1. *Let $n_1, \ldots, n_h, q$ and $n'_1, \ldots, n'_h$ be two sets of positive integers with $n'_i \leq n_i$ for $1 \leq i \leq h$. Then for $t \geq 1, q \geq 1$*

$$\mathsf{Adv}^{(h,n_1,\ldots,n_h)\text{-}\mathcal{G}_2}_{(h,n'_1,\ldots,n'_h)\text{-}\boldsymbol{\mathcal{M}}_2^+}(t, q) \leq \mathsf{Adv}^{\mathsf{h\text{-}wDBDHI}^*}(t + O(\tau n q))$$

where $n = \sum_{i=1}^{h} n_i$.

Proof : Suppose $\mathcal{A}$ is a (t, q)-CPA adversary for $\mathcal{G}_2$, then we construct an algorithm $\mathcal{B}$ that solves the h-wDBDHI* problem. $\mathcal{B}$ takes as input a tuple $\langle P, Q, Y_1, \ldots, Y_h, T\rangle$ where $Y_i = \alpha^i P$ for some random $\alpha \in \mathbb{Z}_p^*$ and T is either equal to $e(P, Q)^{\alpha^{h+1}}$ or a random element of G_2. We define the modified $\boldsymbol{\mathcal{M}}_2^+$ game between $\mathcal{B}$ and $\mathcal{A}$ as follows.

Initialization: $\mathcal{A}$ outputs sets of target identities for each level of the HIBE as $(\mathcal{I}_1^*, \ldots, \mathcal{I}_u^*)$ where each $\mathcal{I}_i^*$ is a set of cardinality n'_i for any $u \leq h$.

Setup: $\mathcal{B}$ defines polynomials $F_1(x), \ldots, F_h(x)$ where for $1 \leq i \leq u$,

$$\begin{aligned} F_i(x) &= \prod_{\mathsf{v} \in \mathcal{I}_i^*} (x - \mathsf{v}) \\ &= x^{n'_i} + a_{i,n'_i-1}x^{n'_i-1} + \ldots + a_{i,1}x + a_{i,0} \end{aligned}$$

For $u < i \leq h$, define $F_i(x) = a_{i,0}$ where $a_{i,0}$ is a random element of $\mathbb{Z}_p^*$. For $1 \leq i \leq u$, let $a_{i,n'_i} = 1$ and $a_{i,n_i} = \cdots = a_{i,n'_i+1} = 0$. For $u < i \leq h$ we set $n'_i = 0$ and $a_{i,n_i} = \cdots = a_{i,1} = 0$. For $1 \leq i \leq h$ define

$$J_i(x) = b_{i,n_i}x^{n_i} + b_{i,n_i-1}x^{n_i-1} + \ldots + b_{i,1}x + b_{i,0}$$

where $b_{i,j}$ are random elements of $\mathbb{Z}_p$. It then sets

$P_1 = Y_1 = \alpha P$; $P_2 = Y_h + \beta P = (\alpha^h + \beta)P$; and for $1 \leq i \leq h$, $1 \leq j \leq n_i$
$Q_{i,j} = b_{i,j}P + a_{i,j}Y_{h-i+1}$; $P_{3,j} = b_{i,0}P + a_{i,0}Y_{h-i+1}$.

$\mathcal{B}$ declares the public parameters to be

$$(P, P_1, P_2, \overrightarrow{P}_3, \overrightarrow{Q}_1, \ldots, \overrightarrow{Q}_h),$$

where $\overrightarrow{P}_3 = (P_{3,1}, \ldots, P_{3,h})$ and $\overrightarrow{Q}_i = (Q_{i,1}, \ldots, Q_{i,n_i})$. The corresponding master key $\alpha P_2 = Y_{h+1} + \beta Y_1$ is unknown to $\mathcal{B}$. The distribution of the public parameter is as expected by $\mathcal{A}$.

Phase 1: Suppose $\mathcal{A}$ asks for the private key corresponding to an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_{h'})$ for $h' \leq h$. Note that for any $i \leq \eta'$,

$$\begin{aligned} V_i &= P_{3,i} + \sum_{j=1}^{n_i} \mathsf{v}_i^j Q_{i,j} \\ &= b_{i,0}P + a_{i,0}Y_{h-i+1} + \sum_{j=1}^{n_i} \mathsf{v}_i^j (b_{i,j}P + a_{i,j}Y_{h-i+1}) \\ &= F_i(\mathsf{v}_i)Y_{h-i+1} + J_i(\mathsf{v}_i)P. \end{aligned}$$

Hence, V_i is computable from what is known to $\mathcal{B}$.

Recall that $\mathcal{A}$ initially committed to sets of identities up to level u before the set-up phase. If $h' \leq u$, then there must be a $k \leq h'$ such that $F_k(\mathsf{v}_k) \neq 0$, as otherwise $\mathsf{v}_j \in \mathcal{I}_j^*$ for each $j \in \{1, \ldots, h'\}$ – which the adversary is not allowed by the rules of the Game. In case $h' > u$, it is possible that $F_1(\mathsf{v}_1) = \cdots = F_u(\mathsf{v}_u) = 0$. Then by construction $F_{u+1} \neq 0$. So, in either case there is a k such that $F_k(\mathsf{v}_k) \neq 0$. Moreover, k is the first such index in the range $\{1, \ldots, h'\}$. $\mathcal{B}$ picks a random $r \in \mathbb{Z}_p$ and assigns $d_{0|k} = (-J_k(\mathsf{v}_k)/F_k(\mathsf{v}_k))Y_k + \beta Y_1 + rV_k$ and $d_1 = (-1/F_k(\mathsf{v}_k))Y_k + rP$. Now,

$$\begin{aligned} d_{0|k} &= -\frac{J_k(\mathsf{v}_k)}{F_k(\mathsf{v}_k)}Y_k + \beta Y_1 + \alpha^k Y_{h-k+1} - \alpha^k \frac{F_k(\mathsf{v}_k)}{F_k(\mathsf{v}_k)}Y_{h-k+1} + rV_k \\ &= -\frac{J_k(\mathsf{v}_k)}{F_k(\mathsf{v}_k)}\alpha^k P + \alpha P_2 - \alpha^k \frac{F_k(\mathsf{v}_k)}{F_k(\mathsf{v}_k)}Y_{h-k+1} + rV_k \\ &= \alpha P_2 + \tilde{r}V_k \end{aligned}$$

where $\tilde{r} = (r - \frac{\alpha^k}{F_k(\mathsf{v}_k)})$. Also $d_1 = -\frac{1}{F_k(\mathsf{v}_k)}Y_k + rP = -\frac{\alpha^k}{F_k(\mathsf{v}_k)}P + rP = \tilde{r}P$. For any $j \in \{1, \ldots, h'\} \setminus \{k\}$ we have

$$\begin{aligned} \tilde{r}V_j &= (r - \frac{\alpha^k}{F_k(\mathsf{v}_k)})(F_j(\mathsf{v}_j)Y_{h-j+1} + J_j(\mathsf{v}_j)P) \\ &= r(F_j(\mathsf{v}_j)Y_{h-j+1} + J_j(\mathsf{v}_j)P) - \frac{1}{F_k(\mathsf{v}_k)}(F_j(\mathsf{v}_j)Y_{h+k-j+1} + J_j(\mathsf{v}_j)Y_k). \end{aligned}$$

Recall that, k is the smallest in the range $\{1, \ldots, h'\}$, such that, $F_k(\mathsf{v}_k) \neq 0$. Hence, for $j < k$, $F_j(\mathsf{v}_j) = 0$ and $\tilde{r}V_j = rJ_j(\mathsf{v}_j)P - \frac{J_j(\mathsf{v}_j)Y_k}{F_k(\mathsf{v}_k)}$. For $j > k$, $Y_{h+k-j+1}$ varies between Y_1 to

Y_h. So $\mathcal{B}$ can compute all these $\tilde{r}V_j$s from the information it has. It forms

$$d_0 = d_{0|k} + \sum_{j \in \{1,\ldots,h'\} \setminus \{k\}} \tilde{r} V_j = \alpha P_2 + \tilde{r} \sum_{j=1}^{h'} V_j.$$

To form a valid private key, $\mathcal{B}$ also needs to compute $\tilde{r}P_{3,i}$ and $\tilde{r}\overrightarrow{Q}_i$ for $h' < i \leq h$. Now,

$$\begin{aligned}
\tilde{r}P_{3,i} &= \left(r - \frac{\alpha^k}{F_k(\mathsf{v}_k)}\right)(b_{i,0}P + a_{i,0}Y_{h-i+1}) \\
&= r(b_{i,0}P + a_{i,0}Y_{h-i+1}) - \frac{1}{F_k(\mathsf{v}_k)}\left(b_{i,0}Y_k + a_{j,0}Y_{h+k-i+1}\right); \\
\tilde{r}Q_{i,j} &= \left(r - \frac{\alpha^k}{F_k(\mathsf{v}_k)}\right)(b_{i,j}P + a_{i,j}Y_{h-i+1}) \\
&= r(b_{i,j}P + a_{i,j}Y_{h-i+1}) - \frac{1}{F_k(\mathsf{v}_k)}\left(b_{i,j}Y_k + a_{i,j}Y_{h+k-i+1}\right).
\end{aligned}$$

All these values are computable from what is known to $\mathcal{B}$. Hence, $\mathcal{B}$ forms the private key as:

$$d_{\mathsf{v}} = \left(d_0, d_1, \tilde{r}P_{3,\tau+1}, \ldots, \tilde{r}P_{3,h}, \tilde{r}\overrightarrow{Q}_{\tau+1}, \ldots, \tilde{r}\overrightarrow{Q}_h\right)$$

and provides it to $\mathcal{A}$.

Challenge: After completion of Phase 1, $\mathcal{A}$ outputs two messages $M_0, M_1 \in G_2$ together with a target identity $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_{u'}^*)$, $u' \leq u$, on which it wishes to be challenged. The constraint is each $\mathsf{v}_j^* \in \mathcal{I}_j^*$ and hence $F_j(\mathsf{v}_j^*) = 0$ for $1 \leq j \leq u' \leq u$. $\mathcal{B}$ picks a random $b \in \{0, 1\}$ and provides $\mathcal{A}$ the challenge ciphertext

$$\mathsf{CT} = \left(M_b \times T \times e(Y_1, \beta Q),\ Q,\ \left(\sum_{i=1}^{u'} J_i(\mathsf{v}_i^*)\right) \times Q\right).$$

Suppose, $Q = \gamma P$ for some unknown $\gamma \in \mathbb{Z}_p$. Then

$$\begin{aligned}
\sum_{j=1}^{u'} J_j(\mathsf{v}_j^*)Q &= \gamma \sum_{j=1}^{u'} \left(J_j(\mathsf{v}_j^*)P + F_j(\mathsf{v}_j^*)Y_{h-j+1}\right) \\
&= \gamma \left(\sum_{j=1}^{u'} V_j\right).
\end{aligned}$$

If the input provided to $\mathcal{B}$ is a true h-wDBDHI* tuple, i.e., $T = e(P, Q)^{(\alpha^{h+1})}$, then

$$T \times e(Y_1, \beta Q) = e(P, Q)^{(\alpha^{h+1})} \times e(Y_1, \beta Q) = e(Y_h + \beta P, Q)^{\alpha} = e(P_1, P_2)^{\gamma}.$$

So, the challenge ciphertext is

$$\mathsf{CT} = \left(M_b \times e(P_1, P_2)^{\gamma}, \gamma P, \gamma \left(\sum_{j=1}^{u'} V_j \right) \right).$$

CT is a valid encryption of M_b under $\mathsf{v}^* = (\mathsf{v}_1^*, \ldots, \mathsf{v}_{u'}^*)$. On the other hand, when T is random, CT is random from the view point of $\mathcal{A}$.

Phase 2: This is similar to Phase 1. Note that $\mathcal{A}$ places at most q queries in Phase 1 and 2 together.

Guess: Finally, $\mathcal{A}$ outputs its guess $b' \in \{0, 1\}$. $\mathcal{B}$ outputs $1 \oplus b \oplus b'$.

$\mathcal{A}$'s view in the above simulation is identical to that in a real attack. This gives us the required bound on the advantage of the adversary in breaking the HIBE protocol. ■

9.5 Composite Scheme

We have mentioned in Section 3.4 that Boneh-Boyen-Goh [19] proposed a "product" construction based on the BBG-HIBE and the BB-HIBE. A similar construction is possible based on the HIBE $\mathcal{G}_1$ of Section 9.3 and BB-HIBE. The resulting HIBE is secure in s^+ID model. On the other hand, in Chapter 7 we have presented a construction $\mathcal{H}_1$ which is secure in model $\boldsymbol{\mathcal{M}}_1$. This construction is in a sense an extension of the BB-HIBE. We propose a composite scheme based on $\mathcal{H}_1$ and $\mathcal{G}_2$ which we denote as (h, n)-$\mathcal{G}_3$ or simply $\mathcal{G}_3$.

The essential idea, as in [19] is to form a product of two HIBEs. For this we represent an identity tuple in the form of a matrix (say $\mathit{I\!I}$) having (a-priori) fixed number of columns, h. When we look at a row of $\mathit{I\!I}$, it forms a constant ciphertext HIBE, $\mathcal{H}$, while each row taken together as a single identity forms another HIBE, $\mathcal{H}'$. We obtain a product construction by instantiating $\mathcal{H}'$ to be $\mathcal{H}_1$ of Chapter 7 and $\mathcal{H}$ to be the constant size ciphertext HIBE $\mathcal{G}_2$ of Section 9.4. In this case, the components of the identity tuples are from $\mathbb{Z}_p$ and we obtain security in $\boldsymbol{\mathcal{M}}_1$. Since $\boldsymbol{\mathcal{M}}_1$ allows the target identity to be of any length up to the maximum height of the HIBE, the adversary has the flexibility to choose the length the target identity in the challenge phase.

9.5.1 Construction

Let the maximum depth of the HIBE be $h \leq \ell_1 \times \ell_2$. Here individual identity components are elements of $\mathbb{Z}_p$.

Setup: Let P be a generator of G_1. Choose a random secret $x \in \mathbb{Z}_p$ and set $P_1 = xP$. Randomly choose P_2; an $\ell_1 \times \ell_2$ matrix $\mathcal{R}$ where

$$\mathcal{R} = \begin{bmatrix} R_{1,1} & \cdots & R_{1,\ell_2} \\ \vdots & \vdots & \vdots \\ R_{\ell_1,1} & \cdots & R_{\ell_1,\ell_2} \end{bmatrix}$$

and ℓ_2 many vectors $\overrightarrow{U_1}, \ldots, \overrightarrow{U_{\ell_2}}$ from G_1, where each $\overrightarrow{U_i} = (U_{i,1}, \ldots, U_{i,n})$. The public parameters are $\langle P, P_1, P_2, \mathcal{R}, \overrightarrow{U_1}, \ldots, \overrightarrow{U_{\ell_2}} \rangle$, while the master secret is xP_2.

Key Generation: Given an identity $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_u)$ for any u, this algorithm generates the private key d_{v} of v as follows.

Let $u = k_1\ell_2 + k_2$ with $k_2 \in \{1, \ldots, \ell_2\}$. We represent v by a (possibly incomplete) $(k_1 + 1) \times \ell_2$ matrix $\mathcal{I}$ where the last row has k_2 elements. Choose $(k_1 + 1)$ many random elements $r_1, \ldots, r_{k_1}, r_{k_2} \in \mathbb{Z}_p$ and output

$$\begin{aligned} d_{\mathsf{ID}} = & \left(xP_2 + \sum_{i=1}^{k_1} r_i \left(\sum_{j=1}^{h} V_{i,j} + R_{i,j} \right) + r_{k_2} \left(\sum_{j=1}^{k_2} V_{k_1+1,j} + R_{k_1+1,j} \right), r_1P, \ldots, r_{k_1}P, r_{k_2}P, \right. \\ & \left. r_{k_2}R_{k_1+1,k_2+1}, \ldots, r_{k_2}R_{k_1+1,h}, r_{k_2}\overrightarrow{U_{k_2+1}}, \ldots, r_{k_2}\overrightarrow{U_{\ell_2}} \right) \end{aligned}$$

where $\boxed{V_{i,j} = \sum_{j=1}^{n} \mathsf{v}^j U_{i,j}}$ and $r_{k_2}\overrightarrow{U_i}$ denotes that each element of $\overrightarrow{U_i}$ is multiplied by the scalar r_{k_2}.

The private key of v can also be generated given the private key of $\mathsf{v}_{|_{u-1}} = \mathsf{v}_1, \ldots, \mathsf{v}_{u-1}$ as required. There are two cases to be considered.

Case 1: Suppose $u - 1 = k_1\ell_2 + \ell_2$, then

$$\begin{aligned} d_{\mathsf{v}_{|_{u-1}}} &= \left(xP_2 + \sum_{i=1}^{k_1+1} r_i \left(\sum_{j=1}^{h} V_{i,j} + R_{i,j} \right), r_1P, \ldots, r_{k_1}P, r_{k_1+1}P \right) \\ &= (a_0, a_1, \ldots, a_{k_1}, a_{k_1+1}) \quad (say) \end{aligned}$$

Choose a random $r^* \in \mathbb{Z}_p$ and form d_{v} as

$$d_{\mathsf{v}} = a_0 + r^*(V_{k_1+2,1} + R_{k_1+2,1}), a_1, \ldots, a_{k_1+1}, r^*P, r^*R_{k_1+2,2}, \ldots, r^*R_{k_1+2,h}, r^*\overrightarrow{U_2}, \ldots, r^*\overrightarrow{U_{\ell_2}}$$

Case 2: Let, $u - 1 = k_1\ell_2 + k_2'$ with $k_2' < \ell_2$ then,

$$\begin{aligned} d_{\mathsf{v}_{|_{u-1}}} = & \left(xP_2 + \sum_{i=1}^{k_1} r_i \left(\sum_{j=1}^{h} V_{i,j} + R_{i,j} \right) + r'_{k_2} \left(\sum_{j=1}^{k_2'} V_{k_1,j} + P_{k_1+1,j} \right), r_1P, \ldots, r_{k_1}P, r'_{k_2}P, \right. \\ & \left. r'_{k_2}R_{k_1+1,k_2'+1}, \ldots, r'_{k_2}R_{k_1+1,h}, r'_{k_2}\overrightarrow{U}_{k_2'+1}, \ldots, r'_{k_2}\overrightarrow{U}_{\ell_2} \right) \\ = & (a_0, a_1, \ldots, a_{k_1}, a_{k_1+1}, b_{k_2'+1}, \ldots, b_{\ell_2}, \overrightarrow{c}_{k_2'+1}, \ldots, \overrightarrow{c}_{\ell_2}) \quad (say) \end{aligned}$$

Choose a random $r^* \in \mathbb{Z}_p$ and form d_{v} as

$$\begin{aligned} d_{\mathsf{v}} &= a_0 + \sum_{j=1}^{n} \mathsf{v}_u^j c_{k_2'+1,j} + b_{k_2'+1,j} + r^* \left(\sum_{j=1}^{k_2'+1} V_{k_1+1,j} + P_{k_1+1,j} \right), a_1, \ldots, a_{k_1}, a_{k_1+1} + r^* P, \\ & b_{k_2'+2} + r^* R_{k_1+1,k_2'+2}, \ldots, b_{\ell_2} + r^* R_{k_1+1,\ell_2}, \overrightarrow{c}_{k_2'+2} + r^* \overrightarrow{U}_{k_2'+2}, \ldots, \overrightarrow{c}_{\ell_2} + r^* \overrightarrow{U}_{\ell_2} \end{aligned}$$

It can be verified that d_{v} is a proper private key for v.

Encrypt: To encrypt a message $M \in G_2$ under the public key $\mathsf{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_u)$ choose a random $s \in \mathbb{Z}_p$ and then the ciphertext is

$$C = \left(e(P_1, P_2)^s \times M, sP, s \sum_{j=1}^{\ell_2} (V_{1,j} + R_{1,j}), \ldots, s \left(\sum_{j=1}^{\ell_2} V_{k_1,j} + R_{k_1,j} \right), s \left(\sum_{j=1}^{k_2} V_{k_1+1,j} + P_{k_1+1,j} \right) \right)$$

where $V_{i,j}$ is as defined in Key Generation part.

Decrypt: Let $\mathsf{CT} = (A, B, C_1, \ldots, C_{k_1}, C_{k_1+1})$ be a cipher text and $\mathsf{v} = \mathsf{v}_1, \ldots, \mathsf{v}_u$ be the corresponding identity. Then we decrypt CT using $d_{\mathsf{ID}} = (d_0, d_1, \ldots, d_{k_1+1}, \ldots)$ as

$$A \times \frac{\prod_{i=1}^{k_1+1} (d_i, C_i)}{e(B, d_0)} = M.$$

9.5.2 Security

Security of the above hybrid construction in the generalised selective-ID model (h, n)-$\boldsymbol{\mathcal{M}}_1$ of Chapter 7 can be reduced from the hardness of ℓ_2-wDBDHI* problem. Here we give a brief sketch of the proof.

The simulator is provided with a tuple $\langle P, Q, Y_1, \ldots, Y_{\ell_2}, T \rangle \in G_1^{\ell_2+2} \times G_2$. It has to decide whether this is a proper ℓ_2-wDBDHI* instance or not.

Adversary's commitment: $\mathcal{A}$ commits to a set $\mathcal{I}^*$, where $|\mathcal{I}^*| = n$. The restriction on the adversary is that in the private key extraction query at least one component of the identity tuple should be outside $\mathcal{I}^*$; while in the challenge phase it asks for the encryption under an identity v^* all of whose components are from $\mathcal{I}^*$.

Set-up: The simulator defines

$$\begin{aligned} F(x) &= \prod_{\mathsf{v} \in \mathcal{I}^*} (x - \mathsf{v}) = a_n x^n + \cdots + a_1 x + a_0 \\ J_i^{(j)}(x) &= b_{i,n} x^n + \cdots + b_{i,1} x + b_{i,0}^{(j)} \text{ for } 1 \le i \le \ell_1, \quad 1 \le j \le \ell_2 \end{aligned}$$

where each $b_{i,j} \in \mathbb{Z}_p^*$. The simulator defines $P_1 = Y_1$, $P_2 = Y_{\ell_2} + \beta P$ in a similar manner as in the set-up of Section 9.3.2. It further defines $U_{i,j} = b_{i,j} P + a_i Y_{h-i+1}$ for $1 \le i \le \ell_2$, $1 \le j \le n$ and $R_{k,j} = b_{k,0}^{(j)} P + a_0 Y_{\ell_2 - j + 1}$ for $1 \le k \le \ell_1$, $1 \le j \le \ell_2$.

Phase 1: Suppose $\mathcal{A}$ asks for the private key of an identity $\mathsf{v} = \mathsf{v}_1, \ldots, \mathsf{v}_m$ where $m = k_1 \times \ell_2 + k_2$. The simulator first forms the $(k_1 + 1) \times \ell_2$ matrix $\mathbb{I}$ where v_1 is indexed as $\mathsf{v}_{1,1}$ and v_m as v_{k_1+1,k_2}. As per the rule of the game there is at least one identity, say v_j, such that $F(\mathsf{v}_j) \neq 0$. Suppose, v_j is indexed as k_1', k_2' in $\mathbb{I}$. Using the identity tuple in the k_1'-th row and the technique of Section 9.3.2 the simulator forms a private key for $(\mathsf{v}_{k_1',1}, \ldots, \mathsf{v}_{k_1',k_2'})$ as $a_0', a_{k_1'+1}, b_{k_2'+1}, \ldots, b_{\ell_2}, c_{k_2'+1}, \ldots, c_{\ell_2}$. It next chooses $r_1, \ldots, r_{k_1'} \in \mathbb{Z}_p$ and computes the private key for $\mathsf{v}_1, \ldots, \mathsf{v}_j$ as

$$\begin{aligned} d_0 &= a_0' + \sum_{i=1}^{k_1'} r_i \sum_{j=1}^{\ell_2} (V_{i,j} + R_{i,j}) \\ d_i &= r_i P \quad \textit{for} \quad 1 \leq i \leq k_i' \end{aligned}$$

$(d_0, d_1, \ldots, d_{k_i'}, a_{k_1'+1}, b_{k_2'+1}, \ldots, b_{\ell_2}, c_{k_2'+1}, \ldots, c_{\ell_2}$ is a proper private key for $\mathsf{v}_1, \ldots, \mathsf{v}_j$ from which the simulator forms a private key for v and gives it to $\mathcal{A}$.

Challenge: The challenge identity $\mathsf{v}^* = \mathsf{v}_1^*, \ldots, \mathsf{v}_u^*$ should have each $\mathsf{v}_j \in \mathcal{I}^*$ and hence $F(\mathsf{v}_j^*) = 0$ for $1 \leq j \leq u$. Based on this fact the simulator is able to form a proper encryption if the tuple provided to it is a true h-wDBDHI* instance.

This way we can relate the adversarial success in breaking the composite scheme with the simulator's success in solving the ℓ_2-wDBDHI* problem.

Note that, in the commitment stage we may give the adversary some more flexibility by allowing it to commit to sets of identities $\mathcal{I}_1^*, \ldots, \mathcal{I}_h^*$, where $\mathcal{I}_j^*$ corresponds to the commitment for the jth level of the constant size ciphertext HIBE. In this case the restrictions in $\boldsymbol{\mathcal{M}}_2$ regarding the private key queries and challenge generation apply. This added flexibility, however, does not affect the efficiency of the protocol.

9.6 Discussion

The private key corresponding to an identity in a HIBE has two roles. The first role is to enable decryption of a message encrypted using this identity, while the second role is to enable generation of lower level keys. Not all components of the private key are necessarily required for decryption, i.e., the decryption subkey can have less number of components than the whole private key. This has also been observed in [19] and in case of the BBG-HIBE, the decryption subkey consists of only two components. In $\mathcal{G}_1$ and $\mathcal{G}_2$, the decryption subkeys also consist of two components as in the BBG-HIBE. In $\mathcal{G}_3$ the size of the decryption subkey is reduced by a factor of h compared to the size of the decryption subkeys in $\mathcal{H}_1$.

Having a small decryption subkey is important, since the decryption subkey may need to be loaded on smart cards for frequent and online decryptions. This is achieved in all the HIBE constructions described in this chapter. On the other hand, the entire private key is required for key delegation to lower level entities. Key delegation is a relatively infrequent activity which will typically be done by an entity from a workstation. Storage in a workstation is less restrictive and a larger size private key required only for key delegation is more tolerable.

The size of the private key in the BBG-HIBE and $\mathcal{G}_1$ is proportional to the number of levels in the HIBE. For $\mathcal{G}_2$ this size is proportional to $n \times h$, where h is the number of levels of the HIBE and n is the maximum number of challenge identities that the adversary can commit to for any level. The size of the private key in $\mathcal{G}_3$ varies cyclically with the number of components j in the identity. Let $j = j_1 h + j_2$, where h is the number of levels in $\mathcal{H}_1$ used in the product construction and $j_2 \in \{1, \ldots, h\}$. The size of the private key then varies as $j_1 + n \times j_2$, where n is the number of elements in the set from which the adversary can construct the challenge identity. Since j_2 varies in a cyclic manner with period h, the size of the private key also shows a similar behaviour. (A similar behaviour is also shown by the size of the private key in the product construction in [19].) A modification of the protocols eliminates the dependence of the size of the private key on j_2. Suppose that key delegation is only allowed to be performed by the PKG and entities at levels $h, 2h, 3h, \ldots$. In this case, the size of the private key varies only with j_1 and in fact, the private key and the decryption subkey become identical.

9.7 Conclusion

In this chapter, we have augmented the selective-ID security model for hierarchical identity-based encryption by allowing the adversary some flexibility in choosing the target identity tuple during the challenge phase of the security reduction. We have denoted this model by selective$^+$-ID model (s$^+$ID model). The Boneh-Boyen HIBE satisfies this notion of security while the constant size ciphertext HIBE of Boneh, Boyen and Goh needs some modification in the security reduction to do so. This modification introduces a multiplicative security degradation. We have further augmented the BBG-HIBE to construct a new protocol secure in s$^+$ID model without any degradation which maintains all the attractive features of BBG-HIBE. We build on this new construction another constant size ciphertext HIBE. The security of our second construction is proved under a generalization of the selective-ID security model. Our third construction of HIBE is a "product" construction that allows a controllable trade-off between the ciphertext size and the private key size.

Chapter 10

Conclusion

We conclude the dissertation by summing-up our contribution. In the process we also mention some challenging open problems in the area.

All the identity-based encryption protocols described in this dissertation use bilinear pairing as the primary building block and pairing computation is the most computationally intensive part in these protocols. Protocol designers usually assume the bilinear map to be a black box. This, no doubt, helps in a modular description and hence, easier understanding of the protocols. However, when we want to implement a protocol then the issue of actually computing the pairing takes a pivotal role.

This way, the problem of efficient pairing computation is closely associated with the question of efficient design of pairing based protocols. In our study of efficient and secure construction of (H)IBE, we started with an efficient algorithm for the computation of (modified) Tate pairing over elliptic curves having embedding degree two. Using the Jacobian coordinates for representation of the elliptic curve points together with a technique of encapsulated computation, we are able to improve the efficiency of Tate pairing computation with respect to the previous works on the same type of curves.

We note that, there remain many issues to be resolved on the question of pairing computation [63]. Which curve (e.g., elliptic or hyperelliptic, super-singular or non super-singular), over small characteristic or large characteristic and which pairing (Weil, Tate, Eta or Ate) to use and under what conditions are some of the pertinent issues that will keep the researchers in the relevant field busy in the coming future.

The pairing computation time depends on the sizes of the groups G_1 and G_2 (here we concentrate on symmetric pairing only). The sizes of G_1 and G_2 in turn depends on what level of security (e.g., 80 bits or 128 bits etc.) we are interested in. If a protocol suffers from a degradation in the security reduction, then that degradation must be compensated by working in larger size groups. Otherwise, we may well end up with a protocol without any practical significance [62, 64].

Our contribution in protocol development start with a generalisation of Waters IBE, which is proved secure in the full model without random oracle. We do not just stop at the

generalisation and move on to address the issue of concrete security in the context of this IBE protocol. To the best of our knowledge, this kind of concrete security analysis in the IBE scenario was not much emphasised before.

Our next contribution is an extension of this IBE to HIBE. By reusing the public parameters, we are able to significantly reduce the size of the public parameter. The security reduction for the HIBE is not tight and suffers from a degradation along the line of the earlier works in the full model (with or without random oracle). We propose another HIBE secure in the full model in Chapter 8. This is a variant of constant size ciphertext HIBE originally proposed by Boneh, Boyen and Goh modified along the line of our generalisation of Waters suggestion in Chapter 5. In Table 10.1 we make a comparative study of these two HIBEs with earlier suggestions in the same security model. For all these HIBEs the security degradation is exponential in the number of levels. Consequently, construction of a HIBE in the full model where the security degradation is sub-exponential in the number of levels is an outstanding research problem in this area.

A weaker security model, called the selective-ID model has been suggested in the literature where it is possible to avoid the security degradation. BB-HIBE and BBG-HIBE are the only HIBE protocols proposed in this model. We augment this selective-ID security model by allowing some flexibility to the adversary in choosing the target identity and call this augmented version selective$^+$-ID model. We show that BB-HIBE can be easily proved to be secure in s$^+$ID model and modify the original security reduction of BBG-HIBE to achieve security in this model. The modified reduction of the BBG-HIBE introduces a multiplicative security degradation. Next, we make some augmentation in the BBG-HIBE to achieve security in s$^+$ID model without any degradation. This new constant size ciphertext HIBE secure in the s$^+$ID model is called $\mathcal{G}_1$. In Table 10.2 we make a comparison of $\mathcal{G}_1$, BBG-HIBE and BB-HIBE.

Though the selective-ID model for (H)IBE avoids security degradation, it is too restrictive on the adversary. This motivated us to propose a generalisation of this model where the adversary commits to sets of identities. In Chapter 7 we define two models $\boldsymbol{\mathcal{M}}_1$ and $\boldsymbol{\mathcal{M}}_2$ based on the adversarial behavior. Two HIBE protocols $\mathcal{H}_1$ and $\mathcal{H}_2$ secure respectively in $\boldsymbol{\mathcal{M}}_1$ and $\boldsymbol{\mathcal{M}}_2$ are also suggested. These protocols can be seen as augmentation of the BB-HIBE. We further propose a constant size ciphertext HIBE called ccHIBE by adapting the BBG-HIBE in $\boldsymbol{\mathcal{M}}_2$. Another constant size ciphertext HIBE $\mathcal{G}_2$ secure in the augmented (in the sense of s$^+$ID model) $\boldsymbol{\mathcal{M}}_2$ is also suggested. A comparative study of these HIBEs is given in Table 10.3

To conclude, the HIBE protocols proposed in the dissertation broaden the choice of a designer of identity-based encryption protocols. Several different options of secure and efficient HIBE protocols are now available to choose from. HIBE as a cryptographic primitive has several interesting applications. Exploring these applications further based on the HIBEs proposed here would be an interesting future work.

Table 10.1: Comparison of HIBE Protocols Secure in Full Model.

Protocol	PP size (elts. of G_1)	Pvt. Key size (elts. of G_1)	CT size (elts. of G_1)	Pairing	
				Enc.	Dec.
GS [49]	2	j	j	1	j
Waters [89]	$(n+1)h+3$	$j+1$	$j+1$	None	$j+1$
HIBE-spp	$h+l+3$	$j+1$	$j+1$	None	$j+1$
FullccHIBE	$(\ell+1)h+3$	$(\ell+1)(h-1)+2$	2	None	2

The columns for Pvt. Key size, ciphertext (CT) size and pairing in decryption corresponds to an identity at a level j, $1 \leq j \leq h$. The GS-HIBE uses random oracle in the security reduction while the other three do not.

Table 10.2: Comparison of HIBE protocols Secure in sID/s$^+$ID Model.

protocol	security model	id comp	public parameter	max pvt key size	decryption subkey size
$\mathcal{G}_1$	s$^+$ID	$\mathbb{Z}_p$	$3+2h$	$2h$	2
BBG	s$^+$ID	$\mathbb{Z}_p^*$	$4+h$	$h+1$	2
BBG	sID	$\mathbb{Z}_p^*$	$4+h$	$h+1$	2
BB	s$^+$ID	$\mathbb{Z}_p$	$3+h$	$h+2$	$h+2$

protocol	ciphertext expansion	encryption efficiency	decryption efficiency	Security degradation
$\mathcal{G}_1$	2	$h+2$	2	Nil
BBG in s$^+$ID	2	$h+2$	2	h
BBG in sID	2	$h+2$	2	Nil
BB	$h+1$	$2h+1$	$h+1$	Nil

For a HIBE of maximum height h, the columns for public parameter, max pvt key size, decryption subkey size and ciphertext expansion denote the number of elements of G_1, encryption efficiency denotes the number of scalar multiplications in G_1 and decryption efficiency denotes the number of pairing computations.

Table 10.3: Comparison of HIBE protocols Secure in Generalised Selective-ID Model.

protocol	id comp.	public parameter	max pvt key size	decryption subkey size
$\mathcal{H}_1$	$\mathbb{Z}_p$	$n+h+3$	$h+1$	$h+1$
$\mathcal{H}_2$	$\mathbb{Z}_p$	$3+(n+1)h$	$h+1$	$h+1$
ccHIBE	$\mathbb{Z}_p^*$	$4+nh$	$2+n(h-1)$	2
$\mathcal{G}_2$	$\mathbb{Z}_p$	$3+(n+1)h$	$h+1+n(h-1)$	2

protocol	ciphertext expansion	encryption efficiency	decryption efficiency	security model
$\mathcal{H}_1$	$h+1$	$h(n+1)+1$	$h+1$	$\mathcal{M}_1$
$\mathcal{H}_2$	$h+1$	$h(n+1)+1$	$h+1$	$\mathcal{M}_2$
ccHIBE	2	$nh+2$	2	$\boldsymbol{\mathcal{M}}_2$
$\mathcal{G}_2$	2	$nh+2$	2	$\mathcal{M}_2^+$

For a HIBE of maximum height h, the columns for public parameter, max pvt key size, decryption subkey size and ciphertext expansion denote the number of elements of G_1, encryption efficiency denotes the number of scalar multiplications in G_1 and decryption efficiency denotes the number of pairing computations.

Bibliography

[1] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable Encryption Revisited: Consistency Properties, Relation to Anonymous IBE, and Extensions. In Victor Shoup, editor, *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 205–222. Springer, 2005.

[2] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In David Naccache, editor, *CT-RSA*, volume 2020 of *Lecture Notes in Computer Science*, pages 143–158. Springer, 2001.

[3] Nuttapong Attrapadung, Benoit Chevallier-Mames, Jun Furukawa, Takeshi Gomi, Goichiro Hanaoka, Hideki Imai, and Rui Zhang. Efficient Identity-Based Encryption with Tight Security Reduction. Cryptology ePrint Archive, Report 2005/320, 2005. http://eprint.iacr.org/.

[4] Joonsang Baek, Reihaneh Safavi-Naini, and Willy Susilo. Efficient Multi-receiver Identity-Based Encryption and Its Application to Broadcast Encryption. In Serge Vaudenay, editor, *Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 380–397. Springer, 2005.

[5] Joonsang Baek and Yuliang Zheng. Identity-Based Threshold Decryption. In Feng Bao, Robert H. Deng, and Jianying Zhou, editors, *Public Key Cryptography*, volume 2947 of *Lecture Notes in Computer Science*, pages 262–276. Springer, 2004.

[6] Paulo S. L. M. Barreto. Pairing-Based Crypto Lounge – A compendium of bilinear pairing related works . Available at: http://paginas.terra.com.br/informatica/paulobarreto/pblounge.html.

[7] Paulo S. L. M. Barreto, Steven Galbraith, Colm O hEigeartaigh, and Michael Scott. Efficient Pairing Computation on Supersingular Abelian Varieties. Cryptology ePrint Archive, Report 2004/375, 2004. http://eprint.iacr.org/.

[8] Paulo S. L. M. Barreto, Hae Yong Kim, Ben Lynn, and Michael Scott. Efficient Algorithms for Pairing-Based Cryptosystems. In Moti Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 354–368. Springer, 2002.

[9] Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Constructing Elliptic Curves with Prescribed Embedding Degrees. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *SCN*, volume 2576 of *Lecture Notes in Computer Science*, pages 257–267. Springer, 2002.

[10] Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. On the Selection of Pairing-Friendly Groups. In Mitsuru Matsui and Robert J. Zuccherato, editors, *Selected Areas in Cryptography*, volume 3006 of *Lecture Notes in Computer Science*, pages 17–25. Springer, 2003.

[11] Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Efficient Implementation of Pairing-Based Cryptosystems. *J. Cryptology*, 17(4):321–334, 2004.

[12] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-Friendly Elliptic Curves of Prime Order. In Bart Preneel and Stafford E. Tavares, editors, *Selected Areas in Cryptography*, volume 3897 of *Lecture Notes in Computer Science*, pages 319–331. Springer, 2005.

[13] Rana Barua, Ratna Dutta, and Palash Sarkar. Extending Joux's Protocol to Multi Party Key Agreement (Extended Abstract). In Thomas Johansson and Subhamoy Maitra, editors, *INDOCRYPT*, volume 2904 of *Lecture Notes in Computer Science*, pages 205–217. Springer, 2003.

[14] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations Among Notions of Security for Public-Key Encryption Schemes. In Hugo Krawczyk, editor, *CRYPTO*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45. Springer, 1998.

[15] Ian F. Blake, Gadiel Seroussi, and Nigel Smart, editors. *Advances in Elliptic Curve Cryptography*. London Mathematical Society Lecture Note Series. Cambridge University Press, 2004.

[16] Alexandra Boldyreva. Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme. In Yvo Desmedt, editor, *Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2003.

[17] Dan Boneh and Xavier Boyen. Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. In Cachin and Camenisch [25], pages 223–238.

[18] Dan Boneh and Xavier Boyen. Secure Identity Based Encryption Without Random Oracles. In Franklin [42], pages 443–459.

[19] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical Identity Based Encryption with Constant Size Ciphertext. In Cramer [36], pages 440–456. Full version available at Cryptology ePrint Archive; Report 2005/015.

[20] Dan Boneh and Matthew K. Franklin. Identity-Based Encryption from the Weil Pairing. *SIAM J. Comput.*, 32(3):586–615, 2003. Earlier version appeared in the proceedings of CRYPTO 2001.

[21] Dan Boneh and Jonathan Katz. Improved Efficiency for CCA-Secure Cryptosystems Built Using Identity-Based Encryption. In Menezes [70], pages 87–103.

[22] Dan Boneh, Ben Lynn, and Hovav Shacham. Short Signatures from the Weil Pairing. In Colin Boyd, editor, *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer, 2001.

[23] Xavier Boyen, Qixiang Mei, and Brent Waters. Direct Chosen Ciphertext Security from Identity-Based Techniques. In Vijay Atluri, Catherine Meadows, and Ari Juels, editors, *ACM Conference on Computer and Communications Security*, pages 320–329. ACM, 2005.

[24] Xavier Boyen and Brent Waters. Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). Cryptology ePrint Archive, Report 2006/085, 2006. http://eprint.iacr.org/, To appear in CRYPTO 2006.

[25] Christian Cachin and Jan Camenisch, editors. *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*. Springer, 2004.

[26] Ran Canetti, Shai Halevi, and Jonathan Katz. A Forward-Secure Public-Key Encryption Scheme. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 255–271. Springer, 2003.

[27] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-Ciphertext Security from Identity-Based Encryption. In Cachin and Camenisch [25], pages 207–222.

[28] Sanjit Chatterjee and Palash Sarkar. Trading Time for Space: Towards an Efficient IBE Scheme with Short(er) Public Parameters in the Standard Model. In Dong Ho Won and Seungjoo Kim, editors, *ICISC*, volume 3935 of *Lecture Notes in Computer Science*, pages 424–440. Springer, 2005.

[29] Sanjit Chatterjee and Palash Sarkar. Augmented *Selective*-ID Security Model and Constant Size Ciphertext HIBE. Indian Statistical Institute, Technical Report No. ASD/2006/8, 2006.

[30] Sanjit Chatterjee and Palash Sarkar. Generalization of the Selective-ID Security Model for HIBE Protocols. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography*, volume 3958 of *Lecture Notes in Computer Science*, pages 241–256. Springer, 2006. Revised version available at Cryptology ePrint Archive, Report 2006/203.

[31] Sanjit Chatterjee and Palash Sarkar. HIBE with Short Public Parameters Secure in the Full Model Without Random Oracle. Indian Statistical Institute, Technical Report No. ASD/2006/6, 2006.

[32] Sanjit Chatterjee and Palash Sarkar. New Constructions of Constant Size Ciphertext HIBE Without Random Oracle. Indian Statistical Institute, Technical Report No. ASD/2006/7, 2006.

[33] Sanjit Chatterjee, Palash Sarkar, and Rana Barua. Efficient Computation of Tate Pairing in Projective Coordinate over General Characteristic Fields. In Choonsik Park and Seongtaek Chee, editors, *ICISC*, volume 3506 of *Lecture Notes in Computer Science*, pages 168–181. Springer, 2004.

[34] Clifford Cocks. An Identity Based Encryption Scheme Based on Quadratic Residues. In Bahram Honary, editor, *IMA Int. Conf.*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363. Springer, 2001.

[35] Jean-Sébastien Coron. On the exact security of full domain hash. In Mihir Bellare, editor, *CRYPTO*, volume 1880 of *Lecture Notes in Computer Science*, pages 229–235. Springer, 2000.

[36] Ronald Cramer, editor. *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*. Springer, 2005.

[37] Whitfield Diffie and Martin E. Hellman. New Directions in Cryptography. *IEEE Tr. Inf. Th.*, 22:644–654, 1976.

[38] Yevgeniy Dodis and Nelly Fazio. Public Key Broadcast Encryption for Stateless Receivers. In Joan Feigenbaum, editor, *Digital Rights Management Workshop*, volume 2696 of *Lecture Notes in Computer Science*, pages 61–80. Springer, 2002.

[39] Iwan M. Duursma and Hyang-Sook Lee. Tate Pairing Implementation for Hyperelliptic Curves $y^2 = x^p - x + d$. In Chi-Sung Laih, editor, *ASIACRYPT*, volume 2894 of *Lecture Notes in Computer Science*, pages 111–123. Springer, 2003.

[40] Kirsten Eisenträger, Kristin Lauter, and Peter L. Montgomery. Fast Elliptic Curve Arithmetic and Improved Weil Pairing Evaluation. In Joye [59], pages 343–354.

[41] Taher Elgamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[42] Matthew K. Franklin, editor. *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*. Springer, 2004.

[43] Gerhard Frey, Michael Müller, and Hans-Georg Rück. The Tate Pairing and the Discrete Logarithm Applied to Elliptic Curve Cryptosystems. *IEEE Transactions on Information Theory*, 45(5):1717–1719, 1999.

[44] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554. Springer, 1999.

[45] Steven D. Galbraith, Keith Harrison, and David Soldera. Implementing the Tate Pairing. In Claus Fieker and David R. Kohel, editors, *ANTS*, volume 2369 of *Lecture Notes in Computer Science*, pages 324–337. Springer, 2002.

[46] David Galindo. The Exact Security of Pairing Based Encryption and Signature Schemes. Based on a talk at Workshop on Provable Security, INRIA, Paris, 2004. Available from author's website.

[47] David Galindo. Boneh-Franklin Identity Based Encryption Revisited. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *ICALP*, volume 3580 of *Lecture Notes in Computer Science*, pages 791–802. Springer, 2005.

[48] Craig Gentry. Practical Identity-Based Encryption Without Random Oracles. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464. Springer, 2006.

[49] Craig Gentry and Alice Silverberg. Hierarchical ID-Based Cryptography. In Yuliang Zheng, editor, *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer, 2002.

[50] Shafi Goldwasser and Silvio Micali. Probabilistic Encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[51] Robert Granger, Dan Page, and Nigel Smart. High Security Pairing-based Cryptography Revisited. In Alcherio Martinoli, Riccardo Poli, and Thomas Stuetzle, editors, *ANTS*, volume 4096 of *Lecture Notes in Computer Science*, pages 480–494. Springer, 2006.

[52] Robert Granger, Dan Page, and Martijn Stam. On Small Characteristic Algebraic Tori in Pairing Based Cryptography. *LMS J. Comput. and Math.*, 9:64–85, 2006.

[53] Darrel Hankerson, Alfred Menezes, and Scott Vanstone. *Guide to Elliptic Curve Cryptography*. Springer, 2004.

[54] Florian Hess. Efficient Identity Based Signature Schemes Based on Pairings. In Kaisa Nyberg and Howard M. Heys, editors, *Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 310–324. Springer, 2002.

[55] Florian Hess, Nigel P. Smart, and Frederik Vercauteren. The Eta Pairing Revisited. Cryptology ePrint Archive, Report 2006/110, 2006. http://eprint.iacr.org/.

[56] Jeremy Horwitz and Ben Lynn. Toward Hierarchical Identity-Based Encryption. In Lars R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 466–481. Springer, 2002.

[57] Tetsuya Izu and Tsuyoshi Takagi. Efficient Computations of the Tate Pairing for the Large MOV Degrees. In Pil Joong Lee and Chae Hoon Lim, editors, *ICISC*, volume 2587 of *Lecture Notes in Computer Science*, pages 283–297. Springer, 2002.

[58] Antoine Joux. A One Round Protocol for Tripartite Diffie-Hellman. *J. Cryptology*, 17(4):263–276, 2004. Earlier version appeared in the proceedings of ANTS-IV.

[59] Marc Joye, editor. *Topics in Cryptology - CT-RSA 2003, The Cryptographers' Track at the RSA Conference 2003, San Francisco, CA, USA, April 13-17, 2003, Proceedings*, volume 2612 of *Lecture Notes in Computer Science*. Springer, 2003.

[60] Jonathan Katz and Nan Wang. Efficiency Improvements for Signature Schemes with Tight Security Reductions. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *ACM Conference on Computer and Communications Security*, pages 155–164. ACM, 2003.

[61] Neal Koblitz. Elliptic Curve Cryptosystems. *Math. of Compu.*, 48:203–209, 1987.

[62] Neal Koblitz and Alfred Menezes. Another Look at "Provable Security". Cryptology ePrint Archive, Report 2004/152, 2004. http://eprint.iacr.org/, To appear in J. Cryptology.

[63] Neal Koblitz and Alfred Menezes. Pairing-Based Cryptography at High Security Levels. In Nigel P. Smart, editor, *IMA Int. Conf.*, volume 3796 of *Lecture Notes in Computer Science*, pages 13–36. Springer, 2005.

[64] Neal Koblitz and Alfred Menezes. Another Look at "Provable Security" II. Cryptology ePrint Archive, Report 2006/229, 2006. http://eprint.iacr.org/.

[65] Thomas Kuhn. *Structure of Scientific Revolutions*. Chicago University Press, 1962.

[66] Soonhak Kwon. Efficient Tate Pairing Computation for Elliptic Curves over Binary Fields. In Colin Boyd and Juan Manuel González Nieto, editors, *ACISP*, volume 3574 of *Lecture Notes in Computer Science*, pages 134–145. Springer, 2005.

[67] Arjen K. Lenstra and Eric R. Verheul. Selecting Cryptographic Key Sizes. *J. Cryptology*, 14(4):255–293, 2001.

[68] Benoît Libert and Jean-Jacques Quisquater. Efficient Revocation and Threshold Pairing Based Cryptosystems. In Sergio Rajsbaum, editor, *PODC*, pages 163–171. ACM Press, 2003.

[69] Alfred Menezes. *Elliptic Curve Public Key Cryptosystems*. Springer, 1993.

[70] Alfred Menezes, editor. *Topics in Cryptology - CT-RSA 2005, The Cryptographers' Track at the RSA Conference 2005, San Francisco, CA, USA, February 14-18, 2005, Proceedings*, volume 3376 of *Lecture Notes in Computer Science*. Springer, 2005.

[71] Alfred Menezes, Tatsuaki Okamoto, and Scott A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions on Information Theory*, 39(5):1639–1646, 1993.

[72] Alfred Menezes, Paul Oorschot, and Scott Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[73] Victor S. Miller. Use of Elliptic Curves in Cryptography. In Hugh C. Williams, editor, *CRYPTO*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426. Springer, 1985.

[74] Victor S. Miller. Short Programs for Functions on Curves. Unpublished manuscript, 1986. http://crypto.stanford.edu/miller/miller.pdf.

[75] Victor S. Miller. The Weil Pairing, and Its Efficient Calculation. *J. Cryptology*, 17(4):235–261, 2004.

[76] Shigeo Mitsunari, Ryuichi Sakai, and Masao Kasahara. A New Traitor Tracing Scheme using Bilinear Map. *IEICE Trans. Fundamentals*, 84:1234–1243, 2001.

[77] David Naccache. Secure and Practical Identity-Based Encryption. Cryptology ePrint Archive, Report 2005/369, 2005. http://eprint.iacr.org/.

[78] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and Tracing Schemes for Stateless Receivers. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62. Springer, 2001.

[79] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[80] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems Based on Pairing. In *Symposium on Cryptography and Information Security – SCIS*, 2000. In Japanese, English version available from the authors.

[81] Palash Sarkar. Head: Hybrid encryption with delegated decryption capability. In Anne Canteaut and Kapalee Viswanathan, editors, *INDOCRYPT*, volume 3348 of *Lecture Notes in Computer Science*, pages 230–244. Springer, 2004.

[82] Michael Scott. Computing the Tate Pairing. In Menezes [70], pages 293–304.

[83] Michael Scott and Paulo S. L. M. Barreto. Compressed Pairings. In Franklin [42], pages 140–156.

[84] Adi Shamir. Identity-Based Cryptosystems and Signature Schemes. In G. R. Blakley and David Chaum, editors, *CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1984.

[85] Victor Shoup. Sequences of Games: a Tool for Taming Complexity in Security Proofs. Cryptology ePrint Archive, Report 2004/332, 2004. http://eprint.iacr.org/.

[86] Nigel P. Smart. An Identity Based Authenticated Key Agreement Protocol Based on the Weil Pairing. Cryptology ePrint Archive, Report 2001/111, 2001. http://eprint.iacr.org/.

[87] Nigel P. Smart. Access Control Using Pairing Based Cryptography. In Joye [59], pages 111–121.

[88] Eric R. Verheul. Evidence that XTR Is More Secure than Supersingular Elliptic Curve Cryptosystems. *J. Cryptology*, 17(4):277–296, 2004. Earlier version appeared in the Proceedings of Eurocrypt 2001.

[89] Brent Waters. Efficient Identity-Based Encryption Without Random Oracles. In Cramer [36], pages 114–127.