Efficient and Adaptively Secure Constructions of Identity-Based Cyrptographic Primitives

a dissertation submitted in partial fulfillment
of the requirements for the award of the degree of
Doctor of Philosophy in Computer Science

by

Somindu Chaya Ramanna

under the supervision of

Prof. Palash Sarkar

Applied Statistics Unit
Indian Statistical Institute
203, Barrackpore Trunk Road
Kolkata - 700 108

To Appa and Amma.

Acknowledgements

As an intense 5-year journey towards a PhD degree is reaching completion, I would like to thank people (non-exhaustively) without whose support, this would have remained just a dream. My advisor, Prof. Palash Sarkar has been a source of inspiration and optimism all through. For all the invaluable academic/non-academic discussions, for shaping me as a researcher, for making my stubborn mind learn to deal with problems and failures, for believing in my abilities more than I ever did, I will forever be grateful.

I am thankful to Dr. Sanjit Chatterjee with whom I had many interesting discussions and a fruitful collaboration. He arranged my visits to the Indian Institute of Science, Bangalore, where I spent an important and memorable portion of my PhD life. I want to thank Prof. Bimal Roy who sparked my interest in cryptography back in my undergraduate days and also for his support and encouragement during my stay at ISI. I thank Dr. Kishan Chand Gupta for all the encouragement and thought-provoking discussions. My interactions with Prof. Rana Barua have been very helpful and inspiring. I extend my gratitude to all the other (former) members of the Cryptography group at ISI – Dr. Mridul Nandi, Dr. Sushmita Ruj, Sanjay Bhattacherjee, Subhabrata Samajder, Shashank Singh, Dr. Rishiraj Bhattacharyya, Dr. Sourav Sen Gupta, Tapas Pandit, Dr. Ashish Choudhury, Dr. Arpita Patra, Subhadeep Banik and many others for useful discussions and friendly advice. I would like to thank the faculty and the office staff at Applied Statistics Unit and other units/offices at ISI for their support. I am grateful to all the wonderful teachers at ISI who have played an important role in my intellectual development. I thank ISI for providing travel support to attend several conferences and workshops.

I have to certainly acknowledge my friends at ISI and outside of it with whom I have spent many enjoyable non-working hours. I thank many research scholars and students of ISI for the interesting discussions over coffee/tea and interactions at the hostel. I do not mention their names as I'm afraid the list might be incomplete. Special thanks to Ashish-da and Shashank for all the feasts we had together. Thanks to Sutripta, Rituparna, Krithika, Qaiser, Aparajita and others for the wonderful time we spent in and out of the hostel. It gives me pleasure to thank Sreevani, Sanjay, Subhabrata (and others) for their endless support and the memorable moments we spent together ever since the M.Tech. days. They lifted my spirits whenever I felt down and hopeless and helped me in times of need. I would have probably chosen to stay outside the campus if not for Debasmita who was my neighbour, friend, philosopher and guide for the last six years. I thank her for her infinite patience and kindness and also for all the restaurant hunting tours. I thank her family for their support during my stay at Kolkata. I thank my old friends Prajna, Shruti, Chaitra, Bala and many others for their support, all the trips, holidays and amusement. I am indebted to Smitha for her unending supply of encouragement and love.

Next, I express my gratitude to my family for their unconditional love and support. They have always helped me (re)gain the energy and drive necessary to bring this pursuit to an end. I thank my parents who instilled a love of knowledge in me, taught me to be independent and always encouraged me to go after my dreams. I am grateful to Vishu and Veena for their ceaseless support and encouragement. Thanks to Vicky, Vinutha, Harsha, Nikitha and Dhriti for the memorable times we had together. I thank my parents-in-law for their support.

Finally, I thank my husband, Shubhabrata for his perpetual love and friendship, and for being supportive at all times, be it good or bad.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Public-key encryption (PKE) is an elegant solution to the fundamental problem of cryptography – secure communication between two parties over a public (insecure) channel. The basic idea is that every user A holds two keys – a public key ($\mathcal{PK}_A$) and a secret key ($\mathcal{SK}_A$). A user who wants to send a message to A encrypts it with $\mathcal{PK}_A$ and sends the resulting ciphertext along the public channel. A, at the other end of the channel, decrypts the ciphertext using $\mathcal{SK}_A$ and obtains the message. $\mathcal{SK}_A$ is known to user A alone. Any third party intercepting data on the public channel cannot gain much information from the ciphertext provided it is computationally infeasible to retrieve $\mathcal{SK}_A$ given $\mathcal{PK}_A$. Public keys corresponding to different users are stored in a publicly accessibly directory. A lot of public keys could be floating around in the directory. A central trusted authority produces certificates on these public keys to indicate which of these are genuine. A certificate essentially binds a key to a particular user. In a practical implementation of PKE, there should be efficient mechanisms to manage the large number of certificates present in the directory. Since the inception of the PKE notion in 1976, a huge body of work has emerged centered around the problem of constructing highly efficient and provably secure PKE systems. But the dominant and daunting task of certificate management had been the primary obstacle to a widespread deployment of PKE for quite some time.

Identity-Based Encryption. The notion of identity-based encryption (IBE) was introduced by Shamir [145] in 1984. In an IBE system (depicted in Figure 1.1), the identity id_A of a user A (for instance, A's email address) itself is her public key. A does not generate her own secret key. Instead, a trusted centre called the private key generator (PKG) is responsible for creating and distributing secret keys corresponding to identities. The need for certification does not arise simply because the secret keys are communicated to the intended users over a secure *authenticated* channel. Another user B wanting to send a message secretly to A encrypts the message using id_A. Encryption is done using the public parameters of the PKG $\mathcal{PP}$ available for download to any user. The ciphertext thus obtained is sent to A.

Shamir [145] challenged cryptographers to design a practical IBE system. The search for such a system ended nearly 2 decades later with the ideas presented in three different works. Sakai, Ohgishi

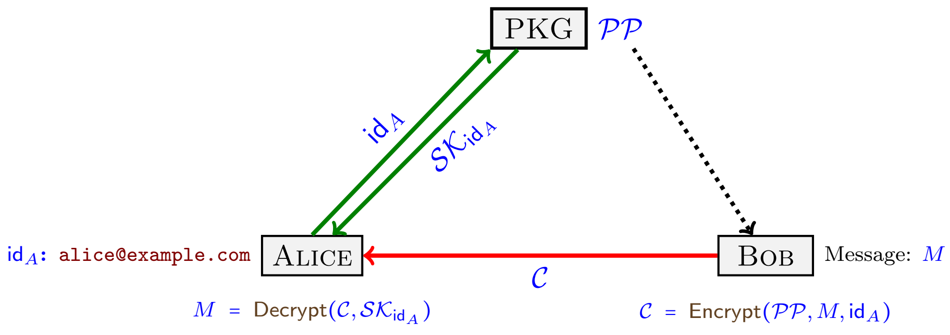

PKG
$\mathcal{PP}$
id_A
$\mathcal{SK}_{\mathsf{id}_A}$
id_A: alice@example.com
Alice
Bob
Message: M
$\mathcal{C}$
$M = \mathsf{Decrypt}(\mathcal{C}, \mathcal{SK}_{\mathsf{id}_A})$
$\mathcal{C} = \mathsf{Encrypt}(\mathcal{PP}, M, \mathsf{id}_A)$

Figure 1.1: Identity-Based Encryption.
Green line: secure authenticated channel; red line: insecure/public channel; dotted line: download

and Kasahara [142] presented an efficient scheme based on *pairings over elliptic curves* but without a formal security model or proof. Boneh and Franklin [25] first formalised the notion of IBE and defined an appropriate security model. They presented an efficient scheme based on pairings and further proved its security using *random oracles*. A parallel work by Cocks [59] proposed an IBE construction based on quadratic residuosity assumption. The three works and in particular that of Boneh and Franklin marked the beginning of a journey aimed at constructing IBE schemes that are both efficient and provably secure. The Boneh-Franklin IBE was constructed using bilinear pairings over elliptic curve groups. Pairings were first used in cryptographic constructions by Joux [99] for realising efficient single-round three-party group key agreement protocols. Following the seminal work of Boneh and Franklin, pairings turned out to be important tools for efficient constructions of a wide range of cryptographic primitives. Currently, most practical constructions of IBE and related primitives rely on pairings.

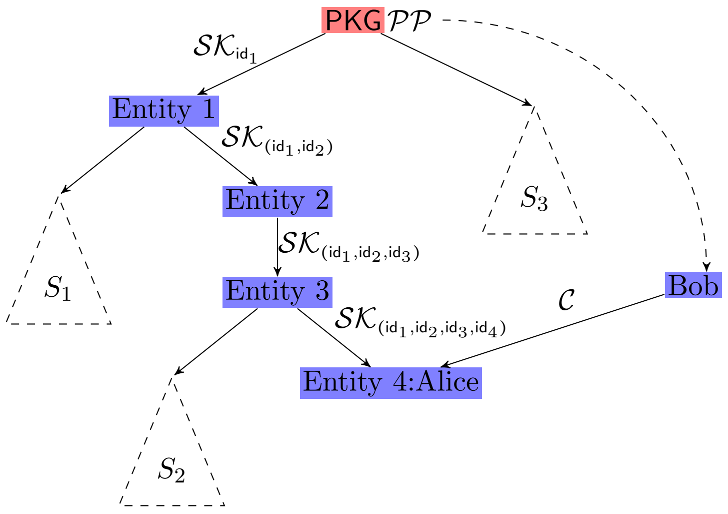

PKG
$\mathcal{PP}$
$\mathcal{SK}_{\mathsf{id}_1}$
Entity 1
$\mathcal{SK}_{(\mathsf{id}_1,\mathsf{id}_2)}$
Entity 2
S_3
S_1
$\mathcal{SK}_{(\mathsf{id}_1,\mathsf{id}_2,\mathsf{id}_3)}$
Entity 3
$\mathcal{C}$
Bob
$\mathcal{SK}_{(\mathsf{id}_1,\mathsf{id}_2,\mathsf{id}_3,\mathsf{id}_4)}$
Entity 4:Alice
S_2

Figure 1.2: Hierarchical identity-based encryption.

Hierarchical Identity-Based Encryption. In IBE, the PKG generates keys for all users of the system. Additionally, keys must be distributed through secure authenticated channels to the corresponding users. As a result there is significant computational overhead on the PKG. An elegant solution to this problem was proposed in [96, 86] which introduced the concept of hierarchical IBE (HIBE) (Figure 1.2). A HIBE system organises the users in a tree structure with PKG at the root and facilitates *delegation* of key generation. Any entity A, using its secret key $\mathcal{SK}_A$, can generate keys for all entities present in the subtree rooted at A. $\mathcal{SK}_A$ can be obtained either from the PKG or from any higher level entity. Entities in a HIBE system are associated to vectors of identities. Several attempts have been made at constructing HIBE systems where certain measures related to efficiency and/or security are independent of maximum depth of the hierarchy.

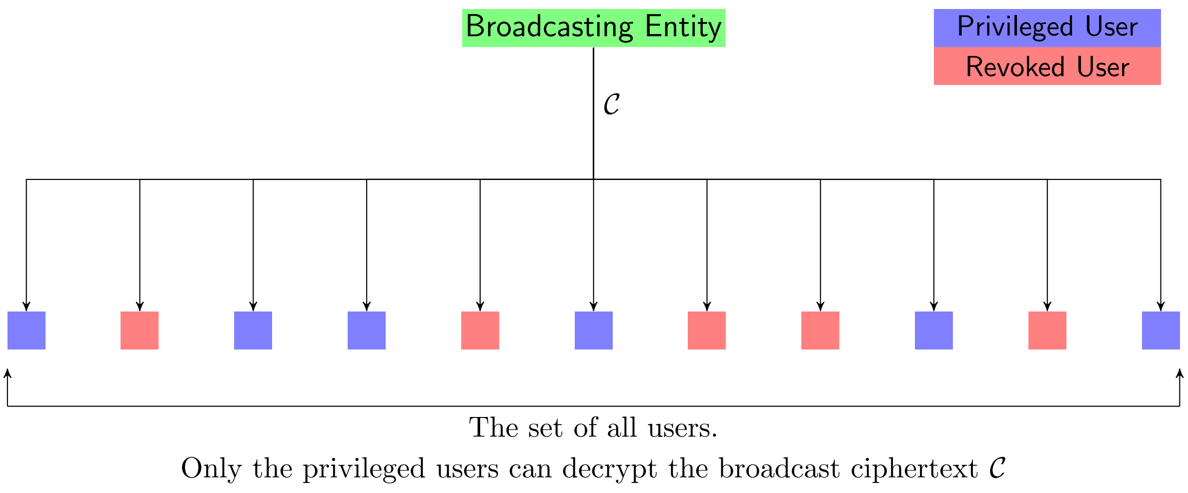


Figure 1.3: Broadcast encryption.

Broadcast Encryption. Broadcast encryption (BE) enables broadcasting encrypted data to a set of users so that only a subset of these users, called *privileged users*, are able to decrypt (see Figure 1.3). Users who are unable to decrypt the broadcasted information are called *revoked* users. The sets of privileged and revoked users form a partition of the set of all users and these sets can vary with each broadcast. BE has a wide range of applications including pay-TV, copyright protection of digital content and encrypted file systems. In public key BE (PKBE), users have public and private keys and anybody can broadcast an encrypted message which can be decrypted only by the set of privileged recipients. Clearly, encryption must be done using the public keys of the privileged users in a way that only they can decrypt the message.

Following the introduction of IBE, the notion of broadcast encryption was soon extended to the identity-based setting. Identity-based broadcast encryption (IBBE) can be seen as an extension of PKBE. As in IBE, there is a PKG which issues decryption keys. A message can be encrypted to a set of privileged identities. The motivation of IBBE is to reduce the communication overhead when the same message is to be sent to a group of identities. Initially studied as multi-receiver identity-based encryption (MR-IBE) in [11] and independently in [10], IBBE has received a lot of attention from the cryptography community.

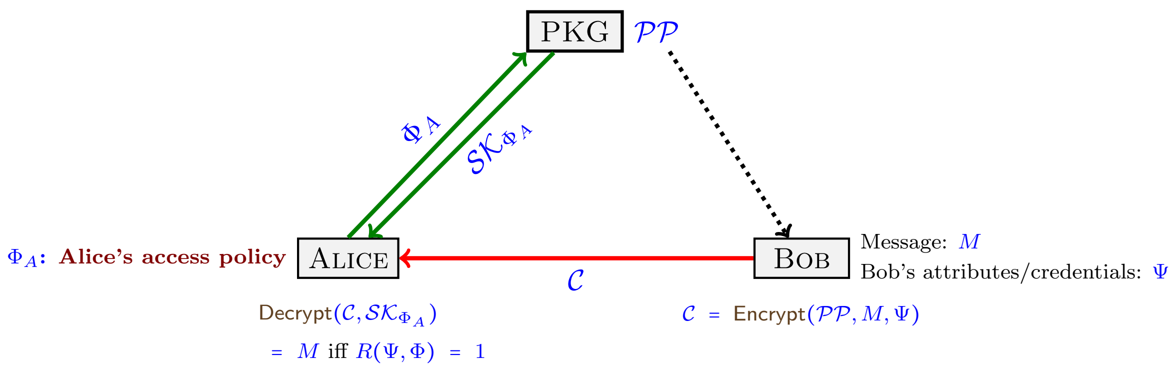

PKG
$\mathcal{PP}$
Φ_A
$\mathcal{SK}_{\Phi_A}$
Φ_A: Alice's access policy
Alice
Bob
Message: M
Bob's attributes/credentials: Ψ
$\mathcal{C}$
Decrypt$(\mathcal{C}, \mathcal{SK}_{\Phi_A})$
$= M$ iff $R(\Psi, \Phi) = 1$
$\mathcal{C} =$ Encrypt$(\mathcal{PP}, M, \Psi)$

Figure 1.4: Attribute-Based Encryption.

Attribute-Based Encryption. Attribute-based encryption (ABE) is a sophisticated form of public key encryption that provides access control on secret data based on certain policies. A more general form called functional encryption (FE) also provides the ability to compute functions over encrypted data (formalised in [32]). In attribute-based encryption, a ciphertext encrypts a message M and an associated attribute or index Ψ that describes the user's credentials. In the *public index model*, the quantity Ψ is revealed in the ciphertext. A key encodes a predicate or an access policy Φ under a relation R. Decryption succeeds and outputs M if a relation $R(\Psi, \Phi)$ holds. Similar to IBE, user secret keys are issued by a trusted authority called the private key generator (PKG). The form of ABE described above is called *key-policy* ABE since the policy is encoded in the key. A complementary form called *ciphertext-policy* ABE is also studied where the policy is embedded in the ciphertext and index in the key. Starting from its introduction in [141, 91, 126], ABE schemes supporting different kinds of access policies have been studied using both bilinear maps and lattices.

The main focus of this work is building efficient IBE, HIBE, IBBE and ABE systems from bilinear pairings that are provably secure.

1.2 Issues Related to the Design of Identity-Based Primitives

Before summarising our results, we present a discussion on some issues to keep in mind while designing IBE and related primitives. This will help the reader better understand the central theme of our work. The features that a system is required to have is determined by the application where it is deployed. On the other hand, features such as implementation efficiency is universally important. We broadly categorise these into two types – efficiency and security – and discuss each in detail.

1.2.1 Efficiency Measures

The efficiency of a particular (H)IBE scheme is usually measured in terms of the following measures – execution time of encryption, decryption and key generation algorithms; sizes of public parameters, ciphertext and secret key. Given below are some remarks regarding an IBE scheme.

- Public parameters are created just once and so are the secret keys corresponding to different users.
- The encryption and decryption algorithms are executed more often than other algorithms.
- As ciphertexts are communicated more often, they consume a significant portion of the communication bandwidth.
- Any user who wishes to send an encrypted message must download the public parameters and the number of such users may be "large".

From the above points, one can infer that an important goal of (H)IBE research is to obtain constructions that have shorter public parameters and ciphertexts as well as faster encryption and decryption algorithms. Towards this goal, a first step would be to choose a suitable mathematical structure to work with for the construction. Most practical constructions of identity-based schemes are obtained from structures called *pairings* and the efficiency of a pairing-based scheme depends on the choice of pairing to a considerable extent.

Choice of Pairing. A pairing is a bilinear, non-degenerate and efficiently computable map e from $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$, where $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are groups of the same order. Practical instantiations of such maps are obtained by suitably choosing $\mathbb{G}_1$ and $\mathbb{G}_2$ to be groups of elliptic curve points and $\mathbb{G}_T$ to be a subgroup of the multiplicative group of a finite field. Three kinds of pairings are identified in the literature: Type-1, where $\mathbb{G}_1 = \mathbb{G}_2$; Type-2, where an efficiently computable isomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$ is known; and Type-3, where there are no known efficiently computable isomorphisms from $\mathbb{G}_1$ to $\mathbb{G}_2$ or vice versa. It has been reported in the literature [153, 76, 44], that among the different types of pairings, it is the Type-3 pairings which provide the most compact parameter sizes and the most efficient algorithms. Further, Type-1 pairings are usually defined over low characteristics fields and recent advances [12, 101, 92, 3, 93] in algorithms for discrete log computations over such fields have raised serious question marks about the security of Type-1 pairings [75]. From both efficiency and security considerations, constructions based on Type-3 pairings are desirable. Less efficient alternatives are when $\mathbb{G}_1$ and $\mathbb{G}_2$ are same (called Type-1 pairings) or when the common group order is a composite number (called composite-order pairings).

Constant-Size Ciphertexts. Among efficiency issues that arise specially in the hierarchical setting (HIBE), the most important is the size of the ciphertexts. Recall that in a HIBE scheme, an individual entity can obtain a private key from either the PKG or from a lower-level entity. In the later case, the complete identity of the entity is obtained by appending its individual identity to the identity of the entity from which it obtains the private key. As a result, identities consist of tuples of varying lengths. Encryption of a message is done for a specified identity tuple. In many HIBE schemes, as the length of the identity tuple increases, so does the length of the resulting ciphertext. Consequently, the bandwidth requirement for communicating the ciphertext also increases. For an individual user, this may actually lead to a disincentive for obtaining decryption keys from lower-level entities. The net result may be in opposition to the basic motivation for a HIBE which is the ability to off-load work to lower-level entities. The solution to this issue is to require the ciphertext size to be independent of the length of the identity tuple. Then, irrespective of the length of the identity tuple, the bandwidth required for the ciphertext would be the same. Such a scheme is called a constant-size ciphertext HIBE.

Throughout the thesis, we use the abbreviation CC-HIBE to denote HIBE schemes with constant-size ciphertexts. We clarify that the constant size here only refers to the number of group elements in the ciphertext. The size of representation of the group elements, however, needs to increase if the value of the security parameter increases.

Broadcast Setting. Below we discuss some efficiency measures specific to the design of (IB)BE schemes.

Header size: In all BE schemes, the actual message undergoes a single encryption with a session key. In addition to this, the ciphertext contains some additional information which allows a privileged user to obtain the session key and recover the message. This additional information constitutes the header of the ciphertext. To reduce the communication overhead it is desirable to reduce the size of the header as much as possible. So, BE schemes with lower header sizes are preferable.

User key size: The amount of key material that a user has to store is an important parameter. Practical deployment may require storing such material in smart cards. Consequently, it is of interest to try and reduce the size of user keys as much as possible.

1.2.2 Security

All security proofs are reductionist arguments i.e., "breaking the security" of a system is reduced to the problem of solving some "computationally hard" problem. What it means to break the security of a particular primitive is defined in terms of a security model which more or less captures real world attacks. Once the model is defined, concrete schemes are proved secure based on the hardness of certain problems. Normally these problems are related to the mathematical structure based on which the scheme is constructed. A security reduction for a concrete scheme $\mathcal{T}$ based on a problem Π would proceed in the following way. An algorithm $\mathscr{B}$ asked to solve a particular instance of Π simulates the real world system $\mathcal{T}$ in an attacker's view. The attacker's ability to break the security of the cryptosystem will be used to $\mathscr{B}$'s advantage in solving Π. Essentially $\mathscr{B}$ uses the attacker's ability to break the security of $\mathcal{T}$ in solving Π.

Degradation. Security guarantee for particular scheme is stated as follows. If an algorithm running in time t breaks the security of the scheme with "advantage" ε, then some computational problem Π can be solved in time t' with advantage ε'. The ratio δ of t'/ε' to t/ε is the tightness gap and the reduction is said to have a degradation of δ. Depending upon the scheme and the reduction, δ could be a constant or could depend on quantities such as the security parameter, the maximum number of corrupt users, the maximum length of an identity tuple (in case of HIBE), number of privileged users (in the broadcast setting) and possibly other parameters. Designing schemes that have low degradation is important.

Hardness Assumptions: As mentioned above, the proof of security of a (id-based) primitive is based on the assumption that some well formulated problem is computationally hard. There is a small subset of such problems which are considered to be standard. Apart from standard hardness assumptions, designers sometimes have to create new hard problems to effect a reduction. These problems are often parametrised by a quantity arising either in the construction or the proof. If not, they are termed *static*. Since such non-standard problems are less studied, a basic theme of research is to try and obtain schemes which can be proved secure under standard and/or static assumptions.

Modelling Security. Proving security of an identity-based encryption scheme involves many subtleties. In traditional PKE, each user creates her own set of keys independent of other users. Hence, while modelling security it is enough to consider malicious behaviour on part of just one individual user. But in case of IBE, all secret keys corresponding to individual users are generated from a single master secret key. A possible attack scenario is when a group of users collude to compromise the privacy of another user outside of the group. Also, this group can grow with time. A strong notion of security must consider such collusion attacks. This attack is modelled by providing an adversary access to a key extraction oracle and the freedom to query the oracle with identities chosen adaptively. The adversary chooses the target identity at some point and then continues to extract keys. Naturally, the attacker is not allowed to extract a key for the target identity. This security model is called the *adaptive identity model* (formalised by Boneh and Franklin [25]). A central goal in building identity-based schemes is to ensure security in the adaptive model.

Anonymity. Another important security notion is *anonymity* which requires that a ciphertext does not reveal any information about the recipient's identity. Abdalla et al. [1] first formalised the notion of anonymity and used anonymous IBE to construct public key encryption with keyword search (PEKS). PEKS enables search on encrypted documents based on some keywords and this capability for search is delegateable. Anonymous HIBE schemes provide means to extend PEKS to more sophisticated primitives such as public key encryption with temporary keyword search (PETKS) and identity-based encryption with keyword search (IBEKS). Obtaining anonymous HIBE schemes is one of the main objectives of this thesis.

Proving Security. While proving security of a system within the adaptive identity model, the simulator should be designed in a way that it can answer the adaptive key extraction queries and simultaneously challenge that adversary to break the security of the target identity.

Initial constructions for IBE and related primitives were proved secure in the *random oracle model* (ROM). The ROM is an idealised model in which every hash function is replaced by a truly random function. In attempts to prove security without random oracles, researchers considered weaker security goals modeled as *selective* security. In this model, the attacker chooses the target identity before looking at the public parameters. Although unrealistic, the selective model was an important step towards proofs without random oracles. A proof technique used in earlier works partitions the identity space into two sets – one for which the simulator can create secret keys and the other containing possible target identities. This is known as the *partitioning* approach.

The partition is either hard coded in the public parameters during setup or programmed into the random oracle. The former method is particularly easy in the selective setting – the target identity (known apriori) is the only identity for which the simulator cannot create a secret key.

Adaptively secure IBE without random oracles was first obtained by Boneh and Boyen [23] and later on by Waters [157]. Both proofs rely on the partitioning approach. The simulator actually guesses the partition and aborts if the target identity does not fall in the corresponding set. When extended to more sophisticated primitives such as HIBE or IBBE this approach introduces an exponential factor in security degradation.

After a long gap, in 2009, Waters [158] introduced a new proof technique called dual system encryption. Using the new technique, Waters presented the first IBE scheme with constant-size public parameters achieving adaptive security under standard assumptions. This IBE was further extended to adaptively secure HIBE and BE schemes. The dual system methodology was indeed a paradigm shift in provable security and has lead to many adaptively secure encryption schemes with more sophisticated structure. Examples include predicate encryption, inner product encryption and attribute-based encryption.

We use dual system technique in all our proofs. Our proofs rely on *static* assumptions (number of elements in the instance is independent of the parameters pertaining to the construction). Furthermore, an exponential loss in security is avoided in case of primitives with richer structure such as HIBE, IBBE and ABE.

1.2.3 Prefix Decryption

We discuss an important feature of HIBE schemes that cannot be really classified as an efficiency or security property. In some HIBE schemes, a ciphertext for an identity vector can be decrypted by any entity possessing a secret key for a prefix of that identity. Let us name this property *prefix decryption*. In constructions with separate ciphertext elements corresponding to individual components of the identity tuple, such as the one in [86], prefix decryption is facilitated – the ciphertext can be truncated to obtain a valid ciphertext under the prefix identity vector and thus can be decrypted using the corresponding key. Prefix decryption in a CC-HIBE could be done as follows: the key for the prefix is used to create a key corresponding to the recipient identity via delegation; the resulting key is used to decrypt the ciphertext. The latter method fails when the scheme is anonymous i.e., when the recipient identity is hidden. Delegation can no longer be done without knowledge of the recipient identity. The above discussion suggests that achieving constant-size ciphertexts and anonymity simultaneously results in the loss of prefix decryption. Although it may seem that this restriction is somehow tied to the property of anonymity, we would like to emphasise that this is a definitional issue. That is, whether prefix decryption is allowed or not must reflect in the HIBE definition. The definition we provide does not explicitly allow prefix decryption. We stress that the prefix decryption property is absent in all known HIBE constructions that concurrently attain constant-size ciphertexts and anonymity. Furthermore, all known HIBE schemes possess at most two out of the three features – constant-size ciphertexts, anonymity and prefix decryption.

On the other hand, not having prefix decryption guards against key escrow to some extent. An entity has the power to delegate keys to lower level entities but cannot decrypt ciphertexts sent to the lower-level entities. This feature may be useful in applications such as email. If the higher level entities are key generating servers, user privacy will not be compromised in the event that any of these servers is corrupted. Further, in primitives such as identity-based searchable encryption obtained from anonymous HIBE schemes [1], this limitation does not make any difference.

We provide a more detailed discussion on prefix decryption in Section 6.6.

1.3 Thesis Organisation

This thesis is a collection of results from five papers [136, 137, 139, 138, 135] organised into nine chapters. Chapter 1 is an introduction to identity-based cryptographic primitives. Chapter 2 contains notation, definitions and mathematical preliminaries required for later chapters. Chapter 3 discusses the evolution of techniques for proving security of identity-based systems. Chapter 4 presents a survey of previous and related works.

In Chapter 5, we have systematically simplified Waters 2009 IBE scheme in the setting of asymmetric pairing. Waters' dual system IBE scheme [158] was described in the setting of symmetric pairing. A key feature of the construction is the presence of random tags in the ciphertext and decryption key. The simplifications retain tags used in the original description. This leads to several variants, the first one of which is based on standard assumptions and in comparison to Waters original scheme reduces ciphertexts and keys by two elements each. Going through several stages of simplifications, we finally obtain a simple scheme whose security can be based on two standard assumptions and a natural and minimal extension of the decision Diffie-Hellman problem for asymmetric pairing groups. The scheme itself is also minimal in the sense that apart from the tags, both encryption and key generation use exactly one randomiser each. This final scheme is more efficient than both the previous dual system IBE scheme in the asymmetric setting due to Lewko and Waters and the more recent dual system IBE scheme due to Lewko. We extend the IBE scheme to hierarchical IBE (HIBE) and broadcast encryption (BE) schemes. Both primitives are secure in their respective full models and have better efficiencies compared to previously known schemes offering the same level and type of security.

In Chapter 6, we consider the problem of constructing constant-size ciphertext HIBE (CC-HIBE) schemes based on asymmetric pairings within the dual system framework. We present a HIBE scheme denoted $\mathcal{LW}$-$\mathcal{AHIBE}$, with constant-size ciphertexts that can be implemented using the most efficient bilinear pairings, namely, Type-3 pairings. In addition to being fully secure, this scheme is anonymous. $\mathcal{LW}$-$\mathcal{AHIBE}$ is obtained by extending the asymmetric pairing based IBE scheme due to Lewko and Waters [114]. The extension uses the approach of Boneh-Boyen-Goh [24] to obtain constant-size ciphertexts. Anonymity comes as a by-product of the dual system proof. In fact, a technique introduced by Boyen-Waters [36] to achieve anonymity is used implicitly. At the time it was proposed, $\mathcal{LW}$-$\mathcal{AHIBE}$ was the only known scheme using Type-3 pairings to achieve constant-size ciphertexts, security against adaptive-identity attacks and anonymity under static assumptions without random oracles.

We then present two more HIBE schemes, denoted as $\mathcal{JR}$-$\mathcal{AHIBE}$ and $\mathcal{JR}$-$\mathcal{HIBE}$, from Type-3 pairings, also with constant sized ciphertexts. Scheme $\mathcal{JR}$-$\mathcal{AHIBE}$ achieves anonymity while $\mathcal{JR}$-$\mathcal{HIBE}$ is non-anonymous. The constructions are obtained by extending the IBE scheme recently proposed by Jutla and Roy [103]. Security is based on the standard decisional Symmetric eXternal

Chapter	Schemes	Category
4	$\mathcal{IBE}1$, $\mathcal{IBE}2$, $\mathcal{IBE}3$, $\mathcal{IBE}4$, $\mathcal{IBE}5$, $\mathcal{IBE}6$	IBE
	$\mathcal{HIBE}6$	HIBE
	$\mathcal{BE}6$	BE
5	$\mathcal{JR}$-$\mathcal{HIBE}$	Non-anonymous CC-HIBE
	$\mathcal{LW}$-$\mathcal{AHIBE}$, $\mathcal{JR}$-$\mathcal{AHIBE}$	Anonymous CC-HIBE
6	$\mathcal{IBBE}_1$, $\mathcal{IBBE}_2$, $\mathcal{IBBE}_1^{\mathrm{ROM}}$, $\mathcal{IBBE}_2^{\mathrm{ROM}}$	IBBE
7	$\mathcal{B}$-$\mathcal{ABE}$, $\mathcal{F}$-$\mathcal{ABE}$	Bounded DFA-Based ABE

Table 1.1: Summary of schemes proposed.

Diffie-Hellman (SXDH) assumption. In terms of provable security properties, previous direct constructions of constant-size ciphertext HIBE had one or more of the following drawbacks: security in the weaker model of selective-identity attacks; exponential security degradation in the depth of the HIBE; and use of non-standard assumptions. The security arguments for $\mathcal{JR}$-$\mathcal{AHIBE}$ and $\mathcal{JR}$-$\mathcal{HIBE}$ avoid all of these drawbacks. These drawbacks can also be avoided by obtaining HIBE schemes by specialising schemes for hierarchical inner product encryption; the downside is that the resulting efficiencies are inferior to those of the schemes reported here. Currently, there is no known anonymous HIBE scheme having the security properties of $\mathcal{JR}$-$\mathcal{AHIBE}$ and comparable efficiency. Based on the current state-of-the-art, $\mathcal{JR}$-$\mathcal{AHIBE}$ and $\mathcal{JR}$-$\mathcal{HIBE}$ are the schemes of choice for efficient implementation of (anonymous) HIBE constructions.

Chapter 7 describes the first constructions of efficient identity-based broadcast encryption (IBBE) schemes which can be proved secure against adaptive-identity attacks based on standard assumptions. The constructions are obtained by extending the currently known most efficient IBE scheme proposed by Jutla and Roy [103]. Ciphertext size and user storage compare favourably to previously known constructions. The new constructions fill both a practical and a theoretical gap in the literature on efficient IBBE schemes.

In Chapter 8, we consider ABE constructions form bilinear maps. Most of the known bilinear-map-based ABE schemes have one property in common – the functions only deal with fixed-size inputs. Waters [160] went beyond fixed-size inputs and proposed an ABE scheme that operates over arbitrary-sized inputs. In this system, a secret key is associated with a deterministic finite automaton (DFA) $\mathcal{M}$ and the index Ψ is a string w over the input alphabet of the DFA. Decryption succeeds if $\mathcal{M}$ accepts w. As a result, the system supports the class of regular languages. This construction was shown to be selectively secure without random oracles based on the eXpanded Decisional Bilinear Diffie-Hellman Exponent (XDBDHE) assumption parametrised by ℓ, the length of the challenge string. Over arbitrary sized inputs, there are no known schemes that achieve adaptive security from static assumptions.

We present an adaptively secure DFA-based ABE scheme. The construction uses composite-order bilinear pairings and is built upon the selectively secure DFA-based FE scheme of Waters (Crypto 2012). The scheme is proven secure using the dual system methodology under static assumptions. A dual system proof requires generating semi-functional components appropriately during simulation. In addition, these components must be shown to be properly distributed in an attacker's view. This can be ensured by imposing a restriction on the automata and strings over which the scheme is built i.e., every symbol can appear at most once in a string and in the set of transition tuples of an automata. First a basic construction with the aforementioned constraint is obtained and proved to be adaptively secure. With the restrictions, our system supports a sub-class of regular languages. We then show how to extend this basic scheme to a full scheme where the restrictions can be relaxed by placing a bound on the number of occurrences of any symbol in a string and in the set of transitions. With the relaxed restrictions, our system supports functionality defined by a larger subset of regular languages.

Finally, Chapter 9 provides a conclusion along with a discussion on open problems and directions for future work. Table 1.1 provides a tabular summary of the different schemes proposed in this work. Most of our constructions, at the time they were proposed, advanced the state-of-the-art in terms of efficiency while providing strong security guarantees. Some of them are to this date among the most efficient schemes known.

Chapter 2

Definitions and Mathematical Preliminaries

In this chapter, we present notation, definitions and a discussion on pairings. First, we define HIBE, PKBE, IBBE and FE schemes followed by the corresponding security models. We provide definitions for both IBE and HIBE but discuss security only for HIBE. The security of IBE can be seen as a special case of HIBE with the number of levels set to one. In case of HIBE schemes, there are different formalisations of security. We discuss them in detail.

2.1 Definitions of Primitives

Before proceeding, we fix some notation. $x_1, \ldots, x_k \xleftarrow{\mathrm{R}} \mathcal{X}$ indicates that elements $x_1, \ldots, x_k$ are sampled independently from the set $\mathcal{X}$ according to some distribution R. We use U to denote the uniform distribution. For a (probabilistic) algorithm $\mathcal{A}$, $y \xleftarrow{\mathrm{R}} \mathcal{A}(x)$ means that y is chosen according to the output distribution of $\mathcal{A}$ on input x. $\mathcal{A}(x; r)$ denotes that $\mathcal{A}$ is run on input x with its internal random coins set to r. For two integers $a < b$, the notation $[a, b]$ represents the set $\{x \in \mathbb{Z} : a \leq x \leq b\}$.

2.1.1 Identity-Based Encryption

The notion of IBE was proposed by Shamir in [145] and later formalised by Boneh and Franklin [25].

Definition 2.1.1 (IBE)**.** An IBE scheme consists of four probabilistic algorithms – $\mathcal{IBE}$.Setup, $\mathcal{IBE}$.KeyGen, $\mathcal{IBE}$.Encrypt and $\mathcal{IBE}$.Decrypt – each of which has runs in time upper bounded by a polynomial in the security parameter, denoted κ. The first two algorithms are run by the PKG, $\mathcal{IBE}$.Encrypt by the sender and $\mathcal{IBE}$.Decrypt by the receiver. Formally, the security parameter is input to the $\mathcal{IBE}$.Setup algorithm as 1^κ indicating that the time complexities of the algorithms should be polynomially bounded in the length of the input, which is *kappa*. For the sake of simplicity, we write κ in place of 1^κ.

$\mathcal{IBE}$.Setup(κ): takes as input the security parameter κ and outputs the public parameters $\mathcal{PP}$ along with the master secret key $\mathcal{MSK}$. $\mathcal{PP}$ is made publicly available. $\mathcal{MSK}$ is kept secret and known only to the PKG. The public parameters $\mathcal{PP}$ also include descriptions of the identity space $\mathscr{I}$, the message space $\mathscr{M}$ and the ciphertext space $\mathscr{C}$.

$\mathcal{IBE}$.KeyGen($\mathcal{MSK}$, id): takes as input an identity id and $\mathcal{MSK}$ and returns a secret key $\mathcal{SK}_{\mathsf{id}}$ corresponding to id. The communication of id and $\mathcal{SK}_{\mathsf{id}}$ happens through a secure authenticated channel between the PKG and the user identified by id.

$\mathcal{IBE}$.Encrypt($\mathcal{PP}$, id, M): takes as input a message M, identity of the intended recipient id, the public parameters $\mathcal{PP}$ and produces as output a ciphertext $\mathcal{C}$.

$\mathcal{IBE}$.Decrypt($\mathcal{C}$, id, $\mathcal{SK}_{\mathsf{id}}$): takes as input a ciphertext $\mathcal{C}$, an identity id, the corresponding secret key $\mathcal{SK}_{\mathsf{id}}$, the public parameters $\mathcal{PP}$ and returns either the corresponding message M or returns $\perp$ indicating failure.

$\mathcal{IBE}$.Setup is a probabilistic algorithm that picks a random pair of public parameters and master secret. The $\mathcal{IBE}$.Encrypt algorithm is randomised. It essentially samples a ciphertext from the set of possible ciphertexts corresponding to a given identity and message. Any ciphertext that can be produced by the $\mathcal{IBE}$.Encrypt algorithm is said to be *valid.* The $\mathcal{IBE}$.KeyGen algorithm, on the other hand, could be either deterministic or probabilistic. In the latter case, the algorithm can be seen as choosing a key at random from the set of all possible keys for a given identity. The algorithm $\mathcal{IBE}$.Decrypt, by definition, can be probabilistic. While most IBE constructions have deterministic decryption, there are a few known constructions in which decryption can fail with some probability. Typically all sampling is done according to the uniform distribution over the respective domains. However, the definition of IBE does not require this.

Technically, the public parameters $\mathcal{PP}$ may be required for $\mathcal{IBE}$.KeyGen and $\mathcal{IBE}$.Decrypt algorithms depending on the construction. In the sequel, we do not explicitly mention this and whether $\mathcal{IBE}.\mathcal{PP}$ must be provided as input or not will be clear from the context.

We now formally state the standard correctness criterion for IBE schemes.

Correctness. An IBE scheme consisting of the above four algorithms is said to satisfy the correctness condition if for all $(\mathcal{PP}, \mathcal{MSK}) \xleftarrow{\text{R}} \mathcal{IBE}.\mathsf{Setup}(\kappa)$, for all $\mathcal{SK}_{\mathsf{id}} \xleftarrow{\text{R}} \mathcal{IBE}.\mathsf{KeyGen}$ and for all $\mathcal{C} \xleftarrow{\text{R}} \mathcal{IBE}.\mathsf{Encrypt}(\mathsf{id}, M)$, with probability 1, it holds that $M = \mathcal{IBE}.\mathsf{Decrypt}(\mathcal{C}, \mathsf{id}, \mathcal{SK}_{\mathsf{id}})$.

2.1.2 Hierarchical Identity-Based Encryption

As mentioned in Chapter 1, HIBE is an extension of IBE where users/entities are organised in a tree-like structure with the PKG at the root. The basic motivation for imposing such a structure is as follows. In the IBE setting, the PKG has to generate keys for every user. In addition, each key needs to be transmitted to the corresponding user through a secure channel. Clearly, when the number of users is large, this will create a huge computation and communication overhead on the PKG. Suppose the entities are organised into a rooted tree and higher level entities have the ability to delegate key generation to lower level entities without the involvement of the PKG. This reduces the load on the PKG significantly.

In a HIBE system, each user is associated with a vector of identities. Let $\mathcal{HIBE}$ denote a HIBE scheme with a hierarchy of maximum depth h. Let $\mathscr{I}$ denote the identity space of $\mathcal{IBE}$. Then each user in $\mathcal{HIBE}$ will be associated with a vector $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ where $1 \le \ell \le h$ and $\mathsf{id}_j \in \mathscr{I}$ for each $j \in [1, \ell]$. A vector $\mathbf{id}' = (\mathsf{id}_1, \ldots, \mathsf{id}'_\ell)$ with $\ell' < \ell$ is called a prefix of $\mathbf{id}$. The PKG can generate secret keys for any identity vector. The public parameters remain the same for all entities and fixed by the PKG during setup. Individual entities are not allowed to have their own parameters. Moreover, the master secret is known only to the PKG. So the question arises as to how delegation of key generation is enabled. A secret key for $\mathbf{id}$ can be generated either by an entity corresponding to a prefix (say, $\mathbf{id}'$) of $\mathbf{id}$ or obtained directly from the PKG. In the former case, any secret key $\mathcal{SK}_{\mathbf{id}'}$ corresponding to $\mathbf{id}'$ contains sufficient information required to generate a key for $\mathbf{id}$. In fact, $\mathcal{SK}_{\mathbf{id}'}$ enables key generation for any $\mathbf{id}$ containing $\mathbf{id}'$ as a prefix.

Definition 2.1.2 (HIBE)**.** We now proceed to the formal definition. A HIBE scheme $\mathcal{HIBE}$ consists of five probabilistic algorithms – $\mathcal{HIBE}.\mathsf{Setup}$, $\mathcal{HIBE}.\mathsf{Encrypt}$, $\mathcal{HIBE}.\mathsf{KeyGen}$, $\mathcal{HIBE}.\mathsf{Delegate}$ and $\mathcal{HIBE}.\mathsf{Decrypt}$.

$\mathcal{HIBE}.\mathsf{Setup}(\kappa)$**:** based on an input security parameter κ, generates and outputs the public parameters $\mathcal{PP}$ and the master secret $\mathcal{MSK}$.

$\mathcal{HIBE}.\mathsf{KeyGen}(\mathcal{MSK}, \mathsf{id})$**:** inputs an identity vector $\mathbf{id}$ and master secret $\mathcal{MSK}$ and outputs the secret key $\mathcal{SK}_{\mathbf{id}}$ corresponding to $\mathbf{id}$.

$\mathcal{HIBE}.\mathsf{Encrypt}(\mathcal{PP}, \mathbf{id}, M)$**:** inputs an identity $\mathbf{id}$, a message M and returns a ciphertext $\mathcal{C}$.

$\mathcal{HIBE}.\mathsf{Delegate}(\mathcal{PP}, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathsf{id})$**:** takes as input $\mathcal{PP}$, a depth ℓ identity vector $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, a secret key $\mathcal{SK}_{\mathbf{id}}$ and an identity id; returns a secret key for the identity vector $(\mathsf{id}_1, \ldots, \mathsf{id}_{\ell+1})$ where $\mathsf{id}_{\ell+1} = \mathsf{id}$.

$\mathcal{HIBE}.\mathsf{Decrypt}(\mathcal{C}, \mathbf{id}, \mathcal{SK}_{\mathbf{id}})$**:** inputs a ciphertext $\mathcal{C}$, an identity vector $\mathbf{id}$, secret key $\mathcal{SK}_{\mathbf{id}}$ and returns either the corresponding message M or $\perp$ indicating failure.

As in IBE, all the above algorithms run in time polynomial in the κ. Furthermore, the other comments regarding the algorithms for IBE hold even in the hierarchical case.

Correctness. The HIBE scheme is said to satisfy the correctness condition if for all $(\mathcal{PP}, \mathcal{MSK}) \xleftarrow{\text{R}} \mathcal{HIBE}.\mathsf{Setup}(\kappa)$, for all $\mathcal{SK}_{\mathbf{id}}$ generated by a sequence of calls to $\mathcal{HIBE}.\mathsf{KeyGen}$ and $\mathcal{HIBE}.\mathsf{Delegate}$ algorithms and for all $\mathcal{C} \xleftarrow{\text{R}} \mathcal{HIBE}.\mathsf{Encrypt}(\mathcal{PP}, \mathbf{id}, M)$, with probability 1, it holds that $M = \mathcal{HIBE}.\mathsf{Decrypt}(\mathcal{C}, \mathbf{id}, \mathcal{SK}_{\mathbf{id}})$.

2.1.3 (Public Key) Broadcast Encryption

A broadcast encryption scheme allows an entity to send an encrypted message to a set $\mathcal{N}$ of users such that only a subset $\mathcal{S} \subseteq \mathcal{N}$ can decrypt. The ciphertext is, however, broadcasted to all the users. We call S the set of privileged users and $\mathcal{R} = \mathcal{N} \setminus \mathcal{S}$ as the set of revoked users. In a public key broadcast encryption (PKBE) scheme, anyone is allowed to encrypt. Traditionally, each user has a pair of keys – public and private. A sender needs the public keys of all users in order to broadcast a ciphertext. In addition these public keys must be certified. We consider a variant where, a set of common public parameters is associated to a PKG (as in IBE) and keys for individual users are generated by the PKG, thus removing the need for certification. In this setting, encryption is done using the public parameters. In addition, we are interested in a *static* system (the composition of $\mathcal{N}$ does not change with time) with stateless receivers (users are not required to update their private keys). Note that the number of users $n = |\mathcal{N}|$ is fixed in the static setting. We require n to be polynomially bounded in the security parameter κ. We write broadcast encryption and omit the prefix "public-key" for simplicity.

For the definition of BE, we refer to [87]. Prior to this work, 3–algorithm definition was used, variants of which were introduced in [69, 33]). In case of dynamic BE [69, 65] another algorithm called Join or Register would be included to enable new users to join the system at any point of time.

Definition 2.1.3 (PKBE). A broadcast encryption scheme $\mathcal{BE}$ is defined by four probabilistic algorithms – $\mathcal{BE}.\mathsf{Setup}$, $\mathcal{BE}.\mathsf{KeyGen}$, $\mathcal{BE}.\mathsf{Encrypt}$, $\mathcal{BE}.\mathsf{Decrypt}$ whose run times are bounded above by a polynomial in the security parameter κ. The first three are executed by the broadcasting entity and the last one by a recipient of the broadcast message.

$\mathcal{BE}.\mathsf{Setup}(\kappa, n)$**:** takes as input the security parameter κ and the number of users $n = \mathsf{poly}(\kappa)$. Let $\mathcal{N} = [1, n]$. It outputs a public/secret key pair $(\mathcal{PK}, \mathcal{SK})$.

$\mathcal{BE}.\mathsf{KeyGen}(\mathcal{SK}, j)$**:** takes as input $\mathcal{SK}$, a user index $j \in \mathcal{N}$ and returns a private key $\mathcal{SK}_j$ for user j.

$\mathcal{BE}.\mathsf{Encrypt}(\mathcal{PK}, \mathcal{S}, M)$**:** inputs a subset $\mathcal{S} \subseteq \mathcal{N}$ of users to whom the message is to be broadcast, the public key $\mathcal{PK}$ and the message M. It outputs a ciphertext $\mathcal{C}$.

$\mathcal{BE}.\mathsf{Decrypt}(\mathcal{S}, j, \mathcal{SK}_j, \mathcal{C})$**:** takes as input a ciphertext $\mathcal{C}$, a set $\mathcal{S}$ of users to whom $\mathcal{C}$ is encrypted and the user's secret key $\mathcal{SK}_j$. If $j \in \mathcal{S}$ then the corresponding message M is returned.

Correctness: The BE scheme is said to satisfy the correctness condition if for all $(\mathcal{PK}, \mathcal{SK}) \xleftarrow{\mathrm{R}} \mathcal{BE}.\mathsf{Setup}(\kappa, n)$, for all $\mathcal{S} \subseteq \mathcal{N}$, for all $j \in \mathcal{S}$, for all $\mathcal{SK}_j \xleftarrow{\mathrm{R}} \mathcal{BE}.\mathsf{KeyGen}(\mathcal{SK}, j)$ and for all $\mathcal{C} \xleftarrow{\mathrm{R}} \mathcal{BE}.\mathsf{Encrypt}(\mathcal{PK}, \mathcal{S}, M)$, with probability 1, it holds that $M = \mathcal{BE}.\mathsf{Decrypt}(\mathcal{C}, \mathcal{S}, \mathcal{SK}_j)$ if $j \in \mathcal{S}$.

2.1.4 Identity-Based Broadcast Encryption

An identity-based broadcast encryption (IBBE) is similar to the public key broadcast encryption defined in Section 2.1.3 except that each user is associated with an identity. Additionally, the number of users/identities is allowed to be exponential in the security parameter. However, we consider a variant where the maximum number of users in any recipient set $\mathcal{S}$ is fixed at the time of setup.

Definition 2.1.4 (IBBE). An IBBE scheme $\mathcal{IBBE}$ is defined by four probabilistic algorithms – $\mathcal{IBBE}.\mathsf{Setup}$, $\mathcal{IBBE}.\mathsf{KeyGen}$, $\mathcal{IBBE}.\mathsf{Encrypt}$, $\mathcal{IBBE}.\mathsf{Decrypt}$ whose run times are bounded above by a polynomial in the security parameter κ. The identity space is denoted $\mathscr{I}$.

$\mathcal{IBBE}$.$\mathsf{Setup}(\kappa, m)$: Takes as input a security parameter κ and the maximum number m of identities in a privileged recipient group. It outputs the public parameters $\mathcal{PP}$ and the master secret $\mathcal{MSK}$.

$\mathcal{IBBE}$.$\mathsf{KeyGen}(\mathcal{MSK}, \mathsf{id})$: On input an identity id and the master secret $\mathcal{MSK}$, this algorithm generates and outputs a secret key $\mathcal{SK}_{\mathsf{id}}$ for id.

$\mathcal{IBBE}$.$\mathsf{Encrypt}(\mathcal{PP}, S \subseteq \mathscr{I}, M)$: Takes as input a set of identities S that are the intended recipients of the message M. If $|S| \leq m$, the algorithm outputs a ciphertext $\mathcal{C}$.

$\mathcal{IBBE}$.$\mathsf{Decrypt}(\mathcal{PP}, S, \mathsf{id}, \mathcal{SK}_{\mathsf{id}}, \mathcal{C})$: Inputs the public parameters, a set $S = \{\mathsf{id}_1, \ldots, \mathsf{id}_\ell\}$, an identity id, a secret key $\mathcal{SK}_{\mathsf{id}}$ corresponding to id, a header Hdr and outputs the message M.

Correctness. The IBBE scheme satisfies the correctness condition if for all sets $S \subseteq \mathscr{I}$ with $|S| \leq m$, for all $\mathsf{id}_i \in S$, if $(\mathcal{PP}, \mathcal{MSK}) \xleftarrow{\text{R}} \mathcal{IBBE}.\mathsf{Setup}(\kappa, \mathscr{I}, m)$, $\mathcal{SK}_{\mathsf{id}_i} \xleftarrow{\text{R}} \mathcal{IBBE}.\mathsf{KeyGen}(\mathcal{MSK}, \mathsf{id}_i)$, $\mathcal{C} \xleftarrow{\text{R}} \mathcal{IBBE}.\mathsf{Encrypt}(\mathcal{PP}, S, M)$, the $\Pr[M = \mathcal{IBBE}.\mathsf{Decrypt}(\mathcal{PP}, S, \mathsf{id}_i, \mathcal{SK}_{\mathsf{id}_i}, \mathcal{C})] = 1$.

IBBE is usually defined following the hybrid encryption (KEM-DEM) paradigm. The IBBE key encapsulation mechanism (KEM) produces a session key along with a header. This session key is used to encrypt the message via the data encapsulation mechanism (DEM). The DEM can be instantiated to any secure symmetric key encryption scheme. The security of the IBBE would then rely on the security of the KEM. Our main interest is in designing secure IBBE-KEMs. The details of the symmetric encryption portion (DEM) is omitted and also not considered in our security proof.

Definition 2.1.5 (IBBE-KEM)**.** An IBBE-KEM is defined by four probabilistic algorithms – $\mathcal{IBBE}$-$\mathcal{KEM}$.Setup, $\mathcal{IBBE}$-$\mathcal{KEM}$.Encap, $\mathcal{IBBE}$-$\mathcal{KEM}$.KeyGen and $\mathcal{IBBE}$-$\mathcal{KEM}$.Decap. The identity space is denoted $\mathscr{I}$ and the key space for the symmetric encryption scheme is denoted by $\mathscr{K}$.

$\mathcal{IBBE}$-$\mathcal{KEM}$.$\mathsf{Setup}(\kappa, m)$: Takes as input a security parameter κ and the maximum number m of identities in a privileged recipient group. It outputs the public parameters $\mathcal{PP}$ and the master secret $\mathcal{MSK}$.

$\mathcal{IBBE}$-$\mathcal{KEM}$.$\mathsf{KeyGen}(\mathcal{MSK}, \mathsf{id})$: Input is an identity id and master secret $\mathcal{MSK}$; output is a secret key $\mathcal{SK}_{\mathsf{id}}$ for id.

$\mathcal{IBBE}$-$\mathcal{KEM}$.$\mathsf{Encap}(\mathcal{PP}, S \subseteq \mathscr{I})$: Takes as input a set of identities S that are the intended recipients of the message. If $|S| \leq m$, the algorithm outputs a pair (Hdr, K) where Hdr is the header and $K \in \mathscr{K}$ is the session key.

$\mathcal{IBBE}$-$\mathcal{KEM}$.$\mathsf{Decap}(\mathcal{PP}, S, \mathsf{id}, \mathcal{SK}_{\mathsf{id}}, \mathsf{Hdr})$: Inputs the public parameters, a set $S = \{\mathsf{id}_1, \ldots, \mathsf{id}_\ell\}$, an identity id, a secret key $\mathcal{SK}_{\mathsf{id}}$ corresponding to id, a header Hdr and outputs the session key K if $\mathsf{id} \in S$.

The message to be broadcast is encrypted using a symmetric encryption scheme $\mathcal{Sym} = (\mathcal{Sym}.\mathsf{Encrypt}, \mathcal{Sym}.\mathsf{Decrypt})$ with key space $\mathscr{K}$. Let $\mathcal{C} \xleftarrow{\text{R}} \mathcal{Sym}.\mathsf{Encrypt}(K, M)$ where M is the message to be broadcast and K is the session key returned by $\mathcal{IBBE}$-$\mathcal{KEM}$.Encap algorithm. The broadcast consists of the triple $(S, \mathsf{Hdr}, \mathcal{C})$. The full header is given by (S, Hdr). During decryption, the key K output by the $\mathcal{IBBE\text{-}KEM}.\mathsf{Decap}$ algorithm is used to decrypt $\mathcal{C}$ to obtain the message M as $M = \mathcal{Sym}.\mathsf{Decrypt}(K, \mathcal{C})$.

Correctness. The IBBE scheme satisfies the correctness condition if for all sets $S \subseteq \mathscr{I}$ with $|S| \leq m$, for all $\mathsf{id}_i \in S$, if $(\mathcal{PP}, \mathcal{MSK}) \xleftarrow{\mathrm{R}} \mathcal{IBBE\text{-}KEM}.\mathsf{Setup}(\kappa, \mathscr{I}, m)$, $\mathcal{SK}_{\mathsf{id}_i} \xleftarrow{\mathrm{R}} \mathcal{IBBE\text{-}KEM}.\mathsf{KeyGen}(\mathcal{MSK}, \mathsf{id}_i)$, $(\mathsf{Hdr}, K) \xleftarrow{\mathrm{R}} \mathcal{IBBE\text{-}KEM}.\mathsf{Encap}(\mathcal{PP}, S)$, the $\Pr[K = \mathcal{IBBE\text{-}KEM}.\mathsf{Decap}(\mathcal{PP}, S, \mathsf{id}_i, \mathcal{SK}_{\mathsf{id}_i}, \mathsf{Hdr})] = 1$.

2.1.5 Attribute-Based Encryption

The literature contains several variants of attribute-based encryption (ABE) and in many cases the definition and security model for ABE depends on the functionality being studied (i.e., the type of access policies realised). Since our constructions are based on the policy of acceptance by deterministic finite automata, we provide the corresponding definition only. We begin with a discussion on deterministic finite automata.

2.1.5.1 Deterministic Finite Automata

Definition 2.1.6 (Deterministic Finite Automaton)**.** A deterministic finite automaton (DFA) $\mathcal{M}$ is a 5-tuple $(Q, \Sigma, q_0, F, \delta)$ where $Q \neq \varnothing$ is a finite set of *states*, $\Sigma \neq \varnothing$ denotes the *input alphabet*, $q_0 \in Q$ is the *start state*, $\varnothing \neq F \subseteq Q$ is the set of *final states* and $\delta : Q \times \Sigma \to Q$ is called the *transition function.*

An automaton $\mathcal{M}$ is said to *accept* a string $w = w_1 \cdots w_\ell \in \Sigma^*$ if there is a sequence of states $p_0, \ldots, p_\ell$ such that $p_0 = q_0$, $\delta(p_{i-1}, w_i) = p_i$ for each $i \in \{1, \ldots, \ell\}$ and $p_\ell \in F$. The set $L = \{w \in \Sigma^* : \mathcal{M} \text{ accepts } w\}$ is the *language accepted by* $\mathcal{M}$. Languages accepted by DFAs are called *regular languages.*

It is well-known [95] that any DFA $\mathcal{M}$, one can construct $\mathcal{M}'$ such that $\mathcal{M}'$ has a unique final state and both $\mathcal{M}$ and $\mathcal{M}'$ accept the same set of languages. This is achieved by introducing a special symbol \$ at the end of the string and adding a transition from each final state in $\mathcal{M}$ to a new unique final state in $\mathcal{M}'$ based on the \$. More precisely, if $\mathcal{M} = (Q, \Sigma, q_0, F, \delta)$, then $\mathcal{M}' = (Q', \Sigma', q_0, f, \delta')$ where $Q' = Q \cup \{f\}$, $\Sigma' \Sigma \cup \{\$\}$ and the new transition function δ' is given by $\delta'(q, \sigma) = \delta(q, \sigma)$ for each $(q, \sigma) \in Q \times \Sigma$, $\delta'(f, \sigma) = f$ for all $\sigma \in \Sigma$, $\delta'(q, \$) = f$ for $q \in F$ and $\delta'(q, \$) = q$ for $q \in Q' \setminus F$. Note that the states in F are not final states in $\mathcal{M}'$. Also observe that on input $w \in \Sigma^*$ to $\mathcal{M}'$, f is not reachable (even in an intermediate step) if $\mathcal{M}$ does not accept w.

Our constructions in Chapter 8 are based on DFAs that have a unique final state. We thus use the notation $\mathcal{M} = (Q, \Sigma, q_0, q_f, \delta)$ with q_f being the final state. Transitions of an automaton $\mathcal{M} = (Q, \Sigma, q_0, q_f, \delta)$ are represented as 3-tuples of the form $t = (q_x, q_y, \sigma)$ where $\delta(q_x, \sigma) = \{q_y\}$. Let $\mathcal{T}$ denote the set of all transition tuples t.

2.1.5.2 DFA-Based ABE

The definition of DFA-based attribute-based encryption (described in [160]) is provided here.

Definition 2.1.7 (DFA-ABE)**.** A attribute-based encryption scheme $\mathscr{ABE}$ over DFA's consists of four probabilistic algorithms - $\mathscr{ABE}.\mathsf{Setup}$, $\mathscr{ABE}.\mathsf{KeyGen}$, $\mathscr{ABE}.\mathsf{Encrypt}$ and $\mathscr{ABE}.\mathsf{Decrypt}$.

$\mathscr{ABE}.\mathsf{Setup}(\kappa, \Sigma)$**:** takes as input a security parameter κ, generates the public parameters $\mathcal{PP}$ and the master secret $\mathcal{MSK}$ based on the input alphabet Σ. Σ is part of $\mathcal{PP}$.

$\mathscr{ABE}.\mathsf{KeyGen}(\mathcal{MSK}, \mathcal{M})$**:** receives the description of a DFA $\mathcal{M}$ and master secret $\mathcal{MSK}$ and outputs a secret key $\mathcal{SK}_{\mathcal{M}}$ corresponding to $\mathcal{M}$.

$\mathscr{ABE}.\mathsf{Encrypt}(\mathcal{PP}, m, w)$**:** inputs a message m, a string $w = w_1 w_2 \cdots w_\ell$ over Σ and returns a ciphertext $\mathcal{C}$ (which also contains w).

$\mathscr{ABE}.\mathsf{Decrypt}(\mathcal{C}, \mathcal{SK}_{\mathcal{M}})$**:** inputs a ciphertext $\mathcal{C}$ and secret key $\mathcal{SK}_{\mathcal{M}}$. If $\mathsf{Accept}(\mathcal{M}, w) = 1$, the algorithm returns m; otherwise, returns $\perp$ indicating failure.

Correctness. $\mathscr{ABE}$ satisfies the correctness condition if for all $(\mathcal{PP}, \mathcal{MSK}) \xleftarrow{\text{R}} \mathscr{ABE}.\mathsf{Setup}(\kappa, \Sigma)$, for all $\mathcal{M}$, $\mathcal{SK}_{\mathcal{M}} \xleftarrow{\text{R}} \mathscr{ABE}.\mathsf{KeyGen}(\mathcal{MSK}, \mathcal{M})$, for all $w \in \Sigma^*$, $\mathcal{C} \xleftarrow{\text{R}} \mathscr{ABE}.\mathsf{Encrypt}(\mathcal{PP}, m, w)$, the $\Pr[K = \mathscr{ABE}.\mathsf{Decrypt}(\mathcal{C}, \mathcal{SK}_{\mathcal{M}})] = 1$.

This is a key-policy attribute-based encryption scheme. One can also define a ciphertext-policy scheme but we do not provide the definitions.

2.2 Formalising Security Goals

A natural goal of an attacker against any encryption system would be to learn, given a ciphertext, anything meaningful about the message. We call the system secure if it can guard against such attacks i.e., a ciphertext must not reveal any information about the plaintext[1]. This concept was first formalised by Goldwasser and Micali [89] and termed as *semantic security.* They further showed that semantic security is equivalent to a notion called *indistinguishability of ciphertexts* (also defined in [89]). An important requirement for discussing indistinguishability in the context of public key systems is that the encryption algorithm be probabilistic. This will apply to (H)IBE and (IB)BE schemes as they are public key systems with richer structure.

We now define the notion of indistinguishability of ciphertexts in identity-based systems. An attacker has access to all public information (public parameters $\mathcal{PP}$). In order to model collusion attacks, the attacker is provided the secret keys $(\mathcal{SK}_{\mathsf{id}_1}, \ldots, \mathcal{SK}_{\mathsf{id}_q})$ corresponding to a group of identities $(\mathsf{id}_1, \ldots, \mathsf{id}_q)$ of the attacker's choice. Now, let $\mathscr{M}$ denote the message space for the encryption scheme. Since encryption is probabilistic, there could be several possible ciphertexts corresponding to a single message for a fixed target identity. Let $\mathscr{C}_M$ denote the ciphertext space for a message $M \in \mathscr{M}$. The encryption algorithm, on input $\mathcal{PP}$ and M, can be viewed as sampling a ciphertext from $\mathscr{C}_M$ according to some distribution. The attacker specifies two messages M_0 and M_1. A ciphertext for one of the two messages is given back to the attacker and asked to guess which message was encrypted. That is, a bit $\beta \xleftarrow{\text{U}} \{0,1\}$ is picked and $\widehat{\mathcal{C}}$ generated as an encryption of M_β under $\mathcal{PP}$. The attacker is given only $\widehat{\mathcal{C}}$ and asked to guess β. The system is said to be secure if

[1]We assume that the attacker is only *passive* i.e., it only tries to learn information rather than tampering with ciphertexts whence it is termed *active*

the attacker is unsuccessful in guessing β with probability significantly away from 1/2. This notion of security is termed IND-ID-CPA (short for indistinguishability under the chosen plaintext attack).

An even stronger setting is when the attacker is provided access to a *decryption oracle* i.e., it is allowed to make decryption queries on ciphertexts of its choice. Security against such an attacker is termed IND-ID-CCA (indistinguishability against chosen ciphertext attacks). IND-ID-CCA is considered the strongest notion of security. In Chapter 4, we will discuss generic and non-generic transformations from CPA-secure schemes to CCA-secure scheme in the identity-based setting. Most of these methods are reasonably efficient. Hence only consider only CPA-security and provide the corresponding definitions here.

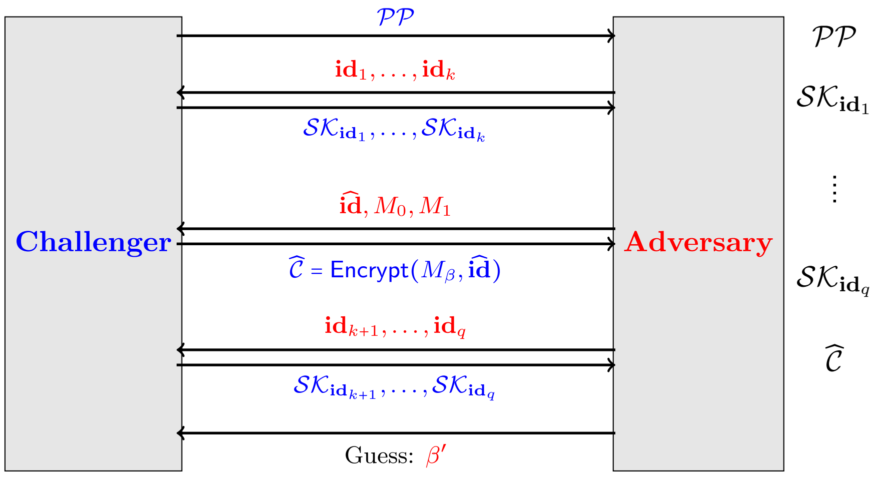
We now present the formal security definitions for the various primitives we consider in this work. The security definition for IBE is omitted. It can be derived as special case of HIBE security with the number of levels set to 1.

Note. The terms attacker and adversary are used interchangeably throughout this work.

2.2.1 HIBE

2.2.1.1 The Standard Model for CPA-Security

CPA-security for IBE schemes was formalised in [25]. The corresponding model for HIBE schemes was first described by Gentry and Silverberg [86]). This model is based on the assumption that the distribution of secret keys generated by KeyGen algorithm is identical to the distribution of keys output by Delegate algorithm. The assumption applies to a large number of schemes. We now describe the model via the following security game, called ind-cpa (depicted in Figure 2.1).



$\mathcal{PP}$
$\mathbf{id}_1, \ldots, \mathbf{id}_k$
$\mathcal{SK}_{\mathbf{id}_1}, \ldots, \mathcal{SK}_{\mathbf{id}_k}$
$\widehat{\mathbf{id}}, M_0, M_1$
Challenger
Adversary
$\widehat{\mathcal{C}} = \mathsf{Encrypt}(M_\beta, \widehat{\mathbf{id}})$
$\mathbf{id}_{k+1}, \ldots, \mathbf{id}_q$
$\mathcal{SK}_{\mathbf{id}_{k+1}}, \ldots, \mathcal{SK}_{\mathbf{id}_q}$
Guess: β'
$\mathcal{PP}$
$\mathcal{SK}_{\mathbf{id}_1}$
$\vdots$
$\mathcal{SK}_{\mathbf{id}_q}$
$\widehat{\mathcal{C}}$

Figure 2.1: HIBE Security Game

There are several stages of the game which are described as follows. An attacker/adversary is an algorithm denoted throughout this work as $\mathscr{A}$.

Setup: The challenger runs the Setup algorithm of the HIBE and gives the public parameters to

$\mathscr{A}$.

Phase 1: $\mathscr{A}$ makes a number of key extraction queries adaptively. For a query on an identity vector $\mathbf{id}$, the challenger responds with a key $\mathcal{SK}_{\mathbf{id}}$.

Challenge: $\mathscr{A}$ provides two message M_0, M_1 and identity $\widehat{\mathbf{id}}$ as challenge with the restriction that no prefix of $\widehat{\mathbf{id}}$ has been queried in **Phase 1**. The challenger then chooses a bit β uniformly at random from $\{0,1\}$ and returns an encryption $\widehat{\mathcal{C}}$ of M_β under $\widehat{\mathbf{id}}$ to $\mathscr{A}$.

Phase 2: $\mathscr{A}$ issues more key extraction queries as in **Phase 1** with the restriction that no queried identity $\mathbf{id}$ is a prefix of $\widehat{\mathbf{id}}$.

Guess: $\mathscr{A}$ outputs a bit β'.

If $\beta = \beta'$, then $\mathscr{A}$ wins the game. The advantage of $\mathscr{A}$ in breaking the security of the HIBE scheme in the game ind-cpa given by

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathrm{HIBE}}(\mathscr{A}) = \left|\Pr[\beta = \beta'] - \frac{1}{2}\right|.$$

The HIBE scheme is said to be (ε, t, q)-IND-ID-CPA secure if every t-time adversary making at most q queries has $\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathrm{HIBE}}(\mathscr{A}) \leq \varepsilon$.

Selective Identity Model. The definition of IND-ID-CPA security is *adaptive* in nature since the adversary is allowed to choose the target identity at any point during the game. The choice could depend on the secret keys received earlier. A weaker notion called *selective* security requires the attacker to commit to the target identity before even looking at the public parameters. The analogous game, named ind-cpa, consists of a stage **Initialise** before **Setup** in which $\mathscr{A}$ commits to the target identity vector $\widehat{\mathbf{id}}$. Advantage is defined similar to the adaptive setting. We call a HIBE scheme selectively secure or (ε, t, q)-IND-sID-CPA secure[2] if any adversary running in time at most t has $\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathrm{HIBE}}(\mathscr{A}) \leq \varepsilon$.

For all the security definitions discussed in the sequel, selective versions can be defined similar to IND-sID-CPA. None of our schemes are selectively secure. We require this notion only in our discussions on the development of the subject in Chapter 4.

2.2.1.2 Shi-Waters Model for CPA-Security

Shi and Waters [146] consider a complete security definition for HIBE schemes where the distribution of a secret key actually depends on the delegation path. We name this notion of security IND-ID-CPA2 and provide a description here. It is used for proving security of a HIBE scheme we present in Chapter 5.

As usual, the security for a HIBE scheme is modelled as a game ind-cpa2 between an adversary $\mathscr{A}$ and a simulator. Following are the different phases of the security game.

Setup: The challenger runs the Setup algorithm of the HIBE and gives the public parameters to $\mathscr{A}$. It also initialises a set $S = \varnothing$ which denotes the set of secret keys it has created but not revealed.

Phase 1: $\mathscr{A}$ makes a number of queries of the following types adaptively.

[2] "sID" denotes that the challenge identity is chosen selectively.

- **Create:** The adversary $\mathscr{A}$ provides an identity $\mathbf{id}$ for which the challenger creates a key $\mathcal{SK}_{\mathbf{id}}$ but does not give it to $\mathscr{A}$. The challenger adds the secret key to S.

- **Delegate:** $\mathscr{A}$ specifies a secret key $\mathcal{SK}_{\mathbf{id}}$ in S and provides an identity id to the challenger. The challenger runs the delegation algorithm of the HIBE with inputs $\mathcal{PP}, \mathcal{SK}_{\mathbf{id}}, \mathsf{id}$ and adds the resulting key for $(\mathbf{id}, \mathsf{id})$ to S.

- **Reveal:** $\mathscr{A}$ specifies an element $\mathcal{SK}$ in S. The challenger removes $\mathcal{SK}$ from S and returns it to $\mathscr{A}$.

Challenge: $\mathscr{A}$ provides a challenge identity $\widehat{\mathbf{id}}$ and two equal length messages M_0 and M_1 to the challenger with the restriction that $\widehat{\mathbf{id}}$ should not have been queried in **Phase 1**. The challenger then chooses a bit β uniformly at random from $\{0,1\}$ and returns an encryption $\widehat{\mathcal{C}}$ of M_β for $\widehat{\mathbf{id}}$ to $\mathscr{A}$.

Phase 2: $\mathscr{A}$ issues more key extraction queries as in **Phase 1** with the restriction that any identity tuple for which the key is revealed is not a prefix of $\widehat{\mathbf{id}}$.

Guess: $\mathscr{A}$ outputs a bit β'. $\mathscr{A}$ wins the above game if $\beta = \beta'$. The advantage of $\mathscr{A}$ in breaking the security of the HIBE scheme in the above game is given by

$$\mathsf{Adv}_{\mathrm{HIBE}}^{\mathsf{ind\text{-}cpa2}}(\mathscr{A}) = \left|\Pr[\beta = \beta'] - \frac{1}{2}\right|.$$

We say that the HIBE is $(\varepsilon, t, q_R, q_A)$-$\mathsf{IND\text{-}ID\text{-}CPA2}$ secure if every t-time adversary making at most q_R reveal queries and q_A create and delegate queries has $\mathsf{Adv}_{\mathrm{HIBE}}^{\mathsf{ind\text{-}cpa2}}(\mathscr{A}) \le \varepsilon$.

2.2.1.3 Anonymity

Another important notion of security for a HIBE scheme is that of *ciphertext anonymity.* The fundamental requirement for a scheme to be called anonymous is that looking at a ciphertext and public parameters, one must not be able to guess the identity to which the ciphertext is encrypted. Anonymity is also modelled in terms of an indistinguishability game similar to CPA-security. The difference lies in the challenge phase – instead of providing two messages and an identity vector, the attacker specifies two identity vectors and a single message. The challenge ciphertext is created under one of the identities. The goal of the attacker is to guess which of the two identities was used to encrypt the challenge ciphertext.

Anonymity in HIBE schemes was first introduced and formalised by Abdalla et al. in [1]. Instead of treating anonymity separately, we follow a combined model which captures both anonymity and CPA-security simultaneously. The game corresponding to this model, which we call $\mathsf{ano\text{-}ind\text{-}cpa}$, is equivalent to the standard security notions for $\mathsf{IND\text{-}ID\text{-}CPA}$-security and anonymity taken together and has been used earlier in [70, 63].

Setup: The challenger runs the Setup algorithm of the HIBE and gives the public parameters to $\mathscr{A}$.

Phase 1: $\mathscr{A}$ makes a number of key extraction queries adaptively. For a query on an identity vector $\mathbf{id}$, the challenger responds with a key $\mathcal{SK}_{\mathbf{id}}$.

Challenge: $\mathscr{A}$ provides two message-identity pairs $(M_0, \widehat{\mathbf{id}}_0)$ and $(M_1, \widehat{\mathbf{id}}_1)$ as challenge with the restriction that neither $\widehat{\mathbf{id}}_0, \widehat{\mathbf{id}}_1$ nor any of their prefixes should have been queried in **Phase 1**. The challenger then chooses a bit β uniformly at random from $\{0,1\}$ and returns an encryption $\widehat{\mathcal{C}}$ of M_β under the identity $\widehat{\mathbf{id}}_\beta$ to $\mathscr{A}$.

Phase 2: $\mathscr{A}$ issues more key extraction queries as in **Phase 1** with the restriction that no queried identity $\mathbf{id}$ is a prefix of either $\widehat{\mathbf{id}}_0$ or $\widehat{\mathbf{id}}_1$.

Guess: $\mathscr{A}$ outputs a bit β'.

If $\beta = \beta'$, then $\mathscr{A}$ wins the game. The advantage of $\mathscr{A}$ in breaking the security of the HIBE scheme in the game $\mathsf{ano\text{-}ind\text{-}cpa}$ given by

$$\mathsf{Adv}^{\mathsf{ano\text{-}ind\text{-}cpa}}_{\mathrm{HIBE}}(\mathscr{A}) = \left|\Pr[\beta = \beta'] - \frac{1}{2}\right|.$$

The HIBE scheme is said to be (ε, t, q)-$\mathsf{ANO\text{-}IND\text{-}ID\text{-}CPA}$ secure if every t-time adversary making at most q queries has $\mathsf{Adv}^{\mathsf{ano\text{-}ind\text{-}cpa}}_{\mathrm{HIBE}}(\mathscr{A}) \leq \varepsilon$.

2.2.2 PKBE

Again, security is defined in terms of indistinguishability of ciphertexts. The target of an attacker is a set of users $\widehat{\mathcal{S}} \subseteq \mathcal{N}$. The model must also capture collusion attacks. In other words, the attacker gets hold of secret keys for some users outside of $\widehat{\mathcal{S}}$. Described below is adaptive CPA security of a PKBE scheme is defined via the $\mathsf{ind\text{-}be\text{-}cpa}$ between an adversary $\mathscr{A}$ and a challenger.

Setup: The challenger runs the Setup algorithm and gives the public key $\mathcal{PK}$ to $\mathscr{A}$.

Phase 1: $\mathscr{A}$ adaptively issues private key queries for a number of users from $\{1, \ldots, n\}$.

Challenge: $\mathscr{A}$ specifies two messages M_0, M_1 and a challenge set $\widehat{\mathcal{S}}$ such that, for every private key query i, $i \notin \widehat{\mathcal{S}}$. The challenger chooses $\beta \xleftarrow{\mathrm{U}} \{0,1\}$ and returns to $\mathscr{A}$ the encryption of M_β for the set $\widehat{\mathcal{S}}$.

Phase 2: $\mathscr{A}$ issues more private key queries but with the restriction that for every query i, $i \notin \widehat{\mathcal{S}}$.

Guess: The adversary outputs its guess β' for β.

The adversary wins the above game if $\beta = \beta'$. The advantage of $\mathscr{A}$ in breaking the security of the BE scheme is defined in terms of the probability of the event that $\beta = \beta'$ in the $\mathsf{ind\text{-}be\text{-}cpa}$ as shown below.

$$\mathsf{Adv}^{\mathsf{ind\text{-}be\text{-}cpa}}_{\mathrm{PKBE}}(\mathscr{A}) = \left|\Pr[\beta = \beta'] - \frac{1}{2}\right|.$$

The BE scheme is said to be (ε, t, q)-$\mathsf{IND\text{-}B\text{-}CPA}$[3] secure if every t-time adversary making at most q queries has $\mathsf{Adv}^{\mathsf{ind\text{-}be\text{-}cpa}}_{\mathrm{BE}}(\mathscr{A}) \leq \varepsilon$.

2.2.3 IBBE

The security model for PKBE schemes naturally extends to IBBE schemes ([10, 48]). The IBBE security model allows an adversary to specify a target set of identities that it wishes to attack. The

[3] "B" indicates broadcast setting.

model also allows the adversary to corrupt entities and obtain the decryption keys corresponding to their identities with the restriction that the corrupted set of identities is disjoint from the target set of identities. Depending on when the adversary specifies the target set leads to two different security notions. The weaker notion, called selective-identity security (summarised sID), requires the adversary to specify the target set *before* it can corrupt any entity. The stronger notion, called adaptive-identity security (summarised aID), allows the adversary to specify the target set after it has corrupted a set of identities (and also allows it to corrupt identities after specifying the target set). It is desirable to obtain schemes which are secure against adaptive-identity attacks. Since our constructions will be defined within the KEM-DEM framework, we only provide the security definition for IBBE-KEM systems which can be obtained from that of IBBE with a bit of tweaking.

Adaptive security against chosen plaintext attacks in IBBE-KEM systems is defined via the following game ind-ibbe-cpa between an adversary $\mathscr{A}$ and a challenger.

Setup: The challenger runs the Setup algorithm of the IBBE and gives the public parameters to $\mathscr{A}$.

Key Extraction Phase 1: $\mathscr{A}$ makes a number of key extraction queries adaptively. For a query on an identity vector id, the challenger responds with a key $\mathcal{SK}_{\mathsf{id}}$.

Challenge: $\mathscr{A}$ provides a challenge set $\widehat{S}$ with the restriction that if id is queried in the key extraction phase 1, then $\mathsf{id} \notin \widehat{S}$. The challenger computes $(\widehat{\mathsf{Hdr}}, K_0) \xleftarrow{\text{R}} \mathsf{Encap}(\mathcal{PP}, \widehat{S})$ and chooses $K_1 \xleftarrow{\text{U}} \mathscr{K}$. It then chooses a bit β uniformly at random from $\{0,1\}$ and returns $(\widehat{\mathsf{Hdr}}, K_\beta)$ to $\mathscr{A}$.

Key Extraction Phase 2: $\mathscr{A}$ makes more key extraction queries with the restriction that it cannot query a key for any identity in $\widehat{S}$.

Guess: $\mathscr{A}$ outputs a bit β'.

If $\beta = \beta'$, then $\mathscr{A}$ wins the game. The advantage of $\mathscr{A}$ of the IBBE scheme in winning the ind-ibbe-cpa is given by

$$\mathsf{Adv}^{\text{ind-cpa}}_{\text{IBBE}}(\mathscr{A}) = \left|\Pr[\beta = \beta'] - \frac{1}{2}\right|.$$

The IBBE scheme is said to be (ε, t, q)-IND-BID-CPA secure if every t-time adversary making at most q key extraction queries has $\mathsf{Adv}^{\text{ind-ibbe-cpa}}_{\text{IBBE}}(\mathscr{A}) \le \varepsilon$.

2.2.4 DFA-Based ABE

Security is modelled based on the notion of indistinguishability of ciphertexts under a chosen plaintext attack (CPA) ([160]). It is defined via a game ind-abe-cpa between an adversary $\mathscr{A}$ and a challenger consisting of several stages.

Setup: The challenger runs the $\mathscr{ABE}$.Setup algorithm of the ABE scheme and gives the public parameters to $\mathscr{A}$.

Phase 1: $\mathscr{A}$ makes a number of key extraction queries adaptively. For a query on automaton $\mathcal{M}$, the challenger runs the $\mathscr{ABE}$.KeyGen algorithm of the ABE scheme and returns its output $\mathcal{SK}_{\mathcal{M}}$ to $\mathscr{A}$.

Challenge: $\mathscr{A}$ provides two messages pairs m_0, m_1 and a challenge string $\widehat{w} = \widehat{w}_1\widehat{w}_2\cdots\widehat{w}_{\widehat{\ell}}$ subject to the condition that $\mathscr{A}$ does not request keys for any automaton that accepts $\widehat{w}$ in **Phase 1** or

Phase 2. The challenger then picks $\beta \xleftarrow{\mathrm{U}} \{0,1\}$ and returns an encryption $\widehat{\mathcal{C}}$ of m_β under the string $\widehat{w}$ to $\mathscr{A}$.

Phase 2: $\mathscr{A}$ issues more key extraction queries as in **Phase 1** with the restriction that none of the automata that are queried accept $\widehat{w}$.

Guess: $\mathscr{A}$ outputs a bit β'.

In the selective model, there is a stage **Initialise** before **Setup** in which the adversary commits to the input alphabet Σ and the challenge string $\widehat{w}$. Call this game ind-abe-cpa.

If $\beta = \beta'$, then $\mathscr{A}$ wins the game. The advantage of $\mathscr{A}$ in breaking the security of the ABE scheme in the ind-abe-cpa game is given by

$$\mathsf{Adv}_{\mathrm{ABE}}^{\text{ind-abe-cpa}}(\mathscr{A}) = \left|\Pr[\beta = \beta'] - \frac{1}{2}\right|.$$

The ABE scheme is said to be (ε, t, ν)-IND-STR-CPA secure[4] (secure under chosen plaintext attack) if for every adversary $\mathscr{A}$ making at most ν queries and whose running time is t, it holds that $\mathsf{Adv}_{\mathrm{ABE}}^{\text{IND-STR-CPA}}(\mathscr{A}) \leq \varepsilon$. Similarly for the selective case, the notion of IND-sSTR-CPA-security can be defined.

2.3 Mathematical Preliminaries

We define some notation first. The finite field of order q is denoted as $\mathbb{F}_q$. For a positive integer N, $\mathbb{Z}_N$ denotes the ring of integers modulo N. If $\mathbb{G}$ is a finite cyclic group, then $\mathbb{G}^\times$ denotes the set of generators of $\mathbb{G}$. Let $P \in \mathbb{G}^\times$ for some additive cyclic group $\mathbb{G}$. For a vector $(Q_1, \ldots, Q_d) \in \mathbb{G}^d$, $d \geq 1$, define $\mathsf{dlog}_P(Q_1, \ldots, Q_d)$ as $(x_1, \ldots, x_d) \in (\mathbb{Z}_{|\mathbb{G}|})^d$ such that $Q_i = x_i P$ for all $i \in [1, d]$. For an asymmetric pairing function $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, elements of $\mathbb{G}_1$ will be identified by subscript 1 and elements of $\mathbb{G}_2$ by subscript 2.

2.3.1 An Overview of Elliptic Curve Pairings.

A bilinear pairing is a 7-tuple $\mathcal{G} = (N, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$ where $\mathbb{G}_1 = \langle P_1 \rangle$, $\mathbb{G}_2 = \langle P_2 \rangle$ are written additively and $\mathbb{G}_T$ is a multiplicatively written group, all having the same order N and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a map with the following properties.

1. *Bilinear:* For $P_1, Q_1 \in \mathbb{G}_1$ and $P_2, Q_2 \in \mathbb{G}_2$, the following holds:
   $e(P_1, P_2 + Q_2) = e(P_1, P_2)e(P_1, Q_2)$ and $e(P_1 + Q_1, P_2) = e(P_1, P_2)e(Q_1, P_2)$.
2. *Non-degenerate:* If $e(P_1, P_2) = 1_T$, the identity element of $\mathbb{G}_T$, then either P_1 is the identity of $\mathbb{G}_1$ or P_2 is the identity of $\mathbb{G}_2$.
3. *Efficiently computable:* The function e should be efficiently computable.

Practical examples of such maps are the Weil pairing and Tate pairing on elliptic curves over finite fields. $\mathbb{G}_1$ and $\mathbb{G}_2$ are groups of elliptic curve points and $\mathbb{G}_T$ is a subgroup of the multiplicative

[4] The abbreviation “STR” stands for string. “sSTR” denotes that the challenge string is chosen selectively.

group of a related finite field. Let $\mathbb{E}$ denote an elliptic curve over a finite field $\mathbb{F}_q$. The group $\mathbb{G}_1$ is a subgroup of $\mathbb{E}(\mathbb{F}_q)$ and $\mathbb{G}_2$ is typically chosen to be a subgroup of $\mathbb{E}(\mathbb{F}_{q^k})$. Group $\mathbb{G}_T$ would be a subgroup of $\mathbb{F}_{q^k}^{\times}$. The integer k is called the *embedding degree* of the pairing. A lot of factors influence the choice of a pairing. These include

- size of representation of elements in the three groups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$,
- complexity of group operation in $\mathbb{G}_1$ and $\mathbb{G}_2$,
- efficiency of evaluating the pairing function e.

These factors are determined chiefly by the size of the base field q, the embedding degree k and the size of the groups N. These quantities are chosen keeping in mind the desired security level and various algorithms to solve the discrete logarithm problem in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$.

Types of Pairings. Pairings can be broadly categorised into two types based on the common group order N.

Prime-order pairings: In this case, N is a prime number. This type of pairing is further classified into three types in the literature [153, 76].

Type-1 In this type, the groups $\mathbb{G}_1$ and $\mathbb{G}_2$ are the same.

Type-2 $\mathbb{G}_1 \neq \mathbb{G}_2$ and an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ is known.

Type-3 Here, $\mathbb{G}_1 \neq \mathbb{G}_2$ and no efficiently computable isomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$ are known.

It has been reported in the literature [153, 76, 44], that among the different types of pairings, it is the Type-3 pairings which provide the most compact parameter sizes and the most efficient algorithms. Further, Type-1 pairings are usually defined over low characteristics fields and recent advances [12, 101, 92, 3, 93] in algorithms for discrete log computations over such fields have raised serious question marks about the security of Type-1 pairings [75]. From both efficiency and security considerations, constructions based on Type-3 pairings are desirable.

Composite-order pairings: Here the common group order is a composite number and the pairing is symmetric ($\mathbb{G}_1 = \mathbb{G}_2$). This type of pairings were first introduced by Boneh, Goh and Nissim [29]. Their construction requires the group order to be square-free and not divisible by 3. Also, there are no known asymmetric variants of such pairings. Composite-order pairings have the property of orthogonality described next. Let $\mathcal{G} = (N = p_1p_2, \mathbb{G}, \mathbb{G}, \mathbb{G}_T, e, P)^5$ is a composite-order pairing. $P_1 \in \mathbb{G}_{p_1}$, the subgroup of $\mathbb{G}$ of order p_1. Similarly define $\mathbb{G}_{p_2}$ and let $P_2 \in \mathbb{G}_{p_2}$. The orthogonality property is that $e(P_1, P_2) = 1_T$, the identity of $\mathbb{G}_T$. In general, if $N = p_1p_2 \cdots p_k$, then for any two subgroups $\mathbb{G}_{N_1}, \mathbb{G}_{N_2}$ of $\mathbb{G}$ (of order $N_1, N2$ respectively) with $\gcd(N_1, N_2) = 1$, any two points $P \in \mathbb{G}_{N_1}$ and $Q \in \mathbb{G}_{N_2}$ would be orthogonal i.e., $e(P, Q) = 1_T$.

[5] We consider the group order to be a product of two primes for simplicity. The group order could be any square free integer.

Due to this property, composite order groups are very useful for obtaining schemes with dual system proofs. On the other hand, they are very inefficient in comparison to Type-1 prime order groups, let alone Type-3 pairings. reasonable security level, N must be chosen to be quite large.

Note: We write p as the order (in place of N) for a prime-order pairing in the rest of the material.

2.3.2 Hardness Assumptions

2.3.2.1 Type-3 Pairings

Most of our constructions are based on Type-3 pairings. We state some hardness assumptions in Type-3 pairings that are used in our proofs. Let $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$ be an asymmetric pairing. All the assumptions are decisional in nature. We first define a generic static decision problem Π over a $\mathcal{G}$.

Let $\mathscr{A}$, a probabilistic polynomial time (PPT) algorithm $\mathscr{A}$ that outputs 0 or 1. Denote by $\mathcal{D}$, a distribution consisting of a constant number of elements from $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$. Let $\mathcal{T}_1, \mathcal{T}_2$ be two distributions over one of the three groups. The goal of $\mathscr{A}$ is to distinguish between the two distributions – $(\mathcal{D}, \mathcal{T}_1)$ and $(\mathcal{D}, \mathcal{T}_2)$. The advantage of $\mathscr{A}$ in solving Π is given by

$$\mathsf{Adv}^{\Pi}_{\mathcal{G}}(\mathscr{A}) = |\Pr[\mathscr{A}(\mathcal{D}, \mathcal{T}_1) = 1] - \Pr[\mathscr{A}(\mathcal{D}, \mathcal{T}_2) = 1]|.$$

We say that (ε, t)-Π assumption holds if for any t-time algorithm $\mathscr{A}$, $\mathsf{Adv}^{\Pi}_{\mathcal{G}}(\mathscr{A}) \leq \varepsilon$.

Next the required assumptions are stated as instantiations of Π by suitably defining $\mathcal{D}$ and $\mathcal{T}_1, \mathcal{T}_2$.

DDH1: Decisional Diffie-Hellman in $\mathbb{G}_1$.

$F_1 \xleftarrow{\text{U}} \mathbb{G}_1^{\times}$; $F_2 \xleftarrow{\text{U}} \mathbb{G}_2^{\times}$, $x, y, \mu \xleftarrow{\text{U}} \mathbb{Z}_p$;
$\mathcal{D} = (\mathcal{G}, F_1, xF_1, yF_1, F_2)$,
$\mathcal{T}_1 = xyF_1, \quad \mathcal{T}_2 = (xy + \mu)F_1$.

DDH2: Decisional Diffie-Hellman in $\mathbb{G}_2$.

$F_1 \xleftarrow{\text{U}} \mathbb{G}_1^{\times}$; $F_2 \xleftarrow{\text{U}} \mathbb{G}_2^{\times}$, $x, y, \gamma \xleftarrow{\text{U}} \mathbb{Z}_p$;
$\mathcal{D} = (\mathcal{G}, F_1, F_2, xF_2, yF_2)$,
$\mathcal{T}_1 = xyF_2, \quad \mathcal{T}_2 = (xy + \gamma)F_2$.

SXDH: Symmetric eXternal Diffie-Hellman.

SXDH in $\mathcal{G}$ is DDH1 and DDH2 assumptions combined.

DDH2v: A variant of DDH2.

$F_1 \xleftarrow{\text{U}} \mathbb{G}_1^{\times}$; $F_2 \xleftarrow{\text{U}} \mathbb{G}_2^{\times}$, $x_1, x_2, d, z, \gamma \xleftarrow{\text{U}} \mathbb{Z}_p$;
$\mathcal{D} = (\mathcal{G}, F_1, dF_1, dzF_1, zx_1F_1, F_2, dF_2, x_1F_2, x_2F_2)$,
$\mathcal{T}_1 = x_1x_2F_2, \quad \mathcal{T}_2 = (x_1x_2 + \gamma)F_2$.

LW1: Assumption 1 of Lewko and Waters [114].

$F_1 \overset{U}{\longleftarrow} \mathbb{G}_1^\times$; $F_2 \overset{U}{\longleftarrow} \mathbb{G}_2^\times$, $a, b, s \overset{U}{\longleftarrow} \mathbb{Z}_p$, $\mu \overset{U}{\longleftarrow} \mathbb{Z}_p$;
$\mathcal{D} = (\mathcal{G}, F_1, bsF_1, sF_1, aF_1, ab^2F_1, bF_1, b^2F_1, asF_1, b^2sF_1, b^3F_1, b^3sF_1, F_2, bF_2)$,
$\mathcal{T}_1 = ab^2sF_1$, $\quad \mathcal{T}_2 = (ab^2s + \mu)F_1$.

LW2: Assumption 2 of Lewko and Waters [114].

$F_1 \overset{U}{\longleftarrow} \mathbb{G}_1^\times$; $F_2 \overset{U}{\longleftarrow} \mathbb{G}_2^\times$, $d, b, c, x \overset{U}{\longleftarrow} \mathbb{Z}_p$, $\gamma \overset{U}{\longleftarrow} \mathbb{G}_2$;
$\mathcal{D} = (\mathcal{G}, F_1, dF_1, d^2F_1, bxF_1, dbxF_1, d^2xF_1, F_2, dF_2, bF_2, cF_2)$,
$\mathcal{T}_1 = bcF_2$, $\quad \mathcal{T}_2 = (bc + \gamma)F_2$.

Assumption A1.

$F_1 \overset{U}{\longleftarrow} \mathbb{G}_1^\times$; $F_2 \overset{U}{\longleftarrow} \mathbb{G}_2^\times$; $a, z, d, s, x, \mu \overset{U}{\longleftarrow} \mathbb{Z}_p$;
$\mathcal{D} = (\mathcal{G}, F_1, zF_1, dzF_1, azF_1, adzF_1, szF_1, F_2, zF_2, aF_2, xF_2, (dz - ax)F_2)$,
$\mathcal{T}_1 = sdzF_1$, $\quad \mathcal{T}_2 = (sdz + \mu)F_1$.

DLin1: Decision Linear in group $\mathbb{G}_1$.

$P_1, F_1, H_1 \overset{U}{\longleftarrow} \mathbb{G}_1^\times$; $P_2, F_2, H_2 \overset{U}{\longleftarrow} \mathbb{G}_2^\times$; $x_1, x_2, \mu \overset{U}{\longleftarrow} \mathbb{Z}_p$;
$\mathcal{D} = (\mathcal{G}, P_1, F_1, H_1, P_2, F_2, H_2, x_1P_1, x_2F_1)$,
$\mathcal{T}_1 = (x_1 + x_2)H_1$, $\quad \mathcal{T}_2 = (x_1 + x_2 + \mu)H_1$.

DLin2: Decision Linear in group $\mathbb{G}_2$.

$P_1, F_1, H_1 \overset{U}{\longleftarrow} \mathbb{G}_1^\times$; $P_2, F_2, H_2 \overset{U}{\longleftarrow} \mathbb{G}_2^\times$; $x_1, x_2, \gamma \overset{U}{\longleftarrow} \mathbb{Z}_p$;
$\mathcal{D} = (\mathcal{G}, P_1, F_1, H_1, P_2, F_2, H_2, x_1P_2, x_2F_2)$,
$\mathcal{T}_1 = (x_1 + x_2)H_2$, $\quad \mathcal{T}_2 = (x_1 + x_2 + \gamma)H_2$.

DBDH: Decisional Bilinear Diffie-Hellman.

$F_1 \overset{U}{\longleftarrow} \mathbb{G}_1^\times$; $F_2 \overset{U}{\longleftarrow} \mathbb{G}_2^\times$; $x_1, x_2, x_3, \eta \overset{U}{\longleftarrow} \mathbb{Z}_p$;
$\mathcal{D} = (\mathcal{G}, F_1, x_1F_1, x_2F_1, x_3F_1, F_2, x_1F_2, x_2F_2, x_3F_2)$,
$\mathcal{T}_1 = e(F_1, F_2)^{x_1x_2x_3}$, $\quad \mathcal{T}_2 = e(F_1, F_2)^{x_1x_2x_3+\eta}$.

DBDH-3: Decisional Bilinear Diffie-Hellman in Type-3 pairings [44].

$F_1 \overset{U}{\longleftarrow} \mathbb{G}_1^\times$; $F_2 \overset{U}{\longleftarrow} \mathbb{G}_2^\times$; $x_1, x_2, x_3 \overset{U}{\longleftarrow} \mathbb{Z}_p$; $\eta \overset{U}{\longleftarrow} \mathbb{G}_T$;
$(\mathcal{G}, F_1, x_1F_1, x_2F_1, x_3F_1, F_2, x_1F_2, x_2F_2)$,
$\mathcal{T}_1 = e(F_1, F_2)^{x_1x_2x_3}$, $\quad \mathcal{T}_2 = e(F_1, F_2)^{x_1x_2x_3+\eta}$.

2.3.2.2 Composite-Order Pairings

Our ABE construction (described in Chapter 8) is build upon composite-order pairings. We now state the complexity assumptions used in our security proof. Before proceeding, we introduce some notation.

A composite order pairing is represented as a tuple $\mathcal{G} = (p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, e, G)$ where p_1, p_2, p_3 prime, $|\mathbb{G}| = |\mathbb{G}_T| = N = p_1p_2p_3$, $\mathbb{G} = \langle G \rangle$ and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is the pairing function. Define $\mathcal{G}_{\text{pub}} = (N, \mathbb{G}, \mathbb{G}_T, e, G)$ where $N = p_1p_2p_3$. Also let $\mathbb{G}_B$ denote the subgroup of order B of $\mathbb{G}$. This representation is particular to those pairings where the group order is a product of three distinct primes. In general, the order could be any composite number that is hard to factor. We denote elements of groups $\mathbb{G}_{p_2}, \mathbb{G}_{p_3}$ with subscripts 2 and 3 respectively. Elements of $\mathbb{G}_{p_1}$ and $\mathbb{G}$ are written without a subscript. The meaning will be clear from the context.

We state two Decisional SubGroup (DSG) assumptions followed by an assumption that we term SubGroup Diffie Hellman (SGDH) in composite order groups equipped with a bilinear pairing. We stick to the notation used in Section 2.3.2.1 for presenting the assumptions.

Assumption DSG1

$$\begin{aligned}
&P \xleftarrow{\text{U}} \mathbb{G}_{p_1};\ P_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3};\\
&\mathcal{D} = (\mathcal{G}_{\text{pub}}, P, P_3),\\
&\mathcal{T}_1 \xleftarrow{\text{U}} \mathbb{G}_{p_1}, \quad \mathcal{T}_2 \xleftarrow{\text{U}} \mathbb{G}_{p_1p_2}.
\end{aligned}$$

Assumption DSG2

$$\begin{aligned}
&P, X \xleftarrow{\text{U}} \mathbb{G}_{p_1};\ P_2, X_2 \xleftarrow{\text{U}} \mathbb{G}_{p_2};\ P_3, X_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3};\\
&\mathcal{D} = (\mathcal{G}_{\text{pub}}, P, P_3, X + P_2, X_2 + X_3),\\
&\mathcal{T}_1 \xleftarrow{\text{U}} \mathbb{G}_{p_1p_3}, \quad \mathcal{T}_2 \xleftarrow{\text{U}} \mathbb{G}.
\end{aligned}$$

Assumption SGDH

$$\begin{aligned}
&\alpha, s \xleftarrow{\text{U}} \mathbb{Z}_N;\ P \xleftarrow{\text{U}} \mathbb{G}_{p_1};\ P_2, X_2, Y_2 \xleftarrow{\text{U}} \mathbb{G}_{p_2};\ P_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3};\\
&\mathcal{D} = (\mathcal{G}_{\text{pub}}, P, P_2, P_3, \alpha P + X_2, sP + Y_2),\\
&\mathcal{T}_1 = e(P, P)^{\alpha s}, \quad \mathcal{T}_2 \xleftarrow{\text{U}} \mathbb{G}_T.
\end{aligned}$$

2.3.3 Discussion on the SXDH Assumption

For both the DDH and the DLin problems there are no know efficient algorithms to solve these problems in a suitable subgroup of the points on an elliptic curve. The situation changes when we move to pairing groups. For Type-1 pairings, i.e., in the case $\mathbb{G}_1 = \mathbb{G}_2$, DDH becomes easy to solve, whereas DLin is still conjectured to be hard. On the other hand, for Type-3 pairings, the situation is different. As mentioned earlier, for such pairings, there are no known efficiently computable isomorphisms from $\mathbb{G}_1$ to $\mathbb{G}_2$ or from $\mathbb{G}_2$ to $\mathbb{G}_1$. A consequence of this is that the easy algorithm for solving DDH in Type-1 pairings no longer applies and for Type-3 pairings there are no known efficient algorithms to solve DDH1 or DDH2.

The DDH1 (resp. DDH2) problem would become easy if one were able to find an efficiently computable isomorphism from $\mathbb{G}_1$ to $\mathbb{G}_2$ (resp. $\mathbb{G}_2$ to $\mathbb{G}_1$). The non-existence of such isomorphisms is an underlying assumption required for the SXDH assumption to hold. Presently, the isomorphism problem has perhaps not been studied in detail and so, considering SXDH to be a standard assumption may not be universally accepted. On the other hand, we do mention that there is evidence [156, 77] that SXDH is indeed hard. Further, starting from Waters' remarks about the possible efficiency improvements of his dual-system IBE using the SXDH assumption, several schemes have been proposed whose security relies on this assumption [52, 53, 103]. In view of the above discussion, we consider SXDH to be a 'natural' assumption which has been used earlier and has some evidence to support the assumption. In the current state of knowledge, there is no evidence to suggest that for Type-3 pairings, SXDH problem is easier than DLin.

2.3.4 Generic Security of DDH2v

The generic group model is an idealised model introduced by [148] in which lower bounds on computational complexity of solving certain problems can be obtained without looking into the structure of the actual groups that are used in a protocol. Let $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$ be a Type-3 bilinear group. The elements of groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are encoded as uniform random strings so that the adversary can only test for equality of group elements. Four oracles are provided to the adversary out of which three simulate the group actions in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ and the fourth one simulates the bilinear map e. The group encodings are modelled as three injective maps $\sigma_1 : \mathbb{Z}_p \to \Sigma_1$, $\sigma_2 : \mathbb{Z}_p \to \Sigma_2$ and $\sigma_T : \mathbb{Z}_p \to \Sigma_T$ where $\mathbb{E}_1, \mathbb{E}_2, \mathbb{E}_T \subset \{0,1\}^*$. The following theorem provides an upper bound on the advantage of an adversary that solves the DDH2v problem in a generic bilinear group.

Theorem 2.3.1. *Let $\mathscr{A}$ be an algorithm that attempts to solve the* DDH2v *problem in the generic group model making at most m queries to the oracles computing the group actions in $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ and the bilinear map e. If $d, z, x_1, x_2, y \xleftarrow{\text{U}} \mathbb{Z}_p$, $b \xleftarrow{\text{U}} \{0,1\}$ with $y_b = x_1 x_2$ and $y_{1-b} = y$ and $\sigma_1, \sigma_2, \sigma_T$ are random encodings, then given $p, \sigma_1(1), \sigma_1(d), \sigma_1(dz), \sigma_1(zx_1), \sigma_2(1), \sigma_2(d), \sigma_2(x_1), \sigma_2(x_2), \sigma_2(y_0), \sigma_2(y_1)$ the advantage ε of $\mathscr{A}$ in solving the problem is bounded above by*

$$\varepsilon \le \frac{3(m+10)^2}{2p}.$$

Proof. Let $\mathscr{B}$ denote an algorithm that simulates the generic bilinear group for $\mathscr{A}$. $\mathscr{B}$ maintains three lists

$$L_1 = \{(F_{1,i}, \sigma_{1,i}) : i = 0, 1, \ldots, \delta_1 - 1\},$$

$$L_2 = \{(F_{2,i}, \sigma_{2,i}) : i = 0, 1, \ldots, \delta_2 - 1\},$$

$$L_T = \{(F_{T,i}, \sigma_{T,i}) : i = 0, 1, \ldots, \delta_T - 1\}$$

such that at each step δ of the game the relation $\delta_1 + \delta_2 + \delta_T = \delta + 10$ holds. Here $F_{\star,\star}$'s are multivariate polynomials over 6 variables d, z, x_1, x_2, y_0, y_1 and $\sigma_{\star,\star}$'s are strings from $\{0,1\}^*$. At the beginning of the game i.e., $\delta = 0$, the lists are initialised by setting $\delta_1 = 4$, $\delta_2 = 6$ and $\delta_T = 0$. The polynomials $1, d, dz, zx_1$ are assigned to $F_{1,0}, F_{1,1}, F_{1,2}, F_{1,3}$ and $1, d, x_1, x_2, y_0, y_1$ to $F_{2,0}, F_{2,1}, F_{2,2}, F_{2,3}, F_{2,4}, F_{2,5}$ respectively. The encodings for these polynomials are strings uniformly chosen from $\{0,1\}^*$ without repetition. We assume that $\mathscr{A}$ queries the oracles on strings previously obtained from $\mathscr{B}$ and the index of a given string $\sigma_{j,i}$ in the list L_j can be obtained easily by $\mathscr{B}$. The oracles are simulated as follows.

Group actions in $\mathbb{G}_1$,$\mathbb{G}_2$ and $\mathbb{G}_T$: Consider the group $\mathbb{G}_1$. $\mathscr{A}$ submits two strings $\sigma_{1,i}$ and $\sigma_{1,j}$ and a selection bit indicating addition or subtraction. $\mathscr{B}$ first computes $F_{1,\delta_1} = F_{1,i} \pm F_{1,j}$. If there exists an index k with $0 \leq k < \delta_1$ such that $F_{1,\delta_1} = F_{1,k}$ then $\mathscr{B}$ sets $\sigma_{1,\delta_1} = \sigma_{1,k}$; otherwise it sets σ_{1,δ_1} to a uniform random string from $\{0,1\}^* \setminus \{\sigma_{1,0}, \ldots, \sigma_{1,\delta_1 - 1}\}$. $\mathscr{B}$ then adds the pair $(F_{1,\delta_1}, \sigma_{1,\delta_1})$ to L_1, returns σ_{1,δ_1} to $\mathscr{A}$ and increments δ_1 by one.

Group actions in $\mathbb{G}_2$ and $\mathbb{G}_T$ are simulated similarly with the selection bit indicating multiplication or division in case of $\mathbb{G}_T$.

Bilinear map: $\mathscr{A}$ submits two operands $\sigma_{1,i}, \sigma_{2,j}$. $\mathscr{B}$ fetches the corresponding polynomials $F_{1,i}, F_{2,j}$ and computes $F_{T,\delta_T} = F_{1,i} \cdot F_{2,j}$. If for some k with $0 \leq k < \delta_T$, $F_{T,\delta_T} = F_{T,k}$ then set $\sigma_{T,\delta_T} = \sigma_{T,k}$; otherwise σ_{T,δ_T} is set to a random string chosen uniformly from $\{0,1\}^* \setminus \{\sigma_{T,0}, \ldots, \sigma_{T,\delta_T - 1}\}$. $\mathscr{B}$ then adds the pair $(F_{T,\delta_T}, \sigma_{T,\delta_1})$ to L_1, returns σ_{T,δ_T} to $\mathscr{A}$ and increments δ_T by one.

$\mathscr{A}$ makes at most m oracle queries, terminates and returns a bit b' to the simulator. Let $\vec{v} = (d, z, x_1, x_2.y_0, y_1)$ denote the vector consisting of variables over which the polynomials are defined. Now the simulator chooses at random $d^*, z^*, {x_1}^*, {x_2}^*, y^* \in \mathbb{Z}_p$ and $b \in \{0,1\}$ and sets ${y_b}^* = {x_1}^*{x_2}^*$, ${y_{1-b}}^* = y^*$. Let $\vec{v}^* = (d^*, z^*, {x_1}^*, {x_2}^*, {y_0}^*, {y_1}^*)$. $\mathscr{B}$ assigns $\vec{v}^*$ to the variables $\vec{v}$. The simulation provided by $\mathscr{B}$ is perfect unless this assignment causes any of the following to hold.

1. $F_{1,i} - F_{1,j} = 0$ for some $i \neq j$ and $F_{1,i} \neq F_{1,j}$.
2. $F_{2,i} - F_{2,j} = 0$ for some $i \neq j$ and $F_{2,i} \neq F_{2,j}$.
3. $F_{T,i} - F_{T,j} = 0$ for some $i \neq j$ and $F_{T,i} \neq F_{T,j}$.

Let F denote the event that at least one of the above holds, indicating failure. The following result by Schwartz [143] will be used in arguing that the event failure occurs with low probability. Let p be a prime number and $F(Z_1, \ldots, Z_k)$ be a non-zero polynomial in $\mathbb{Z}_p[Z_1, \ldots, Z_k]$ of degree d. Then, if $z_1, \ldots, z_k$ are uniform elements of $\mathbb{Z}_p$, the probability that $F(z_1, \ldots, z_k) = 0$ is at most d/p.

First, we show that if F does not occur, then the bit b is information theoretically hidden from the adversary. Observe that all variables except y_b and y_{1-b} are independent of the bit b. Since $y_b = x_1x_2$, a polynomial of degree 2, the adversary will win if it somehow produces a polynomial consisting of x_1x_2 in L_T using combinations of polynomials from L_1 and L_2 (through applications of the bilinear map) and generate the same polynomial by combining y_b and elements of L_1. Polynomials in L_T are of degree at most 3. The degree two polynomials that can be constructed are $d^2, dx_1, dx_2, dz, zx_1$ and sums of these, but none of them contain x_1x_2. The only degree 3 polynomial consisting of x_1x_2 is zx_1x_2 which can be constructed using $\sigma_1(zx_1)$ and $\sigma_2(x_2)$. However, to find out the bit b, $\mathscr{A}$ will need to engineer the same polynomial using y_b for which it needs $\sigma_1(z)$, but this element is not available in the instance. Therefore, we have $\Pr[b = b' | \neg\mathsf{F}] = 1/2$.

We now only need to obtain a bound on the probability that F occurs. For fixed i and j, $F_{1,i} - F_{1,j}$ is a polynomial of degree at most 2 and hence is zero at a random $\vec{v}^*$ with probability at most $2/p$. Similarly $F_{2,i} - F_{2,j}$ vanishes at $\vec{v}^*$ with probability at most $1/p$. The list L_3 consists of polynomials of degree at most 3 which implies that the third case holds with probability at most

$3/p$. There are totally $\binom{\delta_1}{2}$, $\binom{\delta_2}{2}$, $\binom{\delta_T}{2}$ pairs of polynomials from L_1,L_2, L_T respectively. Also since there are at most m queries we have $\delta_1 + \delta_2 + \delta_T = \delta + 10 \leq m + 10$. It now follows that

$$\begin{aligned}\Pr[\mathsf{F}] &\leq \binom{\delta_1}{2}\frac{2}{p} + \binom{\delta_2}{2}\frac{1}{p} + \binom{\delta_T}{2}\frac{3}{p} \\ &\leq \frac{3(m+10)^2}{p}.\end{aligned}$$

We have

$$\begin{aligned}\Pr[b = b'] &= \Pr[b = b'|\neg\mathsf{F}]\Pr[\neg\mathsf{F}] + \Pr[b = b'|\mathsf{F}]\Pr[\mathsf{F}] \\ &\leq \Pr[b = b'|\neg\mathsf{F}](1 - \Pr[\mathsf{F}]) + \Pr[\mathsf{F}] \\ &\leq \frac{1}{2} + \frac{1}{2}\Pr[\mathsf{F}]\end{aligned}$$

and

$$\Pr[b = b'] \geq \Pr[b = b'|\neg\mathsf{F}](1 - \Pr[\mathsf{F}]) = \frac{1}{2} - \frac{1}{2}\Pr[\mathsf{F}]$$

together resulting in the required bound on the advantage as follows.

$$\left|\Pr[b = b'] - \frac{1}{2}\right| \leq \frac{\Pr[\mathsf{F}]}{2} \leq \frac{3(m+10)^2}{2p}.$$

□

Chapter 3

Evolution of IBE Proof Techniques

Identity-Based Encryption was first formalised by Boneh and Franklin [25]. They also defined a security model for IBE (discussed in Section 2.2.1.1) and proved security of their construction in this model. The proof relied on the DBDH assumption. This work stimulated further research on IBE and a huge body of work emerged targeted at obtaining fully secure and efficient IBE schemes. Arguably, the strongest notion of security for IBE schemes is security in the adaptive identity model without random oracles. It would not be inappropriate to say that a significant amount of research on IBE following [25] was driven by the goal of achieving adaptive security under simple assumptions.

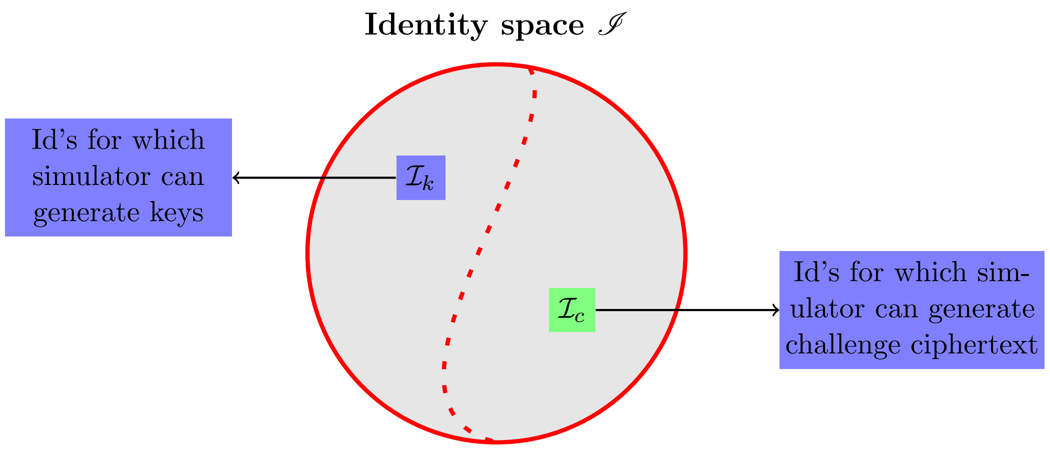
3.1 Early Attempts at Achieving Full Security

The most tricky part in proving security of an IBE scheme is the generation of keys in response to adaptive key extraction queries while being able to provide the adversary a ciphertext on the target identity. It must be ensured that the simulator cannot

- answer key extraction queries on the challenge identity,
- create a secret key for the target identity.

Random Oracles. Initial constructions of IBE [25, 59] were proved secure using *random oracles* (ROs). Random oracle model (ROM) is an idealised model where some or all hash functions used in a system are modelled as (black-box) random functions. In case of Boneh-Franklin IBE the hash function that maps identities to a pairing group is modelled as a random function during simulation. Proof follows a *partitioning* strategy wherein the identity space $\mathscr{I}$ is divided into two subsets – one containing identities for which the simulator can create secret keys and the other containing identities for which challenge ciphertexts can be generated. For systems within the ROM, the partitioning is done at random (i.e., the subset of identities is chosen at random) during simulation.

Random oracles allows security proofs for systems that may not have a proof otherwise. Also, using ROs may lead to more efficient constructions. As a matter of fact, Boneh-Franklin IBE is still one of the most efficient systems known till date. However, a limitation of the ROM is that a truly random function/oracle cannot be implemented by a poly-time algorithm. In addition, there have been some schemes [39, 14] known to be secure in the ROM but insecure when a concrete function is used in place of the RO. Schemes which show such separation may be artificial or the insecurity may arise from certain implementation or "structural" flaws (as pointed out in [18]). Nevertheless, these results place random oracles on shaky ground. So an important question that arises is whether there exist systems that can be proved secure without random oracles?



Identity space $\mathscr{I}$
Id's for which simulator can generate keys
$\mathcal{I}_k$
$\mathcal{I}_c$
Id's for which simulator can generate challenge ciphertext

Figure 3.1: The Partitioning Technique

Towards Security without Random Oracles – Selective Model. The problem of constructing IBE schemes without ROs was first addressed by Canetti, Halevi and Katz [41]. For proving security of their proposed construction, they had to stick to a weaker notion of security, called the selective identity model (refer to Section 2.2.1.1 for details). In this model, the attacker commits to the target identity $\widehat{\mathsf{id}}$ before seeing the public parameters. $\widehat{\mathsf{id}}$ is then used by the simulator to create the public parameters in such a way that keys can be created for identities other than $\widehat{\mathsf{id}}$ and challenge ciphertext can be created for $\widehat{\mathsf{id}}$. In other words, the selective model allows a tight partition of the identity space. Two more IBE constructions in the selective identity model were proposed by Boneh and Boyen [22]. We refer to the as BB-IBE-1 and BB-IBE-2. The schemes were efficient and the algebraic techniques introduced in their works have greatly influenced later works on (H)IBE. The first scheme uses the following identity hash: $H(\mathsf{id}) = U + \mathsf{id}V$ where $U, V \in \mathbb{G}$ (here $\mathbb{G}$ is the source group of a pairing $(p, \mathbb{G}, \mathbb{G}_T, e, P)$) and the identity is mapped to $\mathsf{id} \in \mathbb{Z}_p$ using a collision resistant hash function. This method of hashing the identity has since been used in a number of different works.

Adaptive Security. The first adaptively secure IBE without random oracles was proposed by Boneh and Boyen [23] via a modification of their selectively secure scheme BB-IBE-1. However, the scheme is too inefficient to be used in practice. The problem of constructing a practical adaptively secure IBE without ROs remained open until Waters [157] gave an elegant solution to this problem. Waters used a new identity hash defined as follows. Identities are mapped to

$\{0,1\}^n$ via a collision resistant hash function. $H(\mathsf{id}) = U_0 + \sum_{j=1}^{n} i_j U_j$ where $\mathsf{id} = (i_1, \ldots, i_n) \in \{0,1\}^n$ and $U_0, U_1, \ldots, U_n \in \mathbb{G}$. The elements $U_0, \ldots, U_n$ must be provided in the public parameters to enable creating the hash during encryption. This makes the size of the public parameters rather large. But some trade-offs were obtained by Chatterjee-Sarkar [47] and Naccache [116] between the size of the public parameters and the security loss.

The proof of security of Waters's IBE follows a partitioning strategy. The partition is created during the setup phase and hard-wired into the public parameters. The proof uses an "artificial abort" step which introduces a significant security loss. Bellare and Ristenpart [17] showed how to eliminate this step and obtained a different proof of security for Waters's IBE, also following the partitioning approach.

Limitations of the Partitioning Approach. The partitioning strategy introduces a high security degradation, especially when used for proving security of schemes with richer structure such as HIBE, ABE, etc. This holds true even in the random oracle model. In case of HIBE, the loss in security would be exponential in the maximum depth of the hierarchy h. This makes the reduction less meaningful for large values of h. The basic problem is that HIBE has more structure on the identity space. In addition, delegation in HIBE requires that if an identity vector falls within the key generation partition then all its descendants must also belong to that partition. Taking care of these constraints makes the partitioning reduction very inefficient.

Adaptive Security without Partitioning. In order to avoid the problems introduced by the partitioning paradigm, Gentry [83] proposed an efficient IBE scheme with a tight reduction to an assumption parameterised by the maximum number of key extraction queries q (called q-Augmented Bilinear Diffie-Hellman Exponent). The simulator in the reduction algorithm can generate exactly one secret key for each identity. For an attacker that makes at most q key extraction queries, a degree-q polynomial $f(\mathsf{id})$ is embedded in the secret key corresponding to the id. The challenge ciphertext for $\widehat{\mathsf{id}}$ is constructed in way that it decrypts only with the single key for $\widehat{\mathsf{id}}$ that the simulator can generate. But this provides no information about the distribution of the message to the simulator. These techniques were then extended by Gentry and Halevi [84] to HIBE. They proposed the first adaptively secure HIBE scheme without the exponential security degradation present in earlier HIBE systems. The drawback was that the underlying hardness assumption is more complex. A tight security reduction to an assumption parameterised q is not known to be any better than a loose reduction (with degradation q) to a static assumption.

3.2 Dual System Encryption

Dual system encryption was introduced by Waters in [158] to tackle the innate challenges involved in proving security of identity-based systems in the adaptive identity model. As mentioned in the previous section, a simulator must answer key extraction queries made by the adversary and at the same time be able to use the adversary's success to solve some hard problem. Moreover, in the adaptive identity setting, the simulator must be prepared to produce a secret key for any identity even before looking at the challenge identity. Dual system encryption precisely addresses this problem and prepares the simulator to respond to adaptive key requests in addition to creating the challenge ciphertext. We now define the dual system paradigm.

3.2.1 Semi-functional Ciphertexts and Keys

In a dual system, there are two forms of ciphertexts and keys – normal and semi-functional. The normal form corresponds to their definitions in the actual construction and semi-functional forms are used only in the security proof. A normal ciphertext can be decrypted by a normal or semi-functional key and a normal key can decrypt a normal or semi-functional ciphertext. But decryption of a semi-functional ciphertext by a semi-functional key fails with high probability. Such a decryption causes the message to be blinded by an additional factor determined by the semi-functional components of the key and ciphertext. A proof of security in the dual system framework is usually a hybrid argument over a sequence of games in which the challenge ciphertext and all the keys provided to the adversary are turned into semi-functional. At this point, proving security (in the sense of indistinguishability of ciphertexts) becomes easy as the simulator no longer needs to give out real keys or ciphertext. In other words, none of the keys returned to the adversary provide any information which could possibly assist in decrypting the challenge ciphertext.

In case of Waters's IBE and many other systems, the number of games in the hybrid is $q+3$ where q the number of key extract queries. The first game G_{real} is the real $\mathsf{ind\text{-}cpa}$ game and in the second game (G_0), the challenge ciphertext is semi-functional. In G_k (for $k = 1, \ldots, q$), the challenge ciphertext is semi-functional, the first k keys are semi-functional and the rest of the keys are normal. All keys and challenge ciphertext are semi-functional in G_q. At this stage, the burden on the simulator is greatly reduced as it no longer needs to create any normal keys. The last game is G_{final} where the challenge ciphertext is a semi-functional encryption of a random message. In the proof, one needs to argue that the subsequent games are indistinguishable either computationally or information theoretically. There are essentially three stages in the proof which we term as the first, second and third reductions.

First Reduction: Here we show that the adversary cannot distinguish between a normal and a semi-functional ciphertext so that G_{real} and G_0 are indistinguishable.

Second Reduction: where is it is argued that normal and semi-functional keys are indistinguishable in the adversary's view or equivalently G_{k-1} and G_k are indistinguishable (for $k \in [1, q]$).

Third Reduction: to show that G_q and G_{final} are indistinguishable.

The most crucial step in the dual system proof is the second reduction where it is required to argue that the adversary cannot detect the change of the k-th key (say for id_k) from normal to semi-functional. The simulator embeds the instance of some hard problem in the k-th key so that the attacker's success can be used to solve the problem instance. The simulator could itself create a semi-functional ciphertext for id_k, test whether it decrypts with the k-the key and find out the nature of the k-the key. This must be disallowed. On the other hand, the simulator should be able to create a semi-functional ciphertext for the challenge identity which requires the simulator to have the power to create semi-functional ciphertexts! To deal with this paradox, Waters adopted the following strategy: a single-degree polynomial $F(x) = Ax + B$ is embedded in both the challenge ciphertext and the k-th key where A, B are chosen during setup and information theoretically hidden in the public parameters. The scheme uses random tags in ciphertexts and keys so that decryption succeeds only when the tags are unequal; decryption fails unconditionally otherwise. During simulation the tags in $\mathcal{SK}_{\mathsf{id}_k}$ and $\widehat{\mathcal{C}}$ are set to $F(x)$ evaluated at id_k and $\widehat{\mathsf{id}}$ respectively (i.e., the tags are set to $F(\mathsf{id}_k)$ and $F(\widehat{\mathsf{id}})$ respectively). Since A, B are statistically hidden from the adversary, the tags will appear uniformly and independently distributed in the attacker's view. If the simulator attempts to create a semi-functional ciphertext for id_k, its tag is set to $F(\mathsf{id}_k)$ thus ensuring that encryption with $\mathcal{SK}_{\mathsf{id}_k}$ fails unconditionally. The simulator thus gains no information about the nature of $\mathcal{SK}_{\mathsf{id}_k}$.

3.2.2 Nominal Semi-functionality

Lewko and Waters [114] introduced a new way to tackle the issue discussed above. In contrast to Waters' strategy, the functionality of the public random tags was transferred to the semi-functional space. Instead of allowing the decryption algorithm to fail when tags are equal, the Lewko-Waters method allows decryption to succeed in the presence of a different form of semi-functionality, called *nominal semi-functionality.* A pair of ciphertext and key for a particular identity are nominally semi-functional if they are distributed as semi-functional objects and also correlated to each other in a way that the semi-functional components cancel out and lead to successful decryption. Using nominal semi-functionality, the problem encountered in the second reduction is dealt with in a more elegant manner by designing the simulator as follows. Consider the transition from G_{k-1} to G_k where the secret key $\mathcal{SK}_{\mathsf{id}_k}$ for the k-th query id_k is turned semi-functional.The simulator can only create a nominally semi-functional ciphertext for id_k. Using such a ciphertext, the simulator does not get a clue about the nature of $\mathcal{SK}_{\mathsf{id}_k}$ since decryption would succeed anyway.

The Lewko-Waters method helped in getting rid of tags and the complications they introduce. One is the (negligible) probability of decryption failure and the other being the difficulty of extending to HIBE with *constant-sized ciphertexts.*

3.2.3 Structuring Semi-Functional Spaces

Discussed here are some basic principles underlying the design of semi-functional objects. Consider an IBE scheme and a ciphertext-key pair $\mathcal{C}, \mathcal{SK}_{\mathsf{id}}$ for an identity id. *In the following, we use the abbreviation 'sf' for semi-functional.* Unless both are semi-functional, decryption must be successful. Suppose $\mathcal{C}$ is semi-functional and $\mathcal{SK}_{\mathsf{id}}$ is normal. For the decryption to be successful, the sf-components of $\mathcal{C}$ upon interaction with the normal components of $\mathcal{SK}_{\mathsf{id}}$ must cancel out. Similar condition must hold in case $\mathcal{SK}_{\mathsf{id}}$ is semi-functional and $\mathcal{C}$ is normal. (Figure 3.2 shows the required interaction between ciphertexts and keys). In a sense, the sf-spaces for ciphertext and keys are orthogonal to each other. This is usually achieved by mimicking the structure of the normal components in the corresponding semi-functional terms. Some secret randomness (independent of the public parameters) is used to define these sf-terms. The entropy provided by these terms plays an important role in the security proofs. We discuss the randomness-related issues further in Section 3.2.4. Now we discuss sf-spaces are designed in the pairing setting.

Composite-Order Pairings: The orthogonal property of composite order pairings provides a clean way of defining semi-functional spaces. Let $\mathcal{G} = (N = p_1p_2p_3, \mathbb{G}, \mathbb{G}, \mathbb{G}_T, e, P)$ be a composite order pairing. The group $\mathbb{G}_{p_1}$ provides the space for normal ciphertexts and keys.

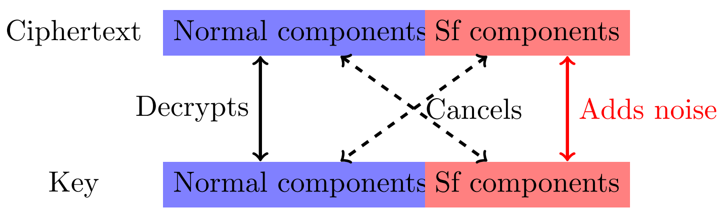


Figure 3.2: Interaction between ciphertexts and keys.

Group $\mathbb{G}_{p_2}$ may be defined as the sf-space. This will ensure that sf-terms cancel out with normal terms (by the orthogonality property). Indistinguishability arguments can be based on the so-called subgroup decision assumptions. In a subgroup decision problem, an element is sampled from one of two subgroups of $\mathbb{G}$ and the task is to decide which group the element was chosen from. Hardness of the problem relies on the fact that N is hard to factor and without the factors there is no trivial way to test membership of the given element.

We need the order to be a product of at least three primes for the following reason. Suppose we take $N = p_1 p_2$. Then to argue that normal and semi-functional ciphertexts are indistinguishable we would need assumption DSG1 (refer to Section 2.3.2.2 for definition), where the subgroups involved are $\mathbb{G}_{p_1}$ and $\mathbb{G}_{p_1 p_2}$. But given another element from either $\mathbb{G}_{p_1}$ or $\mathbb{G}_{p_1 p_2}$, this problem can be solved easily by just testing orthogonality with the challenge element. Hence the group $\mathbb{G}_{p_3}$ is required to additionally randomise either ciphertext or keys.

We use composite-order pairings to design our attribute-based encryption schemes based on deterministic finite automata.

Dual Pairing Vector Spaces: These are elegant mathematical structures that can be instantiated with Type-3 pairings having properties that are found in composite order pairings thus being suitable for constructions within the dual system framework. The first DPVS-based constructions of primitives appeared in [119, 120].

A typical construction for a DPVS $\mathcal{V}_n = (p, \mathbb{V}, \mathbb{V}^*, \mathbb{G}_T, \bar{e}, \mathbb{A}, \mathbb{A}^*)$ is via a direct product over a pairing $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$, where $\mathbb{V} = \underbrace{\mathbb{G}_1 \times \cdots \times \mathbb{G}_1}_{n \text{ times}}$ and $\mathbb{V}^* = \underbrace{\mathbb{G}_2 \times \cdots \times \mathbb{G}_2}_{n \text{ times}}$ are vector spaces of dimension n over $\mathbb{Z}_p$. $\mathbb{A}, \mathbb{A}^*$ form the canonical bases of $\mathbb{V}, \mathbb{V}^*$ respectively. The function $\bar{e}$ is defined as $\bar{e}(\mathbf{x}, \mathbf{y}) = \prod_{i=1}^{n} e(X_i, Y_i)$ where $\mathbf{x} = (X_1, \ldots, X_n) \in \mathbb{V}$ and $\mathbf{y} = (Y_1, \ldots, Y_n) \in \mathbb{V}^*$. One can observe that a form of orthogonality exists between the vectors in $\mathbb{V}$ and $\mathbb{V}^*$. This provides a way to simulate the orthogonality property of composite order groups in the prime order setting. For a DPVS $\mathcal{V}_n$, let $\mathbb{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$ and $\mathbb{B}^* = (\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*)$ be bases of $\mathbb{V}$ and $\mathbb{V}^*$ such that $\bar{e}(\mathbf{b}_i, \mathbf{b}_j^*) = 1$ for $i \neq j$ and $\bar{e}(\mathbf{b}_i, \mathbf{b}_i^*) = e(P_1, P_2)^{\psi}$ for all $i \in \{1, 2, \ldots, n\}$ where ψ is an element of $\mathbb{Z}_p^{\times}$. Then $(\mathbb{B}, \mathbb{B}^*)$ are called *dual bases* of $\mathcal{V}_n$.

Security proofs mostly rely on the *decisional subspace* assumptions in $\mathbb{V}$ (or $\mathbb{V}^*$) wherein a random basis $\mathbb{B}$ (or $\mathbb{B}^*$) is provided in the instance. The task is to decide whether a vector $\mathbf{v} = \sum_{i=1}^{m} x_i \mathbf{b}_i$ for some $1 \leq m < n$ or $\mathbf{v} \xleftarrow{\text{U}} \mathbb{V}$. Of course, if vectors $\mathbf{b}_{m+1}^*, \ldots, \mathbf{b}_n^*$ of the dual basis $\mathbb{B}^*$ of $\mathbb{B}$ are provided, this problem becomes easy to solve - just pair a linear combination of these vectors with $\mathbf{v}$ and check whether the result of the pairing is the identity in $\mathbb{G}_T$.

Details of how features of composite order groups can be obtained in the prime order setting using DPVSs can be found in [111]. Although DPVS-based constructions facilitate more compact dual system proofs and better intuition behind design of ciphertexts and keys, many small details are hidden. For most of these constructions, the dimension n needs to be fixed during system setup. This limits the flexibility in designing ciphertexts and keys in order to optimise efficiency measures. We do not use DPVSs in any of the constructions reported in this thesis.

Prime-Order Bilinear Groups: Most of our constructions are based on Type-3 pairings. We use a more 'ad-hoc' approach to design semi-functional spaces (as first followed by Waters [158]). Typically, a linear relation involving elements from the source groups is embedded in the construction distributed between the public parameters and the master secret. Randomised forms of these elements would then be present in ciphertexts and the keys. During decryption, components containing these elements are paired and cancelled out on obtaining the embedded relation. The sf-space would contain parallel copies of this relation defined via some secret randomness.

In order to ensure that the embedded relation is recovered only upon decryption, more elements may be required in the ciphertext and keys. Due to this reason most dual system constructions obtained via the ad-hoc approach have larger ciphertexts and keys compared to the composite-order constructions. On the other hand, prime-order pairings are faster and have more compact representations. Furthermore, this approach offers more flexibility in designing various objects in the construction to facilitate further efficiency improvements. For these reasons, we mainly follow this approach in our constructions.

3.2.4 Tackling Randomness Issues

In the second reduction, the correlation between challenge ciphertext and the key turned semi-functional is hidden from the attacker by making use of the randomness present in the corresponding semi-functional terms (as mentioned in Sections 3.2.1 and 3.2.2). But this entropy is not enough to deal with more than one key at a time. For this reason, the proof is organised into a hybrid of a sequence of $q + 3$ games so that only one key is dealt with in each game. But in case of primitives with richer structure, the entropy present in sf-terms may not be sufficient to argue about the independence of the ciphertext-key pair. This can be dealt with in either of the following ways – introduce more intermediate types of semi-functionality and devise a way to turn normal objects to semi-functional objects via the intermediate forms. The intermediate forms should be defined in a way that it provides sufficient entropy to argue about independence.

We discuss below the main randomness issues involved in our constructions and the steps taken to resolve them.

Anonymous HIBE: In our HIBE construction $\mathcal{LW}$-$\mathcal{AHIBE}$, a key for an identity vector $\mathbf{id}$ consists of two copies of $\mathcal{H}(\mathbf{id})$ separately randomised. (Here, $\mathcal{H}$ denotes the identity-hash used in the construction.) Semi-functional terms are required to be defined for both components. It is precisely here that the randomness issue arises. Dealing with the entire key at once, along with the challenge ciphertext in the proof, becomes a problem as the entropy in the semi-functional terms is not sufficient. We introduce *partial semi-functional* keys to tackle this issue. The transition from a normal to semi-functional key is now via a partial semi-functional key. As a result the security degradation increases by a factor 2.

DFA-based ABE: Consider a system with Σ as the alphabet. For "small" alphabets, we may associate one group element, say H_σ, in the public parameters to each symbol $\sigma \in \Sigma$ (in contrast to using a hash for "large" alphabets). One can view "small" and "large" to be polynomial and super-polynomial/exponential in the security parameter. These H_σ's appear in both ciphertexts and keys and the components containing them further have sf-components. Since H_σ is fixed during setup, there would be only one way to generate the corresponding sf-terms. For each occurrence of a symbol σ in either a string or an automaton (defining some transition), a copy of sf-terms corresponding to H_σ are revealed. With more than one copy of the sf-terms they no longer remain uncorrelated to other information that an attacker is allowed to obtain.

The solution is to restrict the number of occurrences of symbols in transitions and strings during system setup (first used in [112]). As a result the parameters can be appropriately chosen to accommodate the (fixed) maximum number of occurrences of each symbol. A drawback is that with the restriction only a limited class of regular languages can be supported.

3.3 Recent Developments

Tight Reductions. Chen and Wee [53] proposed a new approach for structuring semi-functional objects as well as organising dual system proofs leading to IBE schemes with a tighter security reduction. The basic idea is to replace Boneh-Boyen hash [22] (that was used in most dual system IBE's) by Waters' hash [157] wherein identities are elements of $\{0,1\}^n$ and the hash for $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_n) \in \{0,1\}^n$ is of the form $\sum_{i=1}^{n} \mathsf{id}_i X_i$. Elements X_i may belong to some pairing group or a dual pairing vector space or more generally to a *dual system group* (also introduced in [53]) designed specially for constructions with proofs in the dual system framework. Then by plugging in the Naor-Reingold pseudorandom function (NR-PRF) [118] taking identities as inputs, sufficient randomness can be generated for handling q (polynomial in the security parameter) key extraction queries over a hybrid of just n games. Consequently, the security degradation is $O(n)$. The drawback in the Chen-Wee IBE scheme was that public parameters contain $O(n)$ group elements. Furthermore, their instantiations for dual system groups were based on composite order pairings and dual pairing vector spaces over prime order pairings leading to a a blow up in the exact size of the public parameters.

A recent work by Blazy, Kiltz and Pan [21] present a method to generically transform message authentication codes (MAC) to (H)IBE. An instantiation of the MAC using NR-PRF leads to an IBE scheme which is more efficient than Chen-Wee scheme while retaining the same security properties. Moreover, they propose the first (almost) tightly secure HIBE scheme i.e., with a security degradation of $O(hn)$ where h is the maximum depth of the hierarchy as opposed to $O(q)$ security loss in previous dual system HIBE constructions.

It is not known whether these techniques are applicable to the broadcast setting or attribute-based encryption. Furthermore, the problem of obtaining an anonymous HIBE scheme with a tight security reduction remains open.

General Framework for Dual System Encryption. Any predicate encryption (PE) scheme is defined by a predicate $\mathcal{P}$. A ciphertext in a PE scheme is associated to an attribute x in addition to the message that it encrypts. Decryption by a secret key associated to attribute y succeeds if and only if $\mathcal{P}(x, y) = 1$. The predicate can be suitably chosen according to the application. Wee[161] proposed an information-theoretic primitive called *predicate encodings* that characterise the underlying algebraic structure of a number of predicate encryption schemes, including known IBE and attribute-based encryption (ABE) schemes. Furthermore these primitives facilitate dual system proofs. In a sense, they provide an abstraction where the functionality achieved is separated from the design and security analysis. A secure PE scheme for predicate $\mathcal{P}$ can be obtained by appropriately instantiating a predicate encoding scheme with $\mathcal{P}$.

In a similar and independent work [8], Attrapadung proposed a generic framework for applying dual system encryption techniques. At the core is a new abstract primitive called *pair encoding scheme* for predicates. As an application of the techniques introduced, some predicate encryption schemes that were known to be only selectively secure have been shown to be adaptively secure.

Chapter 4

A Brief Survey of Identity-Based Cryptography

In this chapter, we provide a brief review of previous and related works on identity-based encryption, hierarchical identity-based encryption and identity-based broadcast encryption. Since most constructions are based on pairings, the survey is broadly divided into two sections – with and without pairings. At the end, a short discussion of other primitives related to IBE is presented. For a more comprehensive treatment of identity-based cryptography, the reader may consult [51] and [102].

4.1 Pairing-Based Schemes

As mentioned in earlier chapters, most practical constructions of identity-based systems are obtained from bilinear pairings. In the following section, we shall take a look at various pairings-based constructions of IBE and related primitives.

4.1.1 Identity-Based Encryption

Initial pairing-based constructions of IBE are by Sakai, Ohgishi, Kasahara [142] and Boneh, Franklin [25]. The latter also provided formal definitions of IBE and IND-ID-CPA and IND-ID-CCA-security along with two constructions, one achieving IND-ID-CPA-security and the other IND-ID-CCA-security in the random oracle model based on the decisional bilinear Diffie-Hellman (DBDH) assumption. A flaw in their security proof was pointed our by Galindo [78] who also presented a corrected version of the proof. A problem that remained open was to construct IBE schemes provably secure without random oracles. Early attempts at building such schemes include the of Canetti, Halevi and Katz [40] which introduced the notion of selective security modelling a weaker security goal. A more efficient selectively secure scheme was proposed by Boneh and Boyen [22]. Evidence [23] suggested the existence of an adaptively secure IBE scheme with a polynomial time security reduction without random oracles but the scheme was too inefficient to be of practical use.

In 2005, Waters [157] presented the first IBE scheme provably secure in the adaptive identity model without random oracles (under DBDH assumption). This scheme is one of the most important and efficient schemes known so far with decryption requiring only 2 pairings in addition to having short ciphertexts and keys. Total Size of the public parameters was $O(n)$ where n represents the length in bits of an identity. Security loss was $O(q)$ where q is the number of key extract queries made by an attacker and the proof required an 'artificial abort' step that leads to weaker concrete security. Bellare and Ristenpart [17] provided an alternate proof of security without the artificial abort. With adaptive security without random oracles being the basic target, few other problems remained open.

1. Are there constructions of IBE with tight security reductions?
2. Is it possible to construct IBE schemes with constant sized public parameters, ciphertexts and keys?

An efficient solution to both problems was first proposed by Gentry [83]. The work proposed a CPA-secure IBE scheme followed by a CCA-secure variant obtained using the techniques of Cramer and Shoup [62]. However, Gentry's scheme was proved secure based on a non-standard assumption parameterised by q. Problem 2 was addressed by Waters [158] who introduced dual system encryption leading to a fundamental change in proof techniques. The resulting IBE had short parameters, ciphertexts and keys with security from the standard and static decisional linear and DBDH assumptions. Security loss, however, was still $O(q)$.

An immediate follow-up work [114] took the route of composite-order pairings. The inherent structure of such pairing groups possibly help in getting a clearer understanding of the technique. (Waters remarks that his scheme [158] was first obtained for composite order groups.) The approach taken by [114] is to look at a realization of the IBE scheme of [22] in the setting of composite order groups so as to obtain adaptive-id security. They also gave a conversion of their composite-order IBE scheme to an IBE scheme using prime-order asymmetric pairing with six elements in both ciphertexts and keys requiring six pairings for decryption. Security was based on 2 static but non-standard assumptions LW1, LW2 (Section 2.3.2.1) and DBDH. Later work by Lewko [111] proposed methods to simulate the features of composite order groups in prime order pairings via dual pairing vector spaces more powerful than earlier techniques of Freeman [73]. Using these methods, Lewko presents a conversion of the composite order pairing-based IBE scheme of [114] into the prime-order setting primarily motivated by efficiency considerations. However the resulting IBE scheme was based on symmetric pairings and consisted of six elements in the ciphertext as well as keys. Decryption required six pairing operations making the scheme rather inefficient. Chen et al. [52] proposed an adaptation of Lewko's technique in the asymmetric prime order pairing setting leading to an IBE construction with 4 group element in ciphertexts/keys with 4 pairings for decryption. The efficiency improvement mainly stems from the fact reductions in asymmetric setting can be based on SXDH as opposed to DLin in the symmetric pairing groups. Further optimisations were not possible as the dual pairing vector space had to be of dimension 4 in order to simulate the properties of composite order pairing groups. A concurrent work by Ramanna, Chatterjee and Sarkar [136] took a different approach – they first converted Waters' IBE [158] to the asymmetric setting and via detailed analysis obtained several simplifications. As a result, they obtained an IBE scheme with 4 group elements plus a tag component (as in Waters' scheme) in ciphertexts/keys with only 3 pairing operations used for decryption. Security was based on DDH1, the static but non-standard

DDH2v (refer to Section 2.3.2.1) and the DBDH assumptions. The most efficient dual system IBE scheme with security based on standard assumptions is due to Jutla and Roy [103]. They take the route of quasi adaptive non-interactive zero knowledge (NIZK) proofs. They obtained short NIZK proofs for linear subspaces leading to efficient signatures from the SXDH assumption. Then using Naor's observation regarding the relationship between IBE and signatures [25], they constructed an IBE scheme with ciphertexts and keys containing 3 and 5 group elements respectively.

An interesting solution to Problem 1 was proposed in a recent work by Chen and Wee [53]. Their main result was an IBE scheme with a security degradation of $O(n)$ where n is the bit-length of identities. Instantiations were provided using both composite order groups and (asymmetric) prime order pairings. The prime order variant is secure under d-linear assumptions. The proof involves a novel use of Naor-Reingold pseudorandom functions (PRFs) [118] to simulate polynomially many key extraction queries with just n instances of d-Lin. However, public parameters in the asymmetric pairing variant contain $O(n)$ many vectors of dimension 2 thus making it inefficient in comparison to the scheme of Jutla and Roy [103]. On the other hand, this work introduced a new method for parameter hiding in dual pairing vector spaces which led to the first efficient DPVS-based HIBE scheme with constant sized ciphertexts. Another work by Blazy et al. [21] presented a generic transform from affine message authentication codes (MACs) to (H)IBE. They presented two instantiations of affine MACs – one from Naor-Reingold PRFs and the other from hash proof systems [61]. The first kind of MACs lead to IBE schemes with a tight security reduction to d-Lin assumptions with shorter ciphertexts and keys compared to [53]. The transform applied on the latter kind of MACs gave rise to efficient IBE schemes with $O(q)$ degradation from d-Lin assumptions. However, for the case $d = 1$ (i.e., SXDH) the scheme is less efficient than that of Jutla and Roy [103]. The authors further defined the notion of delegateable affine MACs that can be transformed to HIBE schemes. The most interesting aspect of this work is the construction of an asymmetric key primitive (HIBE) from a symmetric key primitive (MAC).

CCA-Security. While all the aforementioned works aim primarily at obtaining IND-ID-CPA-secure IBE schemes, an important goal is to obtain efficient IBE schemes secure in the sense of IND-ID-CCA. Boneh and Franklin [25] use the Fujisaki-Okamoto transform [74] to convert their basic IND-ID-CPA-secure scheme to a IND-ID-CCA-secure scheme. This technique, however, applied only in the random oracle setting. An elegant method to obtain CCA-security without ROs was introduced by Canetti, Halevi and Katz [41] (CHK-transform). The core idea was a generic transformation from an $(h+1)$-level HIBE to an h-level HIBE using a strongly unforgeable one-time signature scheme where h is the maximum depth of the hierarchy. Boneh and Katz [30] suggested using MAC in place of the one-time signature for improved efficiency. Another (non-generic) transformation that applies specifically in the pairing based setting is that of Boyen, Mei and Waters [35]. They described a method to convert (pairing-based) IBE schemes with some structure to PKE schemes. But their method can be extended to obtain h-level HIBEs from $(h+1)$-level HIBEs.

Due to the existence of efficient generic methods to obtain CCA-security, little research has been carried out on obtaining direct constructions of IND-ID-CCA-secure IBE schemes. Two recent works by Jutla and Roy [103, 104] propose quasi-adaptive NIZKs based on which they construct *publicly verifiable* CCA-secure IBE schemes. Public verifiability is a notion which requires the well-formedness of the ciphertext to be publicly verifiable. This can be useful in applications where malformed ciphertexts can be filtered at a network level. Earlier generic techniques did not support public verifiability and moreover the CCA-secure scheme of [104] had shorter ciphertexts compared to what is obtained by the CHK-transform [41] applied on the IBE of [103].

4.1.2 Hierarchical Identity-Based Encryption

The concept of hierarchical identity-based encryption was introduced (by Horwitz and Lynn [96]) to reduce the burden of key generation on the PKG by arranging the users into a tree structure rooted at the PKG. An entity with a secret key can generate keys for any lower level entity. HIBE and its security was then formalised by Gentry and Silverberg [86] who also proposed the first HIBE scheme provably secure based on the DBDH assumption using random oracles. As mentioned earlier, the security degraded exponentially with the length of the challenge identity. Canetti, Halevi and Katz [40] obtained the first HIBE whose security did not depend on the use of random oracles. They defined a primitive called binary tree encryption and used it to construct HIBE. However, the trade-off was that the scheme was only secure in the selective identity model. Moreover, the decryption was too inefficient to be of practical use. The first efficient HIBE schemes within the selective model was proposed by Boneh and Boyen [22] with ciphertext length being proportional to the identity depth. Out of the two schemes they propose, security of the first construction is based on the DBDH assumption. The second one is more efficient and is proved secure under the non-standard decisional bilinear Diffie-Hellman inversion (DBDHI) assumption. Boneh, Boyen and Goh [24] considered the problem of constructing HIBEs with constant-sized ciphertexts and provided an elegant solution. The idea was to extend the Boneh-Boyen hash $U + \mathsf{id}V$ for vectors of identities $\mathbf{id} \in \mathscr{I}^{\leq h}$ with length at most h, the maximum depth of the hierarchy. The resulting hash, which we refer to as BBG-hash, is defined as $U + \sum_{j=1}^{\ell} \mathsf{id}_j$ where $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ with $\ell \leq h$. The BBG-hash has proved extremely useful in designing schemes with constant sized ciphertext. Almost all known CC-HIBE schemes that appeared later have either used this hashing technique or a variant. Furthermore, the constant size ciphertexts were particularly useful in cryptographic applications such as forward secure encryption [40] and broadcast encryption [117]. The only problem was that the scheme was selectively secure.

The first HIBE that attained adaptive security without random oracles was the one proposed by Waters [157] as an extension of an efficient IBE scheme. However, the size of the public parameters was too high – $O(hn)$ where n is the length of each individual identity. Later, Chatterjee and Sarkar [45] and Naccache [116] independently showed how to obtain trade-offs between the public parameter size and the security degradation. Another work by Chatterjee and Sarkar [49] combined the BBG technique with Waters' hash and obtained a constant size ciphertext HIBE with adaptive security without random oracles. In all the above schemes, proofs of security relied on the partitioning strategy as a result of which security degraded exponentially with the maximum depth of the HIBE. The degradation was $O(q^h)$ where q is the number of key extraction queries made by the attacker.

Gentry and Halevi [84] presented the first HIBE scheme to achieve adaptive security without an exponential loss in security. Although the reduction was tight, the assumption on which it was based on a very complex parameterised assumption.

The first practical method of constructing HIBE schemes where security does not degrade with the depth of the HIBE is due to Waters [158] who introduced the very important technique of dual-system encryption. Security was based on the decisional linear (DLin) assumptions and DBDH with a security degradation of $O(q)$. Lewko and Waters [114] provided the first CC-HIBE scheme based on composite-order pairings following the dual-system approach. The main disadvantage of this scheme is that it uses composite order pairings. A translation of their HIBE into the asymmetric pairing setting was obtained independently in [137, 107]. Security of both constructions were based on static but non-standard assumptions in Type-3 pairing groups. The most efficient HIBE schemes achieving constant sized ciphertexts were proposed by Ramanna and Sarkar [139] and Chen and Wee [53] with security under the SXDH assumption. While the former has shorter ciphertexts and public parameters, the latter has shorter keys and faster algorithms for key generation and delegation in addition to security from the progressively weaker d-linear assumptions. A recent work by Blazy, Kiltz and Pan [21] proposes a HIBE scheme with a tighter security reduction to d-linear assumptions. More precisely, their reduction incurs a security loss of $O(h\kappa)$ where κ is the security parameter.

Anonymous HIBE schemes provide means to extend PEKS to more sophisticated primitives such as public key encryption with temporary keyword search (PETKS) and identity-based encryption with keyword search (IBEKS). The first construction of anonymous HIBE without random oracles was given by [36] with security in the selective-id model. Later constructions by [144, 63] could achieve security in the adaptive-id setting but were based on composite-order pairings. Two other constructions [70, 129] were based on asymmetric pairings but with security in the selective-id model. The first efficient anonymous CC-HIBE schemes were the ones proposed in [137, 107]. Both the schemes are based on Type-3 pairings and obtain security from non-standard assumptions. The first efficient anonymous CC-HIBE with security from standard assumptions (SXDH) was obtained in [139]. The work [21] also proposes an anonymous HIBE with adaptive security with $O(q)$ degradation from the d-Linear assumptions.

HIBE schemes also be derived as special cases of attribute based encryption (ABE) schemes and predicate encryption (PE) schemes with delegation capabilities [120, 146]. Delegatable PE schemes with linear-sized ciphertexts include the delegateable hidden vector encryption scheme of Shi and Waters [146] based composite order pairings and hierarchical inner product encryption (HIPE) schemes proposed in [120, 112, 121, 122, 124] based on dual pairing vector spaces. Some of these schemes achieve constant-sized ciphertexts and prefix decryption while others have prefix decryption and anonymity.

4.1.3 Identity-Based Broadcast Encryption

The notion of broadcast encryption was introduced by Fiat and Naor in [72]. They describe a symmetric key scheme that achieves bounded collusion resistance. The first fully collusion secure BE (for stateless receivers) was proposed by Naor, Naor and Lotspiech [117]. They describe two symmetric key based BE constructions. Dodis and Fazio [68] used techniques from (hierarchical) identity-based encryption to instantiate the subset cover framework thereby leading to the first fully collusion resistant public key broadcast encryption (PKBE) schemes. The ciphertext size in their constructions is linear in the number of privileged users.

Boneh, Gentry and Waters [28] proposed the first PKBE system achieving constant size ciphertexts. The scheme can be proved secure without random oracles but in the weaker selective model. Delerablee, Paillier and Pointcheval introduced dynamic broadcast encryption in [65] and proposed two (partially) adaptively secure constructions. In dynamic BE schemes, a (new) user can join at any point of time. The confidentiality of a broadcasted message prior to joining of the new user must not be compromised after the join. The join operation requires that the sender is made aware of the public key corresponding to the new user.

The first adaptively secure schemes were proposed by Gentry and Waters [87] in both the public key and identity based settings. They describe two kinds of schemes – one achieving security without random oracles with ciphertext size linear in the number of privileged users and the other consisting of constant size ciphertexts with security relying on the use of random oracles. More recently, the case of adaptive CCA-security was considered in [132, 131]. The construction proposed in the later work has the constant-size ciphertext feature while the former allows users to join the system dynamically.

All the schemes mentioned so far are secure under some non-standard and parametrised assumptions. The first BE scheme secure proved secure under static assumptions was proposed by Waters [158] using the dual system encryption method. The scheme has constant size ciphertexts but the user key size is linear in the total number of users. A revocation system with constant sized keys was proposed in [113] with ciphertext size growing linearly in the number of revoked users and security from static assumptions.

The concept of identity-based broadcast encryption (IBBE) was formalised by Barbosa, Farshim [11] and independently by Baek, Safavi-Naini, Susilo [10]. They called it multi-receiver identity-based encryption (MR-IBE). The work [10] described a pairing based construction based on the Boneh-Franklin IBE [25] that could be proved selectively secure in the random oracle model. A key encapsulation scheme for multiple parties obtained by extending the OR-construction of Smart [152] to the identity-based setting was presented in [11]. Security relies on the use of random oracles.

The construction in [128] (a corrected and improved version of [48]) achieves a trade-off between the ciphertext size and the user key size. Ciphertexts are of size $|S|/N$, and user secret keys are of size N where N is a parameter of the protocol (representing the maximum number of identities that the adversary is allowed to corrupt during simulation). This was the first scheme with sub-linear sized ciphertexts.

Abdalla et al. [2] provided a generic construction from "wicked IBE" with constant-sized ciphertexts but user storage quadratic is m, the maximum number of recipients of a ciphertext. Both schemes ([128] and [2]) are selectively secure without random oracles. In 2007, Delerablee [64], proposed an IBBE construction with constant size ciphertexts and secret keys. The public parameters have size $O(m)$. Security was proved in the selective identity model.

Gentry and Waters [87] were the first to propose adaptively secure IBBE systems achieving linear and sub-linear sized ciphertexts. However, their proofs were based on non-standard assumptions parameterised by m.

Note: A basic functionality of any identity-based system is that it is dynamic. The PKG should be able to generate keys for any identity from the identity space; further, an identity-based system should allow for encryption to be possible to an identity even before a key for that identity has been generated. By extension, any proper identity-based broadcast encryption scheme should also be dynamic and this is true of the schemes that we describe.

4.1.4 Attribute-Based Encryption

Attribute-Based Encryption (ABE) systems are the most well studied candidates of functional encryption systems with public index. There is a huge body of work on ABE. Since the main focus of this work is not ABE, we only mention some works on ABE relevant to the problem addressed in Chapter 8.

Fuzzy Identity-Based Encryption, introduced by Sahai and Waters [141], was as an initial step towards attribute-based encryption. Goyal, Pandey, Sahai and Waters [91] introduced two complimentary forms of ABE named Ciphertext-Policy Attribute-Based Encryption (CP-ABE) and Key-Policy Attribute-Based Encryption (KP-ABE). In a CP-ABE system, keys are associated with sets of attributes and ciphertexts are associated with access policies. In a KP-ABE system, the situation is reversed: keys are associated with access policies and ciphertexts are associated with sets of attributes. Subsequent constructions of ABE schemes focused on one of the following goals - efficiency gains, stronger security guarantee, functional capabilities, security in standard model.

Then followed several constructions of key-policy ABE (KP-ABE) systems [126, 112, 121] and ciphertext-policy ABE (CP-ABE) systems [19, 112, 121, 159, 110]. ABE schemes in the public index model include [91, 126, 159, 110] and the works [105, 147, 120, 123] consider ABE systems without public index.

The first adaptively secure CP-ABE and KP-ABE schemes for *monotone access structures* without random oracles were proposed by Lewko et al. [112]. This construction was based on composite-order groups. Later, Okamoto and Takashima [121] proposed ABE schemes for non-monotone access structures that are adaptively secure under a the standard DLin assumptions in prime order pairing groups. Both the works obtained adaptive security from static assumptions by imposing a bound on the maximum number of times an attribute can occur in access policies and indices at the cost of losing out on efficiency compared to selectively secure schemes. Later work by Lewko and Waters [110] showed how to remove this restriction. However, their proofs had to rely on non-static assumptions.

All the above mentioned works only considered functions over fixed size inputs. Waters [160] first considered ABE over arbitrary sized inputs and proposed a KP-ABE scheme over regular languages. Here the index is a string and policy is a deterministic finite automaton, both over a common alphabet. Since DFA's accept regular languages that may contain strings of any size, Waters' ABE system can support access control over messages of arbitrary size. The KP-ABE scheme of [160] is selectively secure under the non-static decisional ℓ-expanded bilinear Diffie-Hellman assumption.

4.2 Identity-Based Constructions without Pairings

Number-theoretic constructions. Among the first constructions of IBE, Cocks IBE [59] did not use pairings. The scheme was based on quadratic residues. A drawback was that ciphertexts were rather large due to the bit-by-bit encryption strategy used. Boneh, Gentry and Hamburg [26] proposed an IBE scheme based on quadratic residues that had shorter ciphertexts at the cost of losing out on efficiency in comparison to Cocks IBE. Furthermore, security relied on the use of random oracles. However, this line of research sparked little interest and hence not followed up in later works.

Lattice-based constructions. With the introduction of lattices for cryptanalysis, researchers began to explore the possibilities of their use in building cryptographic primitives. Lattices are discrete subgroups of the n-dimensional Euclidean space $\mathbb{R}_n$. Lattices used in cryptography are mostly defined as subgroups of vector spaces over a finite field. An example of a hard problem over a lattice is the shortest vector problem wherein the lattice is specified by a basis and the task is to find a lattice point closest to the origin or equivalently, the shortest vector in the lattice. Other examples include closest vector problem, learning with errors, and so on. The main motivation for using lattices is three-fold.

- Security can be based on worst-case hardness of lattice problems
- Lattice algorithms have efficient and parallel implementations
- There are no known quantum algorithms (until now) to solve lattice-based hard problems

Initial cryptographic constructions included public key encryption and collision-resistant hash functions. Later, techniques were introduced to realise IBE from lattices. The first IBE was obtained via a notion called pre-image sampling introduced in by Gentry, Peikert and Vaikuntanathan [85]. They started with a signature scheme from pre-image sampling and derived an IBE from it on the basis of Naor's transform. Cash et al. [42] showed how to delegate lattice bases and hence obtain lattice-based hierarchical IBE schemes. Following this work, many improved constructions of (H)IBE were proposed by Agrawal, Boneh and Boyen [4, 5]. While lattices provide theoretically satisfying security, they are far less efficient compared to pairing-based constructions.

Multilinear maps from lattices. The concept of multilinear maps in cryptography was first used by Boneh and Silverberg [33]. They showed the possibility of realising certain primitives more efficiently using such maps and further conjectured that such maps would be hard to find at least in the realm of algebraic geometry. Garg, Gentry and Halevi [79] introduced the notion of *graded encoding schemes* based on ideal lattices, which are approximations of multilinear maps. Soon afterwards, graded encoding schemes found several applications in cryptography including some primitives that could not be instantiated with pairings. An important application presented in [33, 79] is a single round n-way Diffie-Hellman key exchange where n users share a secret key in a single round with each user receiving a message. Further applications include constrained pseudorandom functions [34], broadcast encryption with short parameters [79, 33, 34], attribute encryption schemes for general circuits [81, 90], indistinguishability obfuscation [80] and many more. Multilinear maps have opened up many avenues for cryptographic constructions and some of these constructions subsume the functionalities we try to achieve in this work. However, very little cryptanalysis has been done on them. Furthermore, the implementation issues related to lattices mentioned in the previous paragraph also apply here.

4.3 Other IBE-Related Primitives

A number of primitives apart from HIBE, IBBE and ABE are related to identity-based encryption. Examples include key agreement protocols [142, 151, 56, 130], searchable encryption [1], etc. More abstract constructs includes identity-based (lossy) trapdoor functions [15]. Out of these we touch upon only three topics – signatures and methods to deal with the *key escrow problem*.

Signatures A signature scheme allows a user to sign a message using a secret signing key. The message-signature pair can be checked for validity using a (related) public verification key. Moni Naor observed (in [25]) that any identity-based encryption scheme gives rise to a signature scheme. The basic idea is as follows: the message space for the signature scheme is nothing but the identity space of the IBE. The signature for a message m is the secret key generated with m as the identity. Verification of the signature is done by generating a random ciphertext for identity m and testing whether decryption is successful or not with the key/signature. Verification key and signing key correspond to the public parameters and master secret (respectively) of the IBE scheme.

Naor's strategy has been applied on a number of different IBE schemes. Boneh, Lynn and Shacham [31] signatures were the first of this kind obtained from Boneh, Franklin IBE [25] with security in the ROM under the Gap Diffie-Hellman assumption. This construction was further used in constructing aggregate signature schemes [27]. Waters [157] converted the his IBE scheme to a signature scheme that achieved security under the computational Diffie-Hellman assumption without random oracles. Interestingly, the dual system encryption methodology lead to signatures schemes [158, 52, 103] with security under decisional assumptions as opposed to computational assumptions on which previous constructions were based.

In a traditional (public-key) signature scheme, the verification key of a user is certified by some central authority. To eliminate the need for certification, the concept of identity-based signatures (IBS) was proposed by Shamir [145]. The problem of constructing IBS seems relatively easy compared to constructing IBE schemes. Shamir himself proposed an IBE construction whereas the first IBE construction appeared years later. Similar to IBE, there is a setup and key generation algorithm. In addition there are two algorithms for signing and verification. Further constructions of identity-based signature schemes appeared in [94, 13, 16].

Dealing with Key Escrow All IBE schemes suffer from one common problem. The PKG can generate key for any user and hence decrypt all ciphertexts sent to that user. This is the so-called *key escrow problem*. Among the many solutions proposed for dealing with this problem, the most interesting ones are certificate-less encryption (CLE) and certificate-based encryption (CBE). AlRiyami and Paterson [6] introduced the notion of CLE in which the secret key for an identity is generated based on two quantities – the PKG's master secret and the corresponding user's secret information. The term certificate-less stems from the search for a PKE without the need for certificates and at the same time not have the key escrow problem of IBE. They formalise the security model for CLE and also propose a construction secure in that model within the random oracle model. However, constructing schemes without random oracles remained a difficult task and thus lead to several alternative security models for CLE that are weaker than the original model [6]. Dent [66] provides a summary of these models. Early works [97, 115] trying to obtain CLE without random oracles could not achieve security in the full model. The first scheme achieving security without ROs in the full security model was by [67]. There have been several follow up works on certificateless encryption.

Certificate-based encryption (introduced by Gentry in [82]) is reminiscent of public key encryption but with implicit certification on user public keys using identity-based techniques. Gentry also provided an instantiation of CBE based on Boneh-Franklin IBE. Subsequent work [162] established equivalence results between IBE, CBE and CLE schemes. Their generic transformations of do not rely on random oracles but at the same time do not hold even in the weaker security definitions for CLE, let alone the full security model of AlRiyami and Paterson. Later work [7] showed a generic transformation from CLE to CBE along with improved constructions for CLE and hence CBE.

Chapter 5

Efficient Dual System IBE and Related Primitives

As mentioned in the earlier chapters, dual system encryption introduced by Waters is an important technique for proving adaptive security of identity-based encryption and related primitives. Waters IBE scheme in [158] is based on symmetric pairings. Security of the scheme is based on the DLin and DBDH assumptions. It is of interest to convert this to asymmetric pairings. For one thing, this will enable faster and smaller implementations which will arise from the advantages of asymmetric pairings over their symmetric variants. There is, however, another reason. Use of asymmetric pairings brings forward the possibility of reducing the number of group elements in ciphertexts and keys. In fact, Waters [158] himself mentions: "using the SXDH assumption we might hope to shave off three group elements from both ciphertexts and private keys". The rationale for this comment is that in Type-3 pairings, the DDH assumption holds for both $\mathbb{G}_1$ and $\mathbb{G}_2$ (collectively called the SXDH assumption). Using SXDH will potentially lead to a simpler scheme requiring a smaller number of group elements.

Following up on the above mentioned remark by Waters, we have systematically investigated the various possibilities for using asymmetric pairings. To start the study, we performed a straightforward conversion to the setting of asymmetric pairings. The scheme in [158] is quite complex. Several scalars are used in the public parameters, encryption and key generation. These have definite and inter-connected roles in the security proof. Our first task was to pin down the relationships between these scalars and separate them out. This enabled us to work with one group of scalars with minimal changes to other groups.

With a good understanding of the roles of the scalars, we are able to apply simplifications in a stage-wise manner. The first simplification gives an IBE scheme ($\mathcal{IBE}1$) which shrinks ciphertexts and keys by two elements each and whose security can be based on DDH1 (DDH assumption in $\mathbb{G}_1$), DLin and the DBDH assumptions. We argue that the DDH2 assumption cannot be directly used. So, the afore-mentioned suggestion by Waters cannot be fulfilled. On the other hand, we show that using a natural and minimal extension of the DDH2 assumption, a significantly more efficient scheme ($\mathcal{IBE}6$) can be obtained.

Waters' original scheme [158] used random tags in the ciphertext and the decryption key. Simplification of this scheme by both Lewko-Waters [114] and Lewko [109] yielded IBE schemes which did not use such tags. In contrast, all our simplifications retain the tags used in the original description [158]. Even then, we are able to obtain significant simplifications and efficiency improvements.

$\mathcal{IBE}6$ has the interesting feature that exactly one randomiser each is used for encryption and key generation which is minimal in case of ciphertext. However, it is not known whether the key generation could be made deterministic within the dual system framework.

To show that our simplification retains the flexibility of the original technique by Waters, we present an analogue of the HIBE scheme along with a security proof in Section 5.7. This HIBE scheme, denoted $\mathcal{HIBE}6$ inherits all the security properties from [158], but, provides improved efficiency. From this HIBE scheme we construct an adaptively secure BE scheme $\mathcal{BE}6$ with constant-size ciphertexts. The construction is given in Section 5.8.1 and the security proof in Section 5.8.2.

5.1 Framework for Conversion

Our goal is to transform Waters-2009 IBE scheme to the asymmetric setting so that we can reduce the number of components both in the ciphertext and the key. To that end, we first perform a straightforward conversion of Waters IBE from the setting of symmetric pairing to the setting of asymmetric pairing. (See [158] for the original description of Waters 2009 scheme.)

Let $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, F_1, F_2)$ be a Type 3 pairing. Since elements of $\mathbb{G}_1$ have shorter representation compared to $\mathbb{G}_2$, we choose ciphertext elements to be from $\mathbb{G}_1$ and secret key components from $\mathbb{G}_2$. The public parameters will consist of elements of $\mathbb{G}_1$ whereas the master secret key will consist of elements of $\mathbb{G}_2$. We note that if the final goal were to construct a signature scheme, one would choose the secret key from $\mathbb{G}_1$.

A straightforward conversion (named $\mathcal{W}$-$\mathcal{IBE}$) will have the same structure as the one described in [158]. Message space is $\mathbb{G}_T$ and identities are elements of $\mathbb{Z}_p$. The algorithms would be defined as follows.

$\mathcal{W}$-$\mathcal{IBE}$.$\mathsf{Setup}(\kappa)$ Choose $P_1 \xleftarrow{\text{U}} \mathbb{G}_1^\times$, $P_2 \xleftarrow{\text{U}} \mathbb{G}_2^\times$, $\alpha, b, a_1, a_2 \xleftarrow{\text{U}} \mathbb{Z}_p$. Let v, v' and v'' be random elements of $\mathbb{Z}_p$ and define $V_2 = vP_2$, $V_2' = v'P_2$ and $V_2'' = v''P_2$. Let $\tau = v + a_1v'$ and $\tau' = v + a_2v''$. Set $T_1 = \tau P_1$ and $T_1' = \tau' P_1$. Pick elements $Q_1, U_1, W_1 \xleftarrow{\text{U}} \mathbb{G}_1$ and $Q_2, U_2, W_2 \in \mathbb{G}_2$ with $\mathsf{dlog}_{P_2}(Q_2, U_2, W_2) = \mathsf{dlog}_{P_1}(Q_1, U_1, W_1)$. The structure of the $\mathcal{PP}$ and the $\mathcal{MSK}$ are as follows.

$\mathcal{PP}$: $(P_1, bP_1, a_1P_1, a_2P_1, ba_1P_1, ba_2P_1, T_1, T_1', bT_1, bT_1', Q_1, W_1, U_1, e(P_1, P_2)^{ba_1\alpha})$.
$\mathcal{MSK}$: $(P_2, \alpha P_2, a_1\alpha P_2, V_2, V_2', V_2'', Q_2, W_2, U_2)$.

$\mathcal{W}$-$\mathcal{IBE}$.$\mathsf{Encrypt}(M, \mathsf{id}, \mathcal{PP})$: Randomisers $s_1, s_2, t, \mathsf{ctag} \xleftarrow{\text{U}} \mathbb{Z}_p$ are chosen. Define $s = s_1 + s_2$. The ciphertext is $(C_0, C_1, \ldots, C_7, E_1, E_2, \mathsf{ctag})$ with the components defined as follows.

$C_0 = M \cdot e(P_1, P_2)^{ba_1\alpha s_2}$
$C_1 = bsP_1$, $C_2 = ba_1s_1P_1$, $C_3 = a_1s_1P_1$, $C_4 = ba_2s_2P_1$,
$C_5 = a_2s_2P_1, C_6 = s_1T_1 + s_2T_1'$, $C_7 = s_1bT_1 + s_2bT_1' - tW_1$
$E_1 = t(\mathsf{id}Q_1 + \mathsf{ctag}W_1 + U_1)$, $E_2 = tP_1$

$\mathcal{W}$-$\mathcal{IBE}$.$\mathsf{KeyGen}(\mathsf{id}, \mathcal{MSK}, \mathcal{PP})$: Choose $r_1, r_2, z_1, z_2, \mathsf{ktag} \xleftarrow{\text{U}} \mathbb{Z}_p$ and define $r = r_1 + r_2$. The key $\mathcal{SK}_{\mathsf{id}}$ is $(K_1, \ldots, K_7, \mathsf{ktag})$ where the elements are defined as follows.

$K_1 = a_1\alpha P_2 + rV_2,\ K_2 = -\alpha P_2 + rV_2' + z_1P_2,\ K_3 = -z_1bP_2,\ K_4 = rV_2'' + z_2P_2,$
$K_5 = -z_2bP_2,\ K_6 = r_2bP_2,\ K_7 = r_1P_2$
$D = r_1(\mathsf{id}Q_2 + \mathsf{ktag}W_2 + U_2).$

The decryption algorithm $\mathcal{W}\text{-}\mathcal{IBE}.\mathsf{Decrypt}$ (as described by Waters) requires 9 pairings and succeeds only if ctag in the ciphertext is not equal to ktag of the decryption key, an event which occurs with overwhelming probability. See [158] for the details.

We now look at the definition of semi-functional ciphertext and key for Waters' scheme.

$\mathcal{W}\text{-}\mathcal{IBE}.\mathsf{SFEncrypt}(\mathcal{PP}, \mathcal{MSK}, \mathcal{C}')$: Let $\mathcal{C}' = (C_0', \ldots, C_7', E_1', E_2', \mathsf{ctag})$ be a ciphertext normally generated by the Encrypt algorithm for message M and identity id. Choose $\mu \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. Let $V_1' = v'P_1$ and $V_1'' = v''P_1$ so that $V_1' \sim V_2'$ and $V_1'' \sim V_2''$. The semi-functional ciphertext generation algorithm will modify the normal ciphertext as: $C_0 = C_0'$, $C_1 = C_1'$, $C_2 = C_2'$, $C_3 = C_3'$, $E_1 = E_1'$, $E_2 = E_2'$ and

$$C_4 = C_4' + ba_2\mu P_1,\ C_5 = C_5' + a_2\mu P_1,\ C_6 = C_6' - a_2\mu V_1'',\ C_7 = C_7' - ba_2\mu V_1''.$$

$\mathcal{W}\text{-}\mathcal{IBE}.\mathsf{SFKeyGen}(\mathcal{PP}, \mathcal{MSK}, \mathcal{SK}_{\mathsf{id}})$: Let $\mathcal{SK}_{\mathsf{id}}' = (K_1', \ldots, K_7', D', \mathsf{ktag})$ be the secret key normally generated by the KeyGen algorithm for identity id. Choose $\gamma \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. The semi-functional key generation algorithm will modify the normal key as: $K_3 = K_3'$, $K_5 = K_5'$, $K_6 = K_6'$, $K_7 = K_7'$, $D = D'$ and

$$K_1 = K_1' - a_1a_2\gamma P_2,\ K_2 = K_2' + a_2\gamma P_2,\ K_4 = K_4' + a_1\gamma P_2.$$

It is easy to see that one can decrypt a semi-functional ciphertext with a normal key and a normal ciphertext with a semi-functional key. However, decryption of a semi-functional ciphertext with a semi-functional key will fail because the message will be blinded by the factor $e(P_1, P_2)^{ba_1a_2\mu\gamma}$.

Security proof. The security argument for the scheme proceeds through $q+3$ games where q is the number of key extraction queries made by the adversary. These games are

$$\mathsf{G}_{real}, \mathsf{G}_0, \ldots, \mathsf{G}_q, \mathsf{G}_{final}.$$

The transition between these games can be seen as three different reductions.

First reduction: The transition from G_{real} to G_0 is made by replacing the challenge ciphertext by a semi-functional ciphertext. It is argued that detecting this change is hard.

Second reduction: There is a sequence of q changes from G_{k-1} to G_k (for $k = 1, \ldots, q$). The k-th change is as follows. For the queries numbered 1 to $k-1$, the adversary is given a semi-functional key; for queries numbered $k+1$ to q, the adversary is given a normal key. The k-th key is changed from normal to semi-functional. It is required to show that the adversary cannot detect this change with non-negligible probability.

Third reduction: This tackles the transition from G_q to G_{final}. At this point, all responses to key queries are semi-functional and so is the challenge ciphertext. In the last transition, the challenge ciphertext is changed so that it encrypts a random message such that deciding whether it is the encryption of a message or whether it is statistically independent of the $\mathcal{PP}$ and the responses is hard.

The first and second reductions are based on the hardness of the DLin problem whereas the third reduction is based on the hardness of the DBDH problem. In the proof, the second reduction is the most complex step. The subtle point is that the simulator should not be able to generate a semi-functional ciphertext for the k-th identity which will allow it to easily determine whether the key for this identity is semi-functional or not. This is ensured by using algebraic techniques from [22] to create ktag using a pair-wise independent function so that the simulator is able to create a semi-functional ciphertext for id_k only with $\mathsf{ctag} = \mathsf{ktag}$, in which case decryption fails unconditionally and hence the simulator gains no information.

5.2 An Analysis

Our conversion to asymmetric pairing and subsequent simplifications are based on an analysis of the various scalars used in the scheme and their respective roles in the proof. Based on the scheme itself and a study of the three reductions used by Waters, we make the following observations.

1. $\mathcal{PP}$ uses the scalars a_1, a_2 and b, while $\mathcal{MSK}$ uses the scalars α and a_1.
2. Key generation uses scalar randomisers r_1, r_2 and z_1, z_2. The scalar r is set to $r_1 + r_2$. We will call this the split of r.
3. Ciphertext generation uses the scalar randomisers s_1, s_2 and t. The scalar s is set to $s_1 + s_2$. We will call this the split of s.
4. The first two reductions in Waters proof are based on the DLin assumption. The first reduction uses the split of s whereas the second reduction uses the split of r.

For simplification using asymmetric pairings, the following points are to be noted. These have been inferred from a study of the security proof in [158].

1. The scalar α needs to be retained.
2. There are three basic possibilities for simplification: remove the split of s; remove the split of r; remove z_1, z_2.
3. Getting rid of a_1 and a_2 and using a single a will eliminate the requirement of the split of s. This also means that the separate z_1 and z_2 are not required and instead a single z can be used.
4. Removing the split of r does not have a direct influence on the other scalars.
5. Removing the split of r and also z_1, z_2 means that the scalar b is no longer required.
6. In all but one of our schemes, the scalar t is kept either as part of the ciphertext or as part of the key. In the final scheme, we show that the scalar t can also be removed. For this scheme, there is a single randomiser s for the ciphertext and a single randomiser r for the key. Note that further reduction in randomness for the ciphertext is not possible as otherwise the ciphertext becomes unique. However, it may be possible to make key generation deterministic.

7. If the first reduction is to be based on DLin, then the split of s and a_1,a_2 must be retained. If the split is removed, then we can base the first reduction on DDH1.
8. If the split of r is retained, then the second reduction has to be based on DLin. If it is removed, we can no longer base the second reduction on DLin. However, it can neither be based on DDH2 for the following reason. An instance of DDH2 will provide P_1 and some elements of $\mathbb{G}_2$. Apart from P_1 no other element of $\mathbb{G}_1$ is provided. The $\mathcal{PP}$ consists of elements of $\mathbb{G}_1$ which have to be related to the instance in some way. Just having P_1 does not provide any way to construct the $\mathcal{PP}$ in the second reduction. So, removing the split of r implies that the second reduction can be based on neither DLin nor DDH2. The assumption DDH2v introduced in Chapter 2 provides the necessary mechanism for carrying the proof through. We provide a discussion on DDH2v in Section 5.5.2.
9. The tags are chosen randomly and they play a crucial role in the security argument. We do not consider removing tags. If the tags are removed, then it will be necessary to introduce copies of the identity-hash (as done in [114]) to obtain the functionality of tags in the semi-functional components. This leads to an increase in the number of elements in the ciphertext and key.

Based on the above points, we explore the different natural ways in which $\mathcal{W}$-$\mathcal{IBE}$ can be simplified. These are discussed below.

IBE1**:** Remove the split of s. This eliminates the requirement of having separate a_1, a_2 and z_1, z_2. Reductions of ciphertext and key are by two elements each. Removing the split of s allows the first reduction to be based on DDH1. Since the split of r is retained, the second reduction is still based on DLin.

IBE2**.** Retain the split of s; this means that separate a_1 and a_2 are required. Remove the split of r and also remove z_1 and z_2; this means that b can be removed. Leads to reductions of ciphertext and key by 3 elements each. The first reduction of the proof can be based on DLin, but, the second reduction cannot be based on either DLin or DDH2.

IBE3**:** Remove the split of s; retain the split of r but, remove z. Reductions of ciphertext and key are by 3 elements each. In the proof, the first reduction can be based on DLin. The second reduction cannot be based on DDH2. Neither can it be based on DLin. This requires a more involved reasoning which we provide in Section 5.4.

IBE4**:** Remove the splits of both r and s, but, retain z. Ciphertext and key are reduced by 3 elements each. In the proof, the first reduction can be based on DDH1, but, the second reduction cannot be based on either DLin or DDH2.

IBE5**:** Remove the splits of both r and s and also remove z. Ciphertext and keys are reduced by 4 elements each. As in the previous case, the first reduction of the proof can be based on DDH1, but, the second reduction cannot be based on either DLin or DDH2.

IBE6**:** In Schemes 1 to 5, the randomiser t is present in the ciphertext. In *IBE6*, the splits of both r and s are removed; z is removed and the role of t is played by s. This leads to a scheme where there is exactly one randomiser for encryption and exactly one randomiser for key generation. Compared to Waters' IBE [158], reduction of the ciphertext is by 5 elements and the reduction of the key is by 4 elements. The first reduction of the proof can be based on DDH1, while the second reduction is based on assumption DDH2v.

In Table 5.1, we provide the use of scalars in the various schemes. This illustrates the manner in which the simplification has been obtained.

scheme	$\mathcal{PP}$	$\mathcal{MSK}$	key gen	enc
Waters-09 [158]	α, a_1, a_2, b	α, a_1	$r_1, r_2, (r = r_1 + r_2), z_1, z_2$	$s_1, s_2, (s = s_1 + s_2), t$
$\mathcal{IBE}1$	α, a, b	α, b	$r_1, r_2, (r = r_1 + r_2), z$	s, t
Scheme 2	α, a_1, a_2	α	r	$s_1, s_2, (s = s_1 + s_2), t$
Scheme 3	α, a, b	α, b	$r_1, r_2, (r = r_1 + r_2)$	s, t
Scheme 4	α, a, b	α, b	r, z	s, t
Scheme 5	α, a	α	r	s, t
$\mathcal{IBE}6$	α, a	α	r	s

Table 5.1: Usage of scalars in various schemes. Note that all the schemes use ktag for key generation and ctag for encryption.

5.3 Scheme $\mathcal{IBE}1$

We describe here our first construction $\mathcal{IBE}1$ and its proof of security.

5.3.1 Construction

$\mathcal{IBE}1.\mathsf{Setup}(\kappa)$: Choose $P_1 \xleftarrow{U} \mathbb{G}_1^{\times}$ and $P_2 \xleftarrow{U} \mathbb{G}_2^{\times}$; elements $Q_1, W_1, U_1 \xleftarrow{U} \mathbb{G}_1$ and $Q_2, W_2, U_2 \in \mathbb{G}_2$ such that $\mathsf{dlog}_{P_2}(Q_2, U_2, W_2) = \mathsf{dlog}_{P_1}(Q_1, U_1, W_1)$. Let v, v' be chosen randomly from $\mathbb{Z}_p$ and set $V_2 = vP_2$, $V_2' = v'P_2$. Pick α, a, b at random from $\mathbb{Z}_p$. Set $\tau = v + av'$ so that $\tau P_2 = V_2 + aV_2'$.

$\mathcal{PP}$: $(P_1, aP_1, bP_1, abP_1, \tau P_1, b\tau P_1, Q_1, W_1, U_1, e(P_1, P_2)^{b\alpha})$.
$\mathcal{MSK}$: $(P_2, \alpha P_2, bP_2, V_2, V_2', Q_2, U_2, W_2)$.

$\mathcal{IBE}1.\mathsf{Encrypt}(M, \mathsf{id}, \mathcal{PP})$: Choose random s, t, ctag from $\mathbb{Z}_p$. Ciphertext $\mathcal{C}$ is $(C_0, C_1, \ldots, C_5, E_1, E_2, \mathsf{ctag})$ where the elements are computed as follows.

$C_0 = M \cdot e(P_1, P_2)^{b\alpha s}$,
$C_1 = bsP_1$, $C_2 = basP_1$, $C_3 = asP_1$, $C_4 = -\tau sP_1$, $C_5 = -b\tau sP_1 + tW_1$,
$E_1 = t(\mathsf{id}Q_1 + \mathsf{ctag}W_1 + U_1)$, $E_2 = tP_1$.

$\mathcal{IBE}1.\mathsf{KeyGen}(\mathsf{id}, \mathcal{MSK}, \mathcal{PP})$: Choose random $r_1, r_2, z, \mathsf{ktag}$ from $\mathbb{Z}_p$ and let $r = r_1 + r_2$. $\mathcal{SK}_{\mathsf{id}}$ is $(K_1, \ldots, K_5, D, \mathsf{ktag})$ where the elements are computed as follows.

$K_1 = \alpha P_2 + rV_2$, $K_2 = rV_2' - zP_2$, $K_3 = bzP_2$, $K_4 = br_2P_2$, $K_5 = r_1P_2$,
$D = r_1(\mathsf{id}Q_2 + \mathsf{ktag}W_2 + U_2)$.

$\mathcal{IBE}1$.Decrypt$(\mathcal{C}, \mathsf{id}, \mathcal{SK}_{\mathsf{id}}, \mathcal{PP})$: Decryption succeeds only when $\mathsf{ctag} \neq \mathsf{ktag}$, an event which occurs with overwhelming probability. Let $\vartheta = (\mathsf{ctag} - \mathsf{ktag})^{-1}$. The masking factor is computed as follows.

$$e(P_1, P_2)^{b\alpha s} = e(C_1, K_1)e(C_2, K_2)e(C_3, K_3)e(C_4, K_4)e(C_5 - \vartheta E_1, K_5)e(\vartheta E_2, D)$$

By breaking down the computation into several stages, we can show the correctness of decryption. Using the relations $r = r_1 + r_2$ and $V_2 + aV_2' = \tau P_2$, we have

$$\begin{aligned}
A_1 &= e(C_1, K_1)e(C_2, K_2)e(C_3, K_3)e(C_4, K_4)e(C_5, K_5) \\
&= e(P_1, P_2)^{b\alpha s} e(P_1, V_2)^{bsr} e(P_1, V_2')^{absr} e(P_1, P_2)^{-basz} e(P_1, P_2)^{basz} \\
&\quad \times e(P_1, P_2)^{-\tau bsr_2} e(P_1, P_2)^{-\tau bsr_1} e(W_1, P_2)^{tr_1} \\
&= e(P_1, P_2)^{b\alpha s} e(P_1, V_2 + aV_2' - \tau P_2)^{bsr} e(W_1, P_2)^{tr_1} \\
&= e(P_1, P_2)^{b\alpha s} e(P_1, W_2)^{tr_1}
\end{aligned}$$

The last step follows from the fact that $W_1 \sim W_2$. Also, we have

$$\begin{aligned}
A_2 &= \left(\frac{e(E_1, K_5)}{e(E_2, D)} \right)^{\vartheta} \\
&= \left(\frac{e(t(\mathsf{id}Q_1 + \mathsf{ctag}W_1 + U_1), r_1 P_2)}{e(tP_1, r_1(\mathsf{id}Q_2 + \mathsf{ktag}W_2 + U_2))} \right)^{1/(\mathsf{ctag}-\mathsf{ktag})} \\
&= \left(\frac{e(W_1, P_2)^{tr_1 \mathsf{ctag}}}{e(P_1, W_2)^{tr_1 \mathsf{ktag}}} \right)^{1/(\mathsf{ctag}-\mathsf{ktag})} \\
&= e(P_1, W_2)^{tr_1}
\end{aligned}$$

The masking factor is given by A_1/A_2.

5.3.2 Security Proof

Before proving security, we define algorithms that generate semi-functional ciphertexts and keys. These are used only in the security reduction and they cannot be computed without knowledge of the secret elements.

$\mathcal{IBE}1$.SFEncrypt$(\mathcal{PP}, \mathcal{MSK}, \mathcal{C}')$: Let $\mathcal{C}' = (C_0', C_1', C_2', C_3', E', \mathsf{ctag})$ be a normal ciphertext. Choose $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$. The semi-functional ciphertext is $(C_0, C_1, C_2, C_3, E, \mathsf{ctag})$ where $C_0 = C_0'$, $C_1 = C_1'$, $C_2 = C_2' + \mu P_1$, $C_3 = C_3' - \mu V_1'$ and $E = E'$.

$\mathcal{IBE}1$.SFKeyGen$(\mathcal{PP}, \mathcal{MSK}, \mathcal{SK}_{\mathsf{id}}')$: Let $\mathcal{SK}_{\mathsf{id}}' = (K_1', K_2', K_3', D, \mathsf{ktag})$ be a normal key. Choose a random γ from $\mathbb{Z}_p$. The semi-functional key is $(K_1, K_2, K_3, D, \mathsf{ktag})$ where $K_1 = K_1' - a\gamma P_2$, $K_2 = K_2' + \gamma P_2$, $K_3 = K_3'$ and $D = D'$.

One can decrypt a semi-functional ciphertext with a normal key and a normal ciphertext with a semi-functional key. This is easily seen by verifying that

$$e(b\mu P_1, K_2)e(\mu P_1, K_3)e(-\mu V_1', K_4)e(-b\mu V_1', K_5) = 1_T$$

and $e(C_1, -a\gamma P_2)e(C_2, \gamma P_2) = 1_T$ where K_2, K_3, K_4, K_5 and C_1, C_2 are normal key and ciphertext components respectively. However, decryption of a semi-functional ciphertext with a semi-functional key will fail because the message will be blinded by an additional factor of $e(P_1, P_2)^{b\mu\gamma}$.

Theorem 5.3.1. *If $(\varepsilon_{\text{DDH1}}, t_1)$-DDH1, $(\varepsilon_{\text{DLin2}}, t_2)$-DLin2 and $(\varepsilon_{\text{DBDH}}, t_3)$-DBDH assumptions hold in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ respectively, then $\mathcal{IBE}1$ is (ε, t, q)-IND-ID-CPA-secure where $\varepsilon \le \varepsilon_{\text{DDH1}} + q \cdot \varepsilon_{\text{DLin2}} + \varepsilon_{\text{DBDH}}$, $t_1 = t + O(q\rho)$, $t_2 = t + O(q\rho)$ and $t_3 = t + O(q\rho)$. ρ is the maximum time required for one scalar multiplication in $\mathbb{G}_1$ and $\mathbb{G}_2$.*

Proof. As is usual, the proof goes through a sequence of games. Let G_{real} denote the real security game. G_0 is just like G_{real} except that the challenge ciphertext is a semi-functional encryption of the chosen message. Let q be the number of key extraction queries made by the adversary during the attack. Define G_k for $1 \le k \le q$ such that the first k keys returned to the adversary are semi-functional and the rest are normal. Let G_{final} be defined similar to G_q except that now the challenge ciphertext is a semi-functional encryption of a random message. Let $X_\square$ denote the event that the adversary wins in $\mathsf{G}_\square$. Note that since the message is chosen at random, the bit β (in ind-cpa) would be information theoretically hidden from the attacker and therefore $\Pr[X_{final}] = 1/2$.

Using lemmas 5.3.1, 5.3.2 and 5.3.3, we have for any polynomial time attacker $\mathscr{A}$,

$$\begin{aligned}
\mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{IBE}1}(\mathscr{A}) &= \Pr[X_{real}] - \frac{1}{2} \\
&= \Pr[X_{real}] - \Pr[X_{final}] \\
&\le |\Pr[X_{real}] - \Pr[X_0]| + \sum_{k=1}^{q} (|\Pr[X_{k-1}] - \Pr[X_k]|) + |\Pr[X_q] - \Pr[X_{final}]| \\
&\le \varepsilon_{\text{DDH1}} + q\varepsilon_{\text{DLin2}} + \varepsilon_{\text{DBDH}}.
\end{aligned}$$

□

In what follows, let $\mathscr{B}_1$ denote a DDH1-solver, $\mathscr{B}_2$ a DLin2-solver and $\mathscr{B}_3$ is a DBDH-solver.

Lemma 5.3.1. $\Pr[X_{real}] - \Pr[X_0] \le \varepsilon_{\text{DDH1}}$.

Proof. We will show how to build a DDH1-solver $\mathscr{B}_1$ using $\mathscr{A}$'s ability to distinguish between G_{real} and G_0. The algorithm $\mathscr{B}_1$ receives $(\mathcal{G}, P_1, sP_1, aP_1, P_2, (as + \mu)P_1)$ as an instance of DDH1. The goal is to decide whether $\mu = 0$ or $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$. We describe how $\mathscr{B}_1$ will simulate each phase of the security game.

Setup: $\mathscr{B}$ chooses random elements $\alpha, b, y_v, y'_v, y_q, y_w, y_u$ from $\mathbb{Z}_p$ and sets the parameters as follows.

$$P_1 = P_1, aP_1 = aP_1, Q_1 = y_q P_1, W_1 = y_w P_1, U_1 = y_u P_1,$$

$$P_2 = P_2, V_2 = y_v P_2, V'_2 = y'_v P_2.$$

This implicitly sets $\tau = y_v + a y'_v$. Using this, the element τP_1 can be computed as $y_v P_1 + y'_v(aP_1)$. The simulator computes the remaining parameters using b, α and gives the following public parameters to $\mathscr{A}$.

$$\mathcal{PP} = \{\mathcal{G}, P_1, P_2, aP_1, bP_1, baP_1, \tau P_1, b\tau P_1, Q_1, W_1, U_1, e(P_1, P_2)^{b\alpha}\}.$$

Note that a and s (randomiser for challenge ciphertext) come from the assumption and are not known to $\mathscr{B}$.

Phase 1: $\mathscr{A}$ makes a number of key extract queries. $\mathscr{B}$ knows the master secret and using that it returns a normal key generated using the KeyGen algorithm for every key extract query made by $\mathscr{A}$.

Challenge: $\mathscr{B}$ receives the target identity $\widehat{\mathsf{id}}$ and two messages M_0 and M_1 from $\mathscr{A}$. It chooses $\beta \in \{0,1\}$ at random. To encrypt M_β, $\mathscr{B}$ chooses $t, \widehat{\mathsf{ctag}}$ at random from $\mathbb{Z}_p$ and computes the ciphertext elements as follows.

$C_0 = M_\beta \cdot e(sP_1, P_2)^{b\alpha}$,
$C_1 = b(sP_1)$, $C_2 = b(as+\mu)P_1$, $C_3 = (as+\mu)P_1$, $C_4 = -y_v(sP_1) - y'_v(as+\mu)P_1$,
$C_5 = -by_v(sP_1) - by'_v(as+\mu)P_1 + tW_1$
$E_1 = t(\widehat{\mathsf{id}}Q_1 + \widehat{\mathsf{ctag}}W_1 + U_1)$, $E_2 = tP_1$.

$\mathscr{B}$ returns $\widehat{\mathcal{C}} = (C_0, C_1, C_2, C_3, C_4, C_5, E_1, E_2, \widehat{\mathsf{ctag}})$ to $\mathscr{A}$. If $\mu = 0$, it is easy to see that $\widehat{\mathcal{C}}$ a a normal ciphertext. Otherwise, we have

$$C_2 = basP_1 + b\mu P_1,\ C_3 = asP_1 + \mu P_1$$

$$C_4 = -(y_v + ay'_v)sP_1 - y'_v\mu P_1 = -\tau sP_1 - \mu V'_1$$

$$C_5 = -b(y_v + ay'_v)sP_1 - by'_v\mu P_1 + tW_1 = -b\tau sP_1 + tW_1 - b\mu V'_1.$$

Then $\widehat{\mathcal{C}}$ is a semi-functional encryption of M_β. If $\widehat{\mathcal{C}}$ is normal then $\mathscr{B}_1$ simulates G_{real} and if it is semi-functional, $\mathscr{B}_1$ simulates G_0. Note that, to check whether $\widehat{\mathcal{C}}$ is semi-functional or not, $\mathscr{B}$ itself could try to decrypt it with a semi-functional key for $\widehat{\mathsf{id}}$. However since aP_2 is not known to $\mathscr{B}$, it cannot create such a key.

Phase 2: As in first phase, $\mathscr{B}$ returns a normal key for every query.

Guess: The adversary returns its guess β' to $\mathscr{B}$.

If $\beta = \beta'$, then $\mathscr{B}_1$ returns 1 and otherwise returns 0. We have,

$$\begin{aligned}|\Pr[X_{real}] - \Pr[X_0]| &= |\Pr[\beta = \beta' | \mu = 0] = \Pr[\beta = \beta' | \mu \xleftarrow{\text{U}} \mathbb{Z}_p]| \\ &= |\Pr[\mathscr{B}_1 \text{ returns } 1 | \mu = 0] = \Pr[\mathscr{B}_1 \text{ returns } 1 | \mu \xleftarrow{\text{U}} \mathbb{Z}_p]| \\ &= \mathsf{Adv}_{\mathcal{G}}^{\text{DDH1}}(\mathscr{B}_1) \\ &\le \varepsilon_{\text{DDH1}}\end{aligned}$$

□

Lemma 5.3.2. *For $k \in [1, q]$, $|\Pr[X_{k-1}] - \Pr[X_k]| \le \varepsilon_{\text{DLin2}}$.*

Proof. The algorithm $\mathscr{B}_2$ receives $(P_1, F_1, H_1, P_2, F_2, H_2, x_1P_2, x_2F_2, (x_1+x_2)H_2 + \gamma P_2)$ as an instance of the DLin problem. The task is to decide whether $\gamma = 0$ or $\gamma \xleftarrow{\text{U}} \mathbb{Z}_p$.

Setup: $\mathscr{B}$ chooses random elements $\alpha, a, \lambda, \nu, y'_v, y_q, y_w, y_u$ from $\mathbb{Z}_p$ and sets the parameters as follows.

$$P_1 = P_1, bP_1 = F_1, P_2 = P_2, bP_2 = F_2, V_2 = -aH_2, V'_2 = H_2 + y'_vP_2$$

$$Q_1 = -\lambda F_1 + y_q P_1, U_1 = -\nu F_1 + y_u P_1, W_1 = F_1 + y_w P_1$$

This sets $\tau = a y'_v$ which is known to the simulator. The remaining parameters required to provide $\mathcal{PP}$ to $\mathscr{A}$ are computed using a, α and τ and other elements of the DLin instance as shown below.

$$abP_1 = aF_1,\ \tau P_1 = a y'_v P_1,\ b\tau P_1 = a y'_v F_1,\ e(P_1, P_2)^{b\alpha} = e(F_1, P_2)^{\alpha}$$

Phases 1 and 2: Let $\mathsf{id}_1, \mathsf{id}_2, \ldots, \mathsf{id}_q$ denote the identities for which $\mathscr{A}$ requests the corresponding secret keys. In G_k, $\mathscr{B}$ changes the answer to k'th query from a normal key to a semi-functional one. Let $\mathcal{SK}'_{\mathsf{id}_i} = (K'_1, K'_2, K'_3, K'_4, K'_5, D', \mathsf{ktag})$ be a normally generated key for id_i. Note that $\mathscr{B}$ can create such a key because it knows the master secret. For $i < k$, the simulator should return a semi-functional key for id_i. $\mathscr{B}$ chooses $\gamma \in \mathbb{Z}_p$ at random, sets

$$K_1 = K'_1 - a\gamma P_2,\ K_2 = K'_2 + \gamma P_2,\ K_3 = K'_3,\ K_4 = K'_4,\ K_5 = K'_5,\ D = D'$$

and then returns the modified key $\mathcal{SK}_{\mathsf{id}_i}$ to $\mathscr{A}$. For $i > k$, $\mathscr{B}$ returns the normal key generated using the algorithm KeyGen.

When $i = k$, i.e., for identity id_k, suppose that a normal key $\mathcal{SK}'_{\mathsf{id}_k}$ is generated with r'_1, r'_2, z' as the randomisers and $\mathsf{ktag} = \lambda \mathsf{id}_k + \nu$. The simulator then modifies the key elements as follows.

$$K_1 = K'_1 - a((x_1 + x_2)H_2 + \gamma P_2),\ K_2 = K'_2 + (x_1 + x_2)H_2 + \gamma P_2 + y'_v(x_1 P_2),\ K_3 = K'_3 + y'_v(x_2 F_2)$$

$$K_4 = K'_4 + x_2 F_2,\ K_5 = K'_5 + x_1 P_2,\ D = D' + (y_q \mathsf{id}_k + y_w \mathsf{ktag} + y_u)(x_1 P_2)$$

implicitly setting $r_1 = r'_1 + x_1$, $r_2 = r'_2 + x_2$, $r = r' + x_1 + x_2$ and $z = z' + y'_v x_2$. Now $\mathscr{B}$ returns the secret key $\mathcal{SK}_{\mathsf{id}_k} = (K_1, \ldots, K_5, D, \mathsf{ktag})$.

If $\gamma = 0$, we have

$$\begin{aligned}
K_1 &= \alpha P_2 + r' V_2 - a(x_1 + x_2)H_2 \\
&= \alpha P_2 - a r' H_2 - a(x_1 + x_2)H_2 \\
&= \alpha P_2 - (r' + x_1 + x_2) a H_2 \\
&= \alpha P_2 + r V_2
\end{aligned}$$

and

$$\begin{aligned}
K_2 &= K'_2 + Z_2 + y'_v(x_1 P_2) \\
&= r' V'_2 - z' P_2 + (x_1 + x_2)H_2 + y'_v(x_1 P_2) \\
&= r' H_2 + r' y'_v P_2 - z' P_2 + (x_1 + x_2)H_2 + y'_v(x_1 P_2) + y'_v x_2 P_2 - y'_v x_2 P_2 \\
&= (r' + x_1 + x_2)H_2 + (r' + x_1 + x_2) y'_v P_2 - (z' + y'_v x_2) P_2 \\
&= r V'_2 - z P_2
\end{aligned}$$

indicating that $\mathcal{SK}_{\mathsf{id}_k}$ is a normally distributed key and so $\mathscr{B}$ perfectly simulates G_{k-1}. When $\gamma \xleftarrow{\mathrm{U}} \mathbb{Z}_p$, it is easy to see that $\mathcal{SK}_{\mathsf{id}_k}$ would be a semi-functional key for id_k in which case $\mathscr{B}$ is simulating G_k.

Challenge: $\mathscr{B}$ receives the challenge identity $\widehat{\mathsf{id}}$ and two messages M_0 and M_1 from $\mathscr{A}$. It chooses $\beta \xleftarrow{\mathrm{U}} \{0, 1\}$. It cannot generate a semi-functional ciphertext for the challenge identity without knowledge of bV_1' but setting $\widehat{\mathsf{ctag}} = \lambda\widehat{\mathsf{id}} + \nu$ enables it to do so. For a variable X over $\mathbb{Z}_p$, $\lambda X + \nu$ is a pairwise independent function for uniform λ and ν. Since λ and ν are hidden from the attacker, both $\widehat{\mathsf{ctag}}$ and ktag will be independently and uniformly distributed as required. $\mathscr{B}$ first generates a normal ciphertext $\mathcal{C}' = (C_0', C_1', C_2', C_3', C_4', C_5', E_1', E_2', \widehat{\mathsf{ctag}})$ with randomisers s, t' and $\widehat{\mathsf{ctag}} = \lambda\widehat{\mathsf{id}} + \nu$ using the $\mathsf{Encrypt}$ algorithm. Then it picks $\mu \in \mathbb{Z}_p$ at random and modifies $\mathcal{C}'$ as follows.

$$C_0 = C_0',\ C_1 = C_1',\ C_2 = C_2' + \mu F_1,\ C_3 = C_3' + \mu P_1,\ C_4 = C_4' - \mu H_1 - \mu y_v' P_1$$

$$C_5 = C_5' + \mu y_w H_1 - \mu y_v' F_1,\ E_1 = E_1' + \mu(y_q\widehat{\mathsf{id}} + y_w\widehat{\mathsf{ctag}} + y_u)H_1,\ E_2 = E_2' + \mu H_1.$$

Since the simulator does not know bH_1 it has to construct C_5 by setting $tP_1 = t'P_1 + \mu H_1$ implicitly. We need only verify that C_5 is well-formed; rest of the elements are constructed according to the original semi-functional algorithms.

$$\begin{aligned}
C_5 &= C_5' + \mu y_w H_1 - \mu y_v' F_1 \\
&= -b\tau s P_1 + t'W_1 + \mu y_w H_1 - \mu y_v' F_1 \\
&= -b\tau s P_1 + t'(F_1 + y_w P_1) + \mu y_w H_1 - \mu y_v' F_1 \\
&= -b\tau s P_1 + t'bP_1 + t'y_w P_1 + \mu y_w H_1 - \mu y_v' F_1 + b\mu H_1 - b\mu H_1 \\
&= -b\tau s P_1 + b(t'P_1 + \mu H_1) + y_w(t'P_1 + \mu H_1) - b\mu(y_v' P_1 + H_1) \\
&= -b\tau s P_1 + tW_1 - b\mu V_1'.
\end{aligned}$$

$\mathscr{B}$ returns $\widehat{\mathcal{C}} = (C_0, C_1, C_2, C_3, C_4, C_5, E_1, E_2, \widehat{\mathsf{ctag}})$ to $\mathscr{A}$.

In the key extraction phase, in order to generate a semi-functional ciphertext for id_k, the simulator must be able to compute bV_1' or create $t(\mathsf{id}_k Q_1 + \mathsf{ctag} W_1 + U_1) = (\mathsf{ctag} - \lambda\mathsf{id}_k - \nu)(t'F_1 + b\mu H_1) + (y_q\mathsf{id}_k + y_w\mathsf{ctag} + y_u)(t'P_1 + \mu H_1)$ which is possible only when $\mathsf{ctag} = \lambda\mathsf{id}_k + \nu$. Decryption of this ciphertext with $\mathcal{SK}_{\mathsf{id}_k}$ will fail unconditionally and hence the simulator gains no information.

If $\mathscr{A}$ wins i.e., $\beta = \beta'$, then $\mathscr{B}_2$ returns 1 and otherwise returns 0. We have,

$$\begin{aligned}
|\Pr[X_{k-1}] - \Pr[X_k]| &= |\Pr[\beta = \beta' | \gamma = 0] = \Pr[\beta = \beta' | \gamma \xleftarrow{\text{U}} \mathbb{Z}_p]| \\
&= |\Pr[\mathscr{B}_2 \text{ returns } 1 | \gamma = 0] = \Pr[\mathscr{B}_2 \text{ returns } 1 | \gamma \xleftarrow{\text{U}} \mathbb{Z}_p]| \\
&= \mathsf{Adv}_{\mathcal{G}}^{\text{DLin2}}(\mathscr{B}_2) \\
&\le \varepsilon_{\text{DLin2}}
\end{aligned}$$

□

Lemma 5.3.3. $\Pr[X_q] - \Pr[X_{final}] \le \varepsilon_{\text{DBDH}}$.

Proof. $\mathscr{B}_3$ receives $(P_1, xP_1, aP_1, sP_1, P_2, xP_2, aP_2, sP_2, Z)$ as an instance of the DBDH problem.

Setup: With $b, y_v, y_v', y_q, y_w, y_u$ chosen at random from $\mathbb{Z}_p$, $\mathscr{B}_3$ sets the parameters as

$$P_1 = P_1, P_2 = P_2, abP_1 = b(aP_1), V_2 = y_v P_2, V_2' = y_v' P_2,\ \tau P_1 = y_v P_1 + y_v'(aP_1)$$

$$Q_1 = y_q P_1, W_1 = y_w P_1, U_1 = y_u P_1, e(P_1, P_2)^{b\alpha} = e(xP_1, aP_2)^b$$

implicitly setting $a = a$, $\alpha = xa$ and $\tau = y_v + ay'_v$. The remaining parameters can be computed easily. $\mathscr{B}_3$ returns $\mathcal{PP}$ to $\mathscr{A}$.

Phases 1 and 2: When $\mathscr{A}$ asks for the secret key for the i'th identity id_i, $\mathscr{B}_3$ chooses at random $r_1, r_2, z', \mathsf{ktag}, \gamma' \in \mathbb{Z}_p$ with $r = r_1 + r_2$. It implicitly sets $\gamma' = x - \gamma$ and $z = z' + x$. Here the simulator knows γ' and not γ. It then computes a semi-functional key for id_i as follows.

$$\begin{aligned}
&K_1 = \gamma'(aP_2) + rV_2 = xaP_2 - a\gamma P_2 + rV_2 = \alpha P_2 + rV_2 - a\gamma P_2,\\
&K_2 = rV'_2 - z'P_2 - \gamma' P_2 = rV'_2 - z'P_2 - xP_2 + \gamma P_2 = rV'_2 - zP_2 + \gamma P_2,\\
&K_3 = bz'P_2 + b(xP_2) = bzP_2,\ K_4 = br_2P_2,\ K_5 = r_1P_2,\\
&D = r_1(\mathsf{id}_i Q_2 + \mathsf{ktag} W_2 + U_2).
\end{aligned}$$

Observe that $\mathscr{B}_3$ cannot create a normal key without knowing α.

Challenge: $\mathscr{B}_3$ receives the challenge identity $\widehat{\mathsf{id}}$ and two messages M_0 and M_1 from $\mathscr{A}$. It chooses $\beta \in \{0,1\}$ and $\widehat{\mathsf{ctag}}, \mu', t \in \mathbb{Z}_p$ at random and generates a semi-functional challenge ciphertext as follows. Here $\mathscr{B}_3$ implicitly sets $\mu' = \mu + as$ and it does not know μ.

$$\begin{aligned}
&C_0 = M_\beta \cdot Z^b,\ C_1 = b(sP_1)\\
&C_2 = b\mu' P_1 = basP_1 + b\mu P_1, C_3 = \mu' P_1 = asP_1 + \mu P_1\\
&C_4 = -y_v(sP_1) - \mu' y'_v P_1 = -y_v sP_1 - asy'_v P_1 - \mu y'_v P_1 = -\tau sP_1 - \mu V'_1\\
&C_5 = -by_v(sP_1) - b\mu' y'_v P_1 + tW_1 = -by_v sP_1 - basy'_v P_1 - b\mu y'_v P_1 + tW_1 = -\tau sP_1 + tW_1 - \mu V'_1\\
&E_1 = t(\widehat{\mathsf{id}} Q_1 + \widehat{\mathsf{ctag}} W_1 + U_1),\ E_2 = tP_1.
\end{aligned}$$

The challenge ciphertext $\widehat{\mathcal{C}} = (C_0, C_1, C_2, C_3, C_4, C_5, E_1, E_2, \widehat{\mathsf{ctag}})$ is returned to $\mathscr{A}$. If Z equals $e(P_1, P_2)^{xas}$ then $\widehat{\mathcal{C}}$ will be a semi-functional encryption of M_β; if Z is a random element of $\mathbb{G}_T$ then $\widehat{\mathcal{C}}$ will be a semi-functional encryption of a random message. $\mathscr{B}_3$ returns 1 if $\beta = \beta'$; otherwise it returns 0. We have,

$$\begin{aligned}
|\Pr[X_q] - \Pr[X_{final}]| &= |\Pr[\beta = \beta' | Z = e(P_1, P_2)^{xas}] = \Pr[\beta = \beta' | Z \xleftarrow{\text{U}} \mathbb{G}_T]|\\
&= \mathsf{Adv}_{\mathcal{G}}^{\text{DBDH}}(\mathscr{B}_3)\\
&\le \varepsilon_{\text{DBDH}}
\end{aligned}$$

□

5.4 Intermediate IBE Constructions

In this section, we provide the descriptions of schemes $\mathcal{IBE}2$, $\mathcal{IBE}3$, $\mathcal{IBE}4$ and $\mathcal{IBE}5$. Only the constructions are provided here. These schemes primarily serve the purpose of showing the stepping stones in moving from $\mathcal{IBE}1$ to $\mathcal{IBE}6$. Along with the construction, we also describe the semi-functional ciphertexts and keys for all the schemes despite the fact that these are only used in the security proofs. The reason is that the structure of semi-functional components provides an idea of the kind of assumptions a security reduction might rely on. In addition, it provides a picture of how different scalars (including the ones in the semi-functional portions) interact with one another. We use a compact notation to denote normal and semi-functional ciphertexts and keys. The group elements shown in curly brackets { } are the semi-functional components. To get the schemes itself, these components should be ignored.

We fix some common notation and parameters for all the four schemes. $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, F_1, F_2)$ is an asymmetric pairing chosen according to the security parameter κ. $P_1 \xleftarrow{\text{U}} \mathbb{G}_1^\times$ and $P_2 \xleftarrow{\text{U}} G_2$ are generators chosen randomly. $Q_1, W_1, U_1 \xleftarrow{\text{U}} \mathbb{G}_1$ and $Q_2, W_2, U_2 \longleftarrow \mathbb{G}_2$ are chosen such that $\mathsf{dlog}_{P_2}(Q_2, W_2, U_2) = \mathsf{dlog}_{P_1}(Q_1, W_1, U_1)$.

5.4.1 Scheme $\mathcal{IBE}2$

Choose $\alpha, a_1, a_2, v, v', v'' \xleftarrow{\text{U}} \mathbb{Z}_p$ and define $V_2 = vP_2, V_2' = v'P_2, V_2'' = v''P_2$. Define $\tau_1 = v + a_1 v'$ and $\tau_2 = v + a_2 v''$ so that $\tau_1 P_2 = V_2 + a_1 V_2'$ and $\tau_2 P_2 = V_2 + a_2 V_2''$.

The public parameters and master secret are given by

$$\begin{aligned}&\mathcal{PP} \;\;: (\mathcal{G}, P_1, a_1P_1, a_2P_1, \tau_1P_1, \tau_2P_1, Q_1, W_1, U_1, e(P_1, P_2)^\alpha),\\ &\mathcal{MSK}\text{: } (P_2, \alpha P_2, V_2, V_2', V_2'', Q_2, W_2, U_2).\end{aligned}$$

Ciphertext: Ciphertext is given by $\mathcal{C} = (C_1, C_2, C_3, C_4, E_1, E_2, \mathsf{ctag})$ where

$$\begin{aligned}&s_1, s_2, t, \mathsf{ctag} \xleftarrow{\text{U}} \mathbb{Z}_p,\; s = s_1 + s_2,\; \{\mu \xleftarrow{\text{U}} \mathbb{Z}_p\}\\ &C_0 = M \cdot e(P_1, P_2)^{\alpha s},\\ &C_1 = sP_1,\; C_2 = a_1 s_1 P_1,\; C_3 = a_2 s_2 P_1 \,\{+a_2\mu P_1\},\; C_4 = -(\tau_1 s_1 + \tau_2 s_2)P_1 + tW_1 \,\{-a_2\mu V_1''\},\\ &E_1 = t(\mathsf{id}Q_1 + \mathsf{ctag}W_1 + U_1),\; E_2 = tP_1\end{aligned}$$

Here $V_1'' \in \mathbb{G}_1$ is such that $\mathsf{dlog}_{P_1} V_1'' = \mathsf{dlog}_{P_2} V_2''$ and $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$.

Key: Pick $r, \mathsf{ktag} \xleftarrow{\text{U}} \mathbb{Z}_p$, $\{\gamma \xleftarrow{\text{U}} \mathbb{Z}_p\}$ and generate the key $\mathcal{SK}_{\mathsf{id}} = (K_1, K_2, K_3, K_4, D, \mathsf{ktag})$ as follows.

$$\begin{aligned}&K_1 = \alpha P_2 + rV_2 \,\{-a_1a_2\gamma P_2\},\; K_2 = rV_2' \,\{+a_2\gamma P_2\},\; K_3 = rV_2'' \,\{+a_1\gamma P_2\},\; K_4 = rP_2\\ &D = r(\mathsf{id}Q_2 + \mathsf{ktag}W_2 + U_2)\end{aligned}$$

Security: With the DLin assumption, it is possible to show that G_{real} and G_0 are indistinguishable. We give a brief sketch of the proof. Let $(P_1, F_1, H_1, x_1P_1, x_2F_1, P_2, F_2, H_2, Z_1)$ be an instance of DLin that the simulator $\mathscr{B}$ receives. It chooses $\alpha, y_q, y_w, y_u, y_v, y_v', y_v'' \xleftarrow{\text{U}} \mathbb{Z}_p$ and constructs the parameters as $P_1 = P_1, a_1P_1 = F_1, a_2P_1 = H_1, Q_1 = y_qP_1, W_1 = y_wP_1, U_1 = y_uP_1, V_2 = y_vP_2, V_2' = y_v'P_2, V_2'' = y_v''P_2$, implicitly setting $\tau_1 = y_v + a_1y_v'$ and $\tau_2 = y_v + a_1y_v''$. Since $\mathscr{B}$ has the master secret it can create a normal key for any identity. It answers all the key extraction queries with normal keys. In the challenge phase it receives two messages M_0, M_1 and an identity $\widehat{\mathsf{id}}$. It then chooses $\beta \xleftarrow{\text{U}} \{0,1\}$. A normal ciphertext $C_0', \ldots, C_4', E_1', E_2', \mathsf{ctag}$ for identity $\widehat{\mathsf{id}}$ and message M_β is first generated using randomisers s_1', s_2', t and then modified as follows:

$$C_0 = C_0' \cdot e(x_1P_1, P_2)^\alpha, C_1 = C_1' + x_1P_1, C_2 = C_2' - x_2F_1$$

$$C_3 = C_3' + Z_1, C_4 = C_4' - y_vx_1P_1 + y_v'x_2F_1 - y_v''Z_1, E_1 = E_1', E_2 = E_2'$$

implicitly setting $s_1 = s_1' - x_2$, $s_2 = s_2' + x_1 + x_2$ and $s = s_1 + s_2 = s' + x_1$. If $Z_1 = (x_1 + x_2)H_1$ then the ciphertext is normal; otherwise $Z_1 = (x_1 + x_2 + c)H_1$ for some $c \xleftarrow{\text{U}} \mathbb{Z}_p$ whence the ciphertext is semi-functional with μ being set to c.

Now let's take a look at the second reduction and it's proof using DDH2 and DLin2 assumptions. First consider the DDH2 assumption. Let $(P_1, P_2, x_1P_2, x_2P_2, Z_2)$ be an instance of DDH2. As mentioned earlier, except for P_1 such an instance consists of elements of $\mathbb{G}_2$. The $\mathcal{PP}$, on the other hand, consists entirely of elements of $\mathbb{G}_1$. There is no way that the $\mathcal{PP}$ can be based on the given instance and so getting a reduction is not possible. Now consider the DLin assumption. In this scheme, the randomiser r is not split. As such there is no way to base this reduction on the DLin assumption.

5.4.2 Scheme $\mathcal{IBE}3$

Let $\alpha, a, b \xleftarrow{\text{U}} \mathbb{Z}_p$. Also, let $V_2, V_2' \xleftarrow{\text{U}} \mathbb{G}_2$ and $\tau \in \mathbb{Z}_p$ be defined such that $\tau P_2 = V_2 + aV_2'$. The public parameters and master secret are given by

$$\begin{aligned} \mathcal{PP} &: (\mathcal{G}, P_1, P_2, bP_1, abP_1, \tau P_1, b\tau P_1, Q_1, W_1, U_1, e(P_1, P_2)^{b\alpha}), \\ \mathcal{MSK} &: (\alpha P_2, bP_2, V_2, V_2', Q_2, W_2, U_2). \end{aligned}$$

V_1' will have the usual meaning.

Ciphertext: Pick $s, t, \mathsf{ctag} \xleftarrow{\text{U}} \mathbb{Z}_p, \{\mu \xleftarrow{\text{U}} \mathbb{Z}_p\}$ and set the ciphertext as $\mathcal{C} = (C_0, C_1, \ldots, C_4, E_1, E_2, \mathsf{ctag})$ where

$$\begin{aligned} &C_0 = M \cdot e(P_1, P_2)^{b\alpha s}, \\ &C_1 = bsP_1,\ C_2 = basP_1\{+b\mu P_1\},\ C_3 = -\tau sP_1\ \{-\mu V_1'\},\ C_4 = -b\tau sP_1 + tW_1\ \{-b\mu V_1'\}, \\ &E_1 = t(\mathsf{id}Q_1 + \mathsf{ctag}W_1 + U_1),\ E_2 = tP_1. \end{aligned}$$

Key: Compute the key $\mathcal{SK}_{\mathsf{id}} = (K_1, \ldots, K_4, D)$ as follows.

$$\begin{aligned} &r_1, r_2, \mathsf{ktag} \xleftarrow{\text{U}} \mathbb{Z}_p,\ r = r_1 + r_2,\ \{\gamma \xleftarrow{\text{U}} \mathbb{Z}_p\} \\ &K_1 = \alpha P_2 + rV_2\ \{-a\gamma P_2\},\ K_2 = rV_2'\ \{+\gamma P_2\},\ K_3 = br_2P_2,\ K_4 = r_1P_2. \\ &D = r_1(\mathsf{id}Q_2 + \mathsf{ktag}W_2 + U_2). \end{aligned}$$

Security: The first reduction goes through with the DDH1 assumption. We provide a brief sketch. Suppose that $(P_1, aP_1, sP_1, P_2, Z_1)$ is the instance of DDH1. It simulates the game in the following way – choose $\alpha, b, y_v, y_v', y_q, y_u, y_w \xleftarrow{\text{U}} \mathbb{Z}_p$ and set $P_1 = P_1, aP_1 = aP_1, Q_1 = y_qP_1, W_1 = y_wP_1, U_1 = y_uP_1, P_2 = P_2V_2 = y_vP_2, V_2' = y_v'P_2$. Construct the challenge ciphertext as

$$C_0 = M_\beta \cdot e(sP_1, P_2)^{b\alpha}, C_1 = b(sP_1), C_2 = bZ_1, C_3 = -y_v(sP_1) - y_v'Z_1, C_4 = -by_v(sP_1) - by_v'Z_1 + tW_1.$$

Computing E_1 and E_2 is straightforward. Note that the randomiser s comes from the assumption.

For the second reduction, we cannot rely on DDH2 assumption for the same reason as that for Scheme 2. In this case, there is a split of the randomiser r. So, one may hope to base the reduction on the DLin problem.

Consider an instance of the DLin assumption: $(P_1, F_1, H_1, P_2, F_2, H_2, x_1P_2, x_2F_2, Z_2)$. The simulator $\mathscr{B}$ chooses $\alpha, b, y'_v, y_q, y_u, y_w \xleftarrow{\text{U}} \mathbb{Z}_p$ and computes the parameters as $P_1 = P_1, P_2 = P_2, bP_1 = F_1, V_2 = -aH_2, V'_2 = aH_2 + y'_vP_2$ so that Z_2 could be embedded in K_1 and K_2. Observe that $\tau = ay'_v$. We can successfully construct K_1, K_3 and K_4 setting $r_1 = r'_1 + x_1$, $r_2 = r'_2 + x_2$ and $r = r' + x_1 + x_2$, but composing K_2 with the same r is impossible. Any other way of using DLin assumption also results in failure.

5.4.3 Scheme $\mathcal{IBE}4$

Pick $\alpha, a, b \xleftarrow{\text{U}} \mathbb{Z}_p$. τ, V_2, V'_2 are as defined in $\mathcal{IBE}3$. The public parameters and master secret are given by

$\mathcal{PP}$: $(\mathcal{G}, P_1, P_2, aP_1, abP_1, \tau P_1, Q_1, W_1, U_1, e(P_1, P_2)^{\alpha})$,
$\mathcal{MSK}$: $(\alpha P_2, bP_2, V_2, V'_2, Q_2, W_2, U_2)$.

Ciphertext: Pick $s, t \xleftarrow{\text{U}} \mathbb{Z}_p$, $\{\mu \xleftarrow{\text{U}} \mathbb{Z}_p\}$ and compute $\mathcal{C} = (C_1, C_1, \ldots, C_4, E_1, E_2, \mathsf{ctag})$ as

$C_0 = M \cdot e(P_1, P_2)^{\alpha s}$,
$C_1 = sP_1$, $C_2 = asP_1\{+\mu P_1\}$, $C_3 = basP_1\{+b\mu P_1\}$, $C_4 = -\tau sP_1 + tW_1\ \{-\mu V'_1\}$,
$E_1 = t(\mathsf{id}Q_1 + \mathsf{ctag}W_1 + U_1)$, $E_2 = tP_1$.

Key: Let $r, z \xleftarrow{\text{U}} \mathbb{Z}_p$, $\{\gamma \xleftarrow{\text{U}} \mathbb{Z}_p\}$. The secret key $\mathcal{SK}_{\mathsf{id}} = (K_1, \ldots, K_4, D)$ is defined as follows.

$K_1 = \alpha P_2 + rV_2\ \{-a\gamma P_2\}$, $K_2 = rV'_2 - bzP_2\ \{+\gamma P_2\}$, $K_3 = zP_2$, $K_4 = rP_2$,
$D = r(\mathsf{id}Q_2 + \mathsf{ktag}W_2 + U_2)$.

Here, the first reduction goes through with the DDH1 assumption. The argument is similar to that of Scheme 3. As before, the second reduction cannot be based on the DDH2 assumption. Since there is no split of the randomiser r, there is no way to base the second reduction on the DLin assumption.

5.4.4 Scheme $\mathcal{IBE}5$

Choose $\alpha, a \xleftarrow{\text{U}} \mathbb{Z}_p$. Let $V_2, V'_2 \in \mathbb{G}_2$ and $\tau P_2 = V_2 + aV'_2$. V'_1 will have the usual meaning. The $\mathcal{PP}$ and $\mathcal{MSK}$ are given by

$\mathcal{PP}$: $(\mathcal{G}, P_1, P_2, aP_1, \tau P_1, Q_1, W_1, U_1, e(P_1, P_2)^{\alpha})$,
$\mathcal{MSK}$: $(\alpha P_2, V_2, V'_2, Q_2, W_2, U_2)$.

Ciphertext: Choose $s, t, \mathsf{ctag} \xleftarrow{\text{U}} \mathbb{Z}_p$, $\{\mu \xleftarrow{\text{U}} \mathbb{Z}_p\}$ and set $\mathcal{C} = (C_0, C_1, C_2, C_3, E_1, E_2, \mathsf{ctag})$ as below.

$C_0 = M \cdot e(P_1, P_2)^{\alpha s}$, $C_1 = sP_1$, $C_2 = asP_1\ \{+\mu P_1\}$, $C_3 = -\tau sP_1 + tW_1\ \{-\mu V'_1\}$,
$E_1 = t(\mathsf{id}Q_1 + \mathsf{ctag}W_1 + U_1)$, $E_2 = tP_1$.

Key: Let $r \overset{\text{U}}{\longleftarrow} \mathbb{Z}_p$, $\{\gamma \overset{\text{U}}{\longleftarrow} \mathbb{Z}_p\}$. The secret key $\mathcal{SK}_{\mathsf{id}}$ is given by the tuple $(K_1, K_2, K_3, D, \mathsf{ktag})$ where

$K_1 = \alpha P_2 + rV_2 \; \{-a\gamma P_2\}$, $K_2 = rV_2' \; \{+\gamma P_2\}$, $K_3 = rP_2$,
$D = r(\mathsf{id}Q_2 + \mathsf{ktag}W_2 + U_2)$.

Security: The first security reduction can be based on DDH1 assumption but the second reduction does not hold with either DDH2 or the DLin assumption.

5.5 Scheme $\mathcal{IBE6}$

We describe here the construction of $\mathcal{IBE6} = (\mathcal{IBE6}.\mathsf{Setup}, \mathcal{IBE6}.\mathsf{KeyGen}, \mathcal{IBE6}.\mathsf{Encrypt}, \mathcal{IBE6}.\mathsf{Decrypt})$ and also provide a proof of security based on DDH1, the new assumption DDH2v and DBDH.

5.5.1 Construction

$\mathcal{IBE6}.\mathsf{Setup}(\kappa)$: Choose $P_1 \overset{\text{U}}{\longleftarrow} \mathbb{G}_1^{\times}$, $P_2 \overset{\text{U}}{\longleftarrow} \mathbb{G}_2^{\times}$, $Q_1, W_1, U_1 \overset{\text{U}}{\longleftarrow} \mathbb{G}_1$, $Q_2, W_2, U_2 \longleftarrow \mathbb{G}_2$ such that $\mathsf{dlog}_{P_1}(Q_1, U_1, W_1) = \mathsf{dlog}_{P_2}(Q_2, U_2, W_2)$. Also, pick $\alpha \overset{\text{U}}{\longleftarrow} \mathbb{Z}_p$. Let $a, v, v' \overset{\text{u}}{\longleftarrow} \mathbb{Z}_p$. Set $V_2 = vP_2$, $V_2' = v'P_2$ and $\tau = v + av'$ so that $\tau P_2 = V_2 + aV_2'$. Set the parameters as follows.

$\mathcal{PP}\;\;$: $(P_1, aP_1, \tau P_1, Q_1, W_1, U_1, e(P_1, P_2)^{\alpha})$.
$\mathcal{MSK}$: $(P_2, \alpha P_2, V_2, V_2', Q_2, W_2, U_2)$.

$\mathcal{IBE6}.\mathsf{Encrypt}(M, \mathsf{id}, \mathcal{PP})$: Choose random s, ctag from $\mathbb{Z}_p$; $\mathcal{C}$ is $(C_0, C_1, C_2, C_3, E, \mathsf{ctag})$ where the elements are defined as follows.

$C_0 = M \cdot e(P_1, P_2)^{\alpha s}$,
$C_1 = sP_1$, $C_2 = asP_1$, $C_3 = -\tau sP_1 + sW_1$, $E = s(\mathsf{id}Q_1 + \mathsf{ctag}W_1 + U_1)$.

$\mathcal{IBE6}.\mathsf{KeyGen}(\mathsf{id}, \mathcal{MSK}, \mathcal{PP})$: Choose random r, ktag from $\mathbb{Z}_p$; $\mathcal{SK}_{\mathsf{id}}$ is $(K_1, K_2, K_3, D, \mathsf{ktag})$ where the elements are defined as follows.

$K_1 = \alpha P_2 + rV_2$, $K_2 = rV_2'$, $K_3 = rP_2$, $D = r(\mathsf{id}Q_2 + \mathsf{ktag}W_2 + U_2)$.

$\mathcal{IBE6}.\mathsf{Decrypt}(\mathcal{C}, \mathsf{id}, \mathcal{SK}_{\mathsf{id}}, \mathcal{PP})$: As before, decryption succeeds only when $\mathsf{ctag} \neq \mathsf{ktag}$. Define $\vartheta = (\mathsf{ctag} - \mathsf{ktag})^{-1}$. Decryption is done by unmasking the message as follows.

$$M = \frac{C_0}{e(C_1, K_1 + \vartheta D)e(C_2, K_2)e(C_3 - \vartheta E, K_3)}$$

The correctness of decryption is shown by the following calculations. We break up the denominator for unmasking the message into two parts - A_1 and A_2 such that $A_1 A_2$ gives the masking factor.

$$
\begin{aligned}
A_1 &= e(C_1, \vartheta D)e(-\delta E, K_3) \\
&= e(C_1, D)^{\vartheta} e(-E, K_3)^{\vartheta} \\
&= e(sP_1, r(\mathsf{id}Q_2 + \mathsf{ktag}W_2 + U_2))^{\vartheta} e(-s(\mathsf{id}Q_1 + \mathsf{ctag}W_1 + U_1), rP_2)^{\vartheta} \\
&= e(-(\mathsf{id}Q_1 + \mathsf{ktag}W_1 + U_1), P_2)^{-rs\vartheta} e(\mathsf{id}Q_1 + \mathsf{ctag}W_1 + U_1, P_2)^{-rs\vartheta} \\
&= e(\vartheta(\mathsf{ctag} - \mathsf{ktag})W_1, P_2)^{rs} \\
&= e(W_1, P_2)^{-rs} \\
A_2 &= e(C_1, K_1)e(C_2, K_2)e(C_3, K_3) \\
&= e(sP_1, \alpha P_2 + rV_2)e(asP_1, rV_2')e(\tau sP_1 + sW_1, rP_2) \\
&= e(P_1, P_2)^{\alpha s} e(P_1, V_2 + aV_2' - \tau P_2)^{rs} e(W_1, P_2)^{rs} \\
&= e(P_1, P_2)^{\alpha s} e(W_1, P_2)^{rs}
\end{aligned}
$$

Conversion to Signature Scheme: There is a "dual" of $\mathcal{IBE}6$ where the ciphertext elements are in $\mathbb{G}_2$ and decryption keys consist of elements of $\mathbb{G}_1$. Using Naor's observation, this dual of $\mathcal{IBE}6$ can be converted to a secure signature scheme. The signatures will be composed of elements of $\mathbb{G}_1$ and will be smaller than the signatures obtained by the conversion of Waters' 2009 scheme to a signature scheme. In a similar manner, one can convert the dual of the HIBE in Section 5.7 to obtain a HIBS scheme where signatures consist elements of $\mathbb{G}_1$.

Extension to HIBE and BE: Waters extends the IBE scheme in [158] in a natural way to a HIBE scheme. We show that our simplification of Waters scheme retains the original flexibility. In Section 5.7, we describe a HIBE, named $\mathcal{HIBE}6$, which extends $\mathcal{IBE}6$. This HIBE scheme is secure under the DDH1, DDH2v and the DBDH assumptions and provides lesser and smaller parameters and better efficiencies of key generation, delegation, encryption and decryption compared to the HIBE in [158].

The full version of Waters paper described a broadcast encryption (BE) scheme based on the dual system IBE in [158]. In Section 5.8, we describe a BE scheme $\mathcal{BE}6$ built upon on $\mathcal{IBE}6$ along with a security proof based on the hardness of the DDH1, DDH2v and DBDH problems. The new BE scheme provides adaptive security and is more efficient (in terms of header size) than previously known BE schemes providing adaptive security [87, 158].

5.5.2 On the Assumption DDH2v

Here, we present a discussion on why DDH2 is not directly applicable to $\mathcal{IBE}6$ and explain the need for using the new assumption (DDH2v). Consider an instance of DDH2 – $(P_1, P_2, x_1P_2, x_2P_2, Z_1)$ where $Z_1 = x_1x_2P_2$ or $Z_1 \xleftarrow{\text{U}} \mathbb{G}_1$. The instance has a single element P_1 from $\mathbb{G}_1$. For our proofs, we will require some information about x_1P_1 to be carried as part of the instance. If the instance is directly augmented by x_1P_1, then the problem becomes easy, since one can compute the pairing $e(x_1P_1, x_2P_2)$ and compare to $e(P_1, Z_2)$. Suppose that instead of x_1P_1 we include the elements zP_1

and zx_1P_1 where z is chosen randomly from $\mathbb{Z}_p$. This pair of elements carries some information about x_1P_1, but, not the element itself. An instance will now be $(P_1, zP_1, zx_1P_1, P_2, x_1P_2, x_2P_2, Z_2)$. It, however, is easy to check whether Z_2 equals $x_1x_2P_2$ by checking whether $e(zx_1P_1, x_2P_2)$ equals $e(zP_1, Z_2)$. This suggests that the information about zP_1 itself needs to be blinded by another randomiser. So, instead of having zP_1 directly, the elements dP_1, dzP_1 and dP_2 are included where d is a random element of $\mathbb{Z}_p$. The information about x_1P_1 is carried by the elements dP_1, dzP_1, zx_1P_1 and dP_2. Augmenting an instance of DDH2 with these elements embeds information about x_1P_1 but, does not seem to provide any way to use this information to determine whether Z_2 is real or random. The entire thing is formulated as DDH2v assumption.

In DDH2v, there is a two-level blinding of x_1P_1. We have seen that providing x_1P_1 directly or using a single-level blinding makes the problem easy. So, a two-level blinding is the minimum that one has to use to get to an assumption about hardness.

The assumption DDH2v (the "v" stands for variant) is no harder than DDH2. This is because an instance of DDH2v contains an embedded instance of DDH2 and an algorithm to solve DDH2 can be invoked on this embedded instance to solve the instance of DDH2v. On the other hand, there is no clear way of using an algorithm to solve DDH2v to solve DDH2. Intuitively, this is due to the fact that an instance of DDH2v contains some information about x_1P_1 whereas an instance of DDH2 does not contain any such information.

In our security proofs, we will use the assumption DDH2v. Since assumption DDH2v does not appear earlier in the literature, it is a non-standard assumption. Having said this, we would also like to remark that DDH2v arises naturally as a minimal assumption when one tries to augment an instance of DDH2 with some information about x_1P_1 while maintaining the hardness of the problem. For a formal definition of DDH2v, look at Section 2.3.2.1.

In Section 2.3.4, we provide a proof of the security of assumption DDH2v in the generic group model. We feel that assumption DDH2v will have applications elsewhere for schemes based on asymmetric pairings.

5.5.3 Security of Scheme $\mathcal{IBE}6$

Now, we describe algorithms $\mathcal{IBE}6$.SFEncrypt and $\mathcal{IBE}6$.SFKeyGen that generate semi-functional ciphertexts and keys for $\mathcal{IBE}6$.

$\mathcal{IBE}6.\mathsf{SFEncrypt}(\mathcal{PP}, \mathcal{MSK}, \mathcal{C}')$: Let $\mathcal{C}' = (C_0', C_1', C_2', C_3', E', \mathsf{ctag})$ be a ciphertext normally generated by the Encrypt algorithm for message M and identity id. Let V_1' be an element of $\mathbb{G}_1$ such that $\mathsf{dlog}_{P_1} V_1' = \mathsf{dlog}_{P_2} V_2'$. Choose $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$. The semi-functional ciphertext generation algorithm will modify the normal ciphertext as: $C_0 = C_0'$, $C_1 = C_1'$, $E = E'$ and

$$C_2 = C_2' + \mu P_1,\ C_3 = C_3' - \mu V_1'.$$

$\mathcal{IBE}6.\mathsf{SFKeyGen}(\mathcal{PP}, \mathcal{MSK}, \mathcal{SK}_{\mathsf{id}}')$: Let $\mathcal{SK}_{\mathsf{id}}' = (K_1', K_2', K_3', D', \mathsf{ktag})$ be a normal secret key generated by the KeyGen algorithm for identity id. Pick $\gamma \xleftarrow{\text{U}} \mathbb{Z}_p$. The semi-functional key generation algorithm will modify the normal key as: $K_3 = K_3'$, $D = D'$ and

$$K_1 = K_1' - a\gamma P_2,\ K_2 = K_2' + \gamma P_2.$$

Below we state the security theorem for $\mathcal{IBE}6$.

Theorem 5.5.1. *If* $(\varepsilon_{\mathrm{DDH1}}, t_1)$-DDH1*,* $(\varepsilon_{\mathrm{DLin2}}, t_2)$-DDH2v *and* $(\varepsilon_{\mathrm{DBDH}}, t_3)$-DBDH *assumptions hold in* $\mathbb{G}_1$*,* $\mathbb{G}_2$ *and* $\mathbb{G}_T$ *respectively, then* $\mathcal{IBE}6$ *is* (ε, t, q)-IND-ID-CPA*-secure where* $\varepsilon \le \varepsilon_{\mathrm{DDH1}} + q \cdot \varepsilon_{\mathrm{DDH2v}} + \varepsilon_{\mathrm{DBDH}}$*,* $t_1 = t + O(q\rho)$*,* $t_2 = t + O(q\rho)$ *and* $t_3 = t + O(q\rho)$*.* ρ *is the maximum time required for one scalar multiplication in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*.*

Proof. Let G_{real}, G_k (for $0 \le k \le q$) and G_{final} be as defined in the security proof for scheme $\mathcal{IBE}1$.

Using lemmas 5.5.1, 5.5.2 and 5.5.3, we have for any polynomial time attacker $\mathscr{A}$,

$$\begin{aligned}
\mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{IBE}6}(\mathscr{A}) &= |\Pr[X_{real}] - \Pr[X_{final}]| \\
&\le |\Pr[X_{real}] - \Pr[X_0]| + \sum_{k=1}^{q} (|\Pr[X_{k-1}] - \Pr[X_k]|) + |\Pr[X_q] - \Pr[X_{final}]| \\
&\le \varepsilon_{\mathrm{DDH1}} + q \cdot \varepsilon_{\mathrm{DDH2v}} + \varepsilon_{\mathrm{DBDH}}
\end{aligned}$$

□

In the sequel, let $\mathscr{B}_1$ denote a DDH1-solver, $\mathscr{B}_2$ a DDH2-solver and $\mathscr{B}_3$ is a DBDH-solver.

Lemma 5.5.1. $|\Pr[X_{real}] - \Pr[X_0]| \le \varepsilon_{\mathrm{DDH1}}$.

Proof. The algorithm $\mathscr{B}_1$ receives $(P_1, sP_1, aP_1, P_2, (as+\mu)P_1)$ as an instance of DDH1. We describe how it will simulate each phase of the security game.

Setup: $\mathscr{B}_1$ chooses elements $\alpha, y_v, y'_v, y_q, y_w, y_u \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and sets the parameters as follows.

$$P_1 = P_1, aP_1 = aP_1, Q_1 = y_q P_1, W_1 = y_w P_1, U_1 = y_u P_1$$

$$P_2 = P_2, V_2 = y_v P_2, V'_2 = y'_v P_2, Q_2 = y_q P_2, W_2 = y_w P_2, U_2 = y_u P_2$$

The element τP_1 is computed as $y_v P_1 + y'_v(aP_1)$ implicitly setting $\tau = y_v + a y'_v$. The simulator computes the remaining parameters using α and gives the following public parameters to $\mathscr{A}$.

$$\mathcal{PP} = (\mathcal{G}, P_1, P_2, aP_1, \tau P_1, Q_1, W_1, U_1, e(P_1, P_2)^{\alpha})$$

Phase 1: $\mathscr{A}$ makes a number of key extract queries. $\mathscr{B}_1$ knows the master secret and using that it returns a normal key for every key extract query made by $\mathscr{A}$.

Challenge: $\mathscr{B}_1$ receives the target identity $\widehat{\mathsf{id}}$ and two messages M_0 and M_1 from $\mathscr{A}$. It chooses a random bit $\beta \xleftarrow{\mathrm{U}} \{0,1\}$. To encrypt M_β, $\mathscr{B}_1$ picks $\widehat{\mathsf{ctag}} \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and computes the ciphertext elements as follows.

$C_0 = M_\beta \cdot e(sP_1, P_2)^{\alpha}$,
$C_1 = sP_1$, $C_2 = (as+\mu)P_1$, $C_3 = -y_v(sP_1) - y'_v(as+\mu)P_1 + y_w(sP_1)$,
$E = (\widehat{\mathsf{id}} y_q + \widehat{\mathsf{ctag}} y_w + y_u)(sP_1)$.

$\mathscr{B}_1$ returns $\widehat{\mathcal{C}} = (C_0, C_1, C_2, C_3, E, \widehat{\mathsf{ctag}})$ to $\mathscr{A}$.

If $\mu = 0$, the challenge ciphertext is normal; otherwise the ciphertext is semi-functional with μ coming from the instance.

Note that, to check whether $\widehat{\mathcal{C}}$ is semi-functional or not, $\mathscr{B}_1$ itself could try to decrypt it with a semi-functional key for $\widehat{\mathsf{id}}$. However since aP_2 is not known to $\mathscr{B}_1$, it cannot create such a key.

Phase 2: As in first phase, $\mathscr{B}_1$ returns a normal key for every query.

Guess: The adversary returns its guess β' to $\mathscr{B}_1$.

If $\beta = \beta'$, then $\mathscr{B}_1$ returns 1 and otherwise returns 0. We have,

$$\begin{aligned}|\Pr[X_{real}] - \Pr[X_0]| &= |\Pr[\beta = \beta' | \mu = 0] = \Pr[\beta = \beta' | \mu \xleftarrow{\text{U}} \mathbb{Z}_p]| \\ &= |\Pr[\mathscr{B}_1 \text{ returns } 1 | \mu = 0] = \Pr[\mathscr{B}_1 \text{ returns } 1 | \mu \xleftarrow{\text{U}} \mathbb{Z}_p]| \\ &= \mathsf{Adv}_{\mathcal{G}}^{\text{DDH1}}(\mathscr{B}_1) \\ &\le \varepsilon_{\text{DDH1}}\end{aligned}$$

□

Lemma 5.5.2. $|\Pr[X_{k-1}] - \Pr[X_k]| \le \varepsilon_{\text{DDH2v}}$.

Proof. Let $(P_1, dP_1, dzP_1, zx_1P_1, P_2, dP_2, x_1P_2, x_2P_2, (x_1x_2 + \gamma)P_2)$ be the instance of DDH2v that $\mathscr{B}_2$ receives.

Setup: $\mathscr{B}_2$ chooses random elements $a, \alpha, \lambda, \nu, y'_v, y_q, y_u, y_w \xleftarrow{\text{U}} \mathbb{Z}_p$ and sets the parameters as follows. $P_1 = P_1, P_2 = P_2$, $Q_2 = -\lambda(dP_2) + y_qP_2$, $U_2 = -\nu(dP_2) + y_uP_2$, $W_2 = dP_2 + y_wP_2$, $V_2 = -a(x_1P_2)$ and $V'_2 = x_1P_2 + y'_vP_2$ setting $\tau = ay'_v$ using which one can compute $\tau P_1 = ay'_vP_1$. The public parameters Q_1, W_1, U_1 can be computed since $\mathscr{B}_2$ has dP_1. The remaining parameters required to provide $\mathcal{PP}$ to $\mathscr{A}$ are computed using a, α and other elements of the problem instance.

Phases 1 and 2: A key extraction query for identity id_i for $i \in [1, q]$ is answered in the following way. For $i < k$, a semi-functional key is returned and for $i > k$ a normal key is returned. Note that normal and semi-functional keys can be generated since $\mathscr{B}_2$ has the $\mathcal{MSK}$ and knows a. For $i = k$, a normal key K'_1, K'_2, K'_3, D' is generated using randomiser $r' \xleftarrow{\text{U}} \mathbb{Z}_p$, $\mathsf{ktag} = \lambda\mathsf{id}_k + \nu$ and then modified as:

$$\begin{aligned}&K_1 = K'_1 - a(x_1x_2 + \gamma)P_2,\ K_2 = K'_2 + (x_1x_2 + \gamma)P_2 + y'_v(x_2P_2),\ K_3 = K'_3 + x_2P_2, \\ &D = D + (y_q\mathsf{id} + y_w\mathsf{ktag} + y_u)(x_2P_2),\end{aligned}$$

thus implicitly setting $r = r' + x_2$. Since dx_2P_2 is not known to $\mathscr{B}_2$ it can create D only when $\mathsf{ktag} = \lambda\mathsf{id}_k + \nu$. If $\mu = 0$ then the key for id_k will be normal and otherwise it will be semi-functional. Note that a semi-functional ciphertext for id_k with any value of ctag except for $\lambda\mathsf{id}_k + \nu$ cannot be generated without the knowledge of dx_1zP_1 which is neither available from the assumption nor can be computed by $\mathscr{B}_2$. This rules out the obvious way of checking whether the key for id_k is semi-functional or not.

Challenge: $\mathscr{B}_2$ receives two messages M_0, M_1 and a challenge identity $\widehat{\mathsf{id}}$ during the challenge phase. It chooses $\beta \xleftarrow{\text{U}} \{0,1\}$, generates a normal ciphertext with randomiser $s' \xleftarrow{\text{U}} \mathbb{Z}_p$, $\widehat{\mathsf{ctag}} = \lambda\widehat{\mathsf{id}} + \nu$ and changes the ciphertext elements as follows. Since λ and ν are chosen independently and uniformly at random, the function $\lambda X + \nu$ is a pairwise independent function for a variable X over $\mathbb{Z}_p$. This causes the tag values of the challenge ciphertext and the k-th key to appear properly distributed from the adversary's view.

$$C_0 = C_0' \cdot e(zx_1P_1, P_2)^{\alpha}$$
$$C_1 = C_1' + zx_1P_1,\ C_2 = C_2' + a(zx_1P_1) + dzP_1,\ C_3 = C_3' - ay_v'(zx_1P_1) + y_w(zx_1P_1) - y_v'(dzP_1),$$
$$E = E' + (y_q\mathsf{id} + \widehat{\mathsf{ctag}}y_w + y_u)(zx_1P_1),$$

setting $s = s' + zx_1$ and $\mu = dz$. It is easy to check that C_3 is well-formed.

$$\begin{aligned}
C_3 &= C_3' - ay_v'(zx_1P_1) + y_w(zx_1P_1) - y_v'(dzP_1)\\
&= -ay_v'(s' + zx_1)P_1 + s'W_1 + y_w(zx_1P_1) + dzx_1P_1 - dzx_1P_1 - y_v'dzP_1\\
&= -ay_v'sP_1 + s'W_1 + zx_1(dP_1 + y_wP_1) - dz(x_1P_1 + y_v'P_1)\\
&= -ay_v'sP_1 + sW_1 - \mu V_1'
\end{aligned}$$

In the event that $\mathscr{A}$ wins the game i.e., $\beta = \beta'$, $\mathscr{B}_2$ returns 1. Otherwise $\mathscr{B}_2$ returns 0.

$$\begin{aligned}
|\Pr[X_{k-1}] - \Pr[X_k]| &= |\Pr[\beta = \beta' \text{ in } \mathsf{G}_{k-1}] = \Pr[\beta = \beta' \text{ in } \mathsf{G}_k]|\\
&= |\Pr[\beta = \beta' | \gamma = 0] = \Pr[\beta = \beta' | \gamma \xleftarrow{\text{U}} \mathbb{Z}_p]|\\
&= |\Pr[\mathscr{B}_2 \text{ returns } 1 | \gamma = 0] = \Pr[\mathscr{B}_2 \text{ returns } 1 | \gamma \xleftarrow{\text{U}} \mathbb{Z}_p]|\\
&= \mathsf{Adv}_{\mathcal{G}}^{\text{DDH2v}}(\mathscr{B}_2)\\
&\leq \varepsilon_{\text{DDH2v}}
\end{aligned}$$

□

Lemma 5.5.3. $|\Pr[X_q] - \Pr[X_{final}]| \leq \varepsilon_{\text{DBDH}}$.

Proof. $\mathscr{B}_3$ receives $(P_1, xP_1, aP_1, sP_1, P_2, xP_2, aP_2, sP_2, e(P_1,P_2)^{xas+c})$ as an instance of the DBDH problem.

Setup: With $y_v, y_v', y_q, y_w, y_u \xleftarrow{\text{U}} \mathbb{Z}_p$, $\mathscr{B}_3$ sets the parameters as

$$P_1 = P_1, P_2 = P_2, aP_1 = aP_1, V_2 = y_vP_2, V_2' = y_v'P_2,\ \tau P_1 = y_vP_1 + y_v'(aP_1)$$

$$Q_1 = y_qP_1, W_1 = y_wP_1, U_1 = y_uP_1, e(P_1,P_2)^{\alpha} = e(xP_1, aP_2)$$

implicitly setting $a = a$, $\alpha = xa$ and $\tau = y_v + ay_v'$. The remaining parameters can be computed easily. $\mathscr{B}_3$ returns $\mathcal{PP}$ to $\mathscr{A}$.

Phases 1 and 2: When $\mathscr{A}$ asks for the secret key for the i'th identity id_i, $\mathscr{B}_3$ samples $r, \mathsf{ktag}, \gamma' \xleftarrow{\text{U}} \mathbb{Z}_p$ implicitly setting $\gamma' = x - \gamma$. It then computes a semi-functional key for id_i as follows.

$$K_1 = \gamma'(aP_2) + rV_2 = xaP_2 - a\gamma P_2 + rV_2 = \alpha P_2 + rV_2 - a\gamma P_2,$$
$$K_2 = rV_2' - \gamma' P_2 + xP_2 = rV_2' - xP_2 + \gamma P_2 + xP_2 = rV_2' + \gamma P_2,$$
$$K_3 = rP_2, D = r(\mathsf{id}_i Q_2 + \mathsf{ktag} W_2 + U_2).$$

Here $\mathscr{B}_3$ knows γ' but not γ. Also, observe that $\mathscr{B}_3$ does not know α and hence cannot create a normal key.

Challenge: $\mathscr{B}_3$ receives the challenge identity $\widehat{\mathsf{id}}$ and two messages M_0 and M_1 from $\mathscr{A}$. It chooses $\beta \xleftarrow{\text{U}} \{0,1\}$ and $\widehat{\mathsf{ctag}}, \mu' \xleftarrow{\text{U}} \mathbb{Z}_p$ and generates a semi-functional challenge ciphertext as follows. Here $\mathscr{B}_3$ implicitly sets $\mu' = \mu + as$ and it does not know μ.

$$C_0 = M_\beta \cdot e(P_1, P_2)^{xas+c},$$
$$C_1 = sP_1,\ C_2 = \mu' P_1 = asP_1 + \mu P_1,$$
$$C_3 = -y_v(sP_1) - \mu' y_v' P_1 + y_w(sP_1) = -y_v sP_1 - asy_v' P_1 - \mu y_v' P_1 + sW_1 = -\tau sP_1 - \mu V_1' + sW_1,$$
$$E = (y_q \widehat{\mathsf{id}} + y_w \widehat{\mathsf{ctag}} + y_u)(sP_1).$$

The challenge ciphertext $\widehat{\mathcal{C}} = (C_0, C_1, C_2, C_3, E, \widehat{\mathsf{ctag}})$ is returned to $\mathscr{A}$. If $c = 0$ then $\widehat{\mathcal{C}}$ will be a semi-functional encryption of M_β; if $c \xleftarrow{\text{U}} \mathbb{Z}_p$, then then $\widehat{\mathcal{C}}$ will be a semi-functional encryption of a random message given by $M \cdot e(P_1, P_2)^c$.

$\mathscr{B}_3$ returns 1 if $\beta = \beta'$; otherwise it returns 0. Hence,

$$\begin{aligned} |\Pr[X_q] - \Pr[X_{final}]| &= |\Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{k-1}] = \Pr[\mathscr{A} \text{ wins in } \mathsf{G}_k]| \\ &= |\Pr[\beta = \beta' \text{ in } \mathsf{G}_{k-1}] = \Pr[\beta = \beta' \text{ in } \mathsf{G}_k]| \\ |\Pr[\beta = \beta' | c = 0] = \Pr[\beta = \beta' | c \xleftarrow{\text{U}} \mathbb{Z}_p]| & \\ &= \mathsf{Adv}_{\mathcal{G}}^{\text{DBDH}}(\mathscr{B}_3) \\ &\le \varepsilon_{\text{DBDH}} \end{aligned}$$

□

5.6 Efficiency Comparisons

A comparison of the features of various IBE schemes based on dual system technique is shown in Tables 5.2 and 5.3. The columns $\#\mathcal{PP}$, $\#\mathcal{MSK}$, #cpr, #key provide the number of group elements in the public parameters, the master secret key, ciphertexts and decryption keys. The public parameters and ciphertexts consist of elements of $\mathbb{G}_1$ while the master secret key and decryption keys consist of elements of $\mathbb{G}_2$. Encryption efficiency counts the number of scalar multiplications in $\mathbb{G}_1$ while decryption efficiency counts the number of pairings that are required. Key generation (a less frequent activity) efficiency is given by the number of scalar multiplications in $\mathbb{G}_2$. $\mathcal{IBE}6$ is the most efficient among all dual system IBE schemes known prior to this work.

The figures in the table indicate that both $\mathcal{IBE}1$ and Lewko's IBE scheme [111] gain a speed up of about 33% over Waters scheme in terms of number of pairings required for decryption. But since $\mathcal{IBE}1$ uses Type-3 pairings, the speed up will in fact be higher. On the other hand, $\mathcal{IBE}6$ is about twice as fast as Lewko-Waters IBE and both constructions are based on Type-3 pairings.

scheme	$\#\mathcal{PP}$	$\#\mathcal{MSK}$	#cpr	#key	enc eff	dec eff	key gen	assump
Waters-09 [158]	13	5	9	8	14	9	12	DLin, DBDH
Lewko-12 [111]	24	30	6	6	24	6	6	DLin
$\mathcal{IBE}1$	9	8	7	6	10	6	9	DDH1, DLin, DBDH

Table 5.2: Comparison of dual system IBE schemes secure under standard assumptions. Waters-09 and Lewko-11 are originally described using symmetric pairings; the figures are for direct conversion to asymmetric pairings. .

scheme	$\#\mathcal{PP}$	$\#\mathcal{MSK}$	#cpr	#key	enc eff	dec eff	key gen	assump
Lewko-Waters [114]	9	6	6	6	9	6	10	LW1, LW2, DBDH
$\mathcal{IBE}6$	6	7	4	4	7	3	6	DDH1, DDH2v, DBDH

Table 5.3: Comparison of dual system IBE schemes secure under non-standard but static assumptions. Note that DDH1 is a weaker assumption than LW1 and DDH2v is weaker than LW2. .

Compared to Waters scheme, the gain in efficiency of decryption is about 66% for $\mathcal{IBE}6$. Again since $\mathcal{IBE}6$ uses Type-3 pairings, the gain in speed will be much higher.

5.7 Extending $\mathcal{IBE}6$ to a HIBE Scheme

The nature of the extension is completely analogous to the way the IBE in [158] has been extended to a HIBE. The setting is of an asymmetric bilinear map $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, F_1, F_2)$. Messages are elements of $\mathbb{G}_T$. Identities are tuples of $\mathbb{Z}_p$ of lengths varying from 1 to some maximum size h. As in [158], we assume that if two identities agree on some component, then they agree on all previous components.

5.7.1 Construction

The HIBE scheme $\mathcal{HIBE}6$ described below is an extension of $\mathcal{IBE}6$. We describe the various functionalities of this scheme.

$\mathcal{HIBE}6$.Setup(κ): Choose generators $P_1 \xleftarrow{\text{U}} \mathbb{G}_1^\times$ and $P_2 \xleftarrow{\text{U}} \mathbb{G}_2^\times$. Let α, a, v, v' be random elements of $\mathbb{Z}_p$. Set $V_2 = vP_2$, $V_2' = v'P_2$ and $\tau = v + av'$ so that $\tau P_2 = V_2 + aV_2'$. Pick $(Q_{1,j}, U_{1,j})_{j=1}^h, W_1 \xleftarrow{\text{U}} \mathbb{G}_1$ and $(Q_{2,j}, U_{2,j})_{j=1}^h, W_2 \longleftarrow \mathbb{G}_2$ such that $\mathsf{dlog}_{P_1}((Q_{1,j}, U_{1,j})_{j=1}^h, W_1) = \mathsf{dlog}_{P_2}((Q_{2,j}, U_{2,j})_{j=1}^h, W_2)$.

$\mathcal{PP}$: $(P_1, aP_1, \tau P_1, Q_{1,1}, \ldots, Q_{1,h}, W_1, U_{1,1}, \ldots, U_{1,h}, e(P_1, P_2)^\alpha)$.
$\mathcal{MSK}$: $(\alpha P_2, P_2, V_2, V_2', Q_{2,1}, \ldots, Q_{2,h}, W_2, U_{2,1}, \ldots, U_{2,h})$.

Actually, only αP_2 is the master secret. The other components of $\mathcal{MSK}$ has to be provided with the decryption key so as to enable further key delegation.

$\mathcal{HIBE}6$.Encrypt$(M, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{PP})$: Choose random $s, \mathsf{ctag}_1, \ldots, \mathsf{ctag}_\ell \xleftarrow{\text{U}} \mathbb{Z}_p$. Ciphertext is given by $\mathcal{C} = (C_0, C_1, C_2, C_3, (E_j, \mathsf{ctag}_j)_{j=1}^\ell)$ where the individual components are defined as follows.

$C_0 = M \cdot e(P_1, P_2)^{\alpha s}$,
$C_1 = sP_1$, $C_2 = asP_1$, $C_3 = -\tau sP_1 + sW_1$,
$E_i = s(\mathsf{id}_i Q_{1,i} + \mathsf{ctag}_i W_1 + U_{1,i})$ for $1 \le i \le \ell$.

$\mathcal{HIBE6}$.KeyGen($\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{MSK}, \mathcal{PP}$): Choose $r_1, \ldots, r_\ell, \mathsf{ktag}_1, \ldots, \mathsf{ktag}_\ell sampU\mathbb{Z}_p$ and set $r = r_1 + \cdots + r_\ell$. $\mathcal{SK}_{\mathbf{id}}$ consists of the elements $(P_2, V_2, V_2', Q_{2,1}, \ldots, Q_{2,h}, W_2, U_{2,1}, \ldots, U_{2,h})$ along with the tuple $(K_1, K_2, (K_{3,j}, D_j, \mathsf{ktag}_j)_{j=1}^{\ell})$ where

$K_1 = \alpha P_2 + rV_2$, $K_2 = rV_2'$
$K_{3,i} = r_i P_2$ for $i = 1, \ldots, \ell$,
$D_i = r_i(\mathsf{id}_i Q_{2,i} + \mathsf{ktag}_i W_2 + U_{2,i})$ for $i = 1, \ldots, \ell$.

The elements $(P_2, V_2, V_2', Q_{2,1}, \ldots, Q_{2,h}, W_2, U_{2,1}, \ldots, U_{2,h})$ are provided to enable further delegation.

$\mathcal{HIBE6}$.Delegate($\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathsf{id}_{\ell+1}, \mathcal{PP}$): Other than the tags, all elements of $\mathcal{SK}_{\mathbf{id}}$ corresponding to identity components at the previous levels are re-randomised. The components required for further delegation are not re-randomised.

Choose random $r_1, \ldots, r_\ell, r_{\ell+1}, \mathsf{ktag}_{\ell+1}$ from $\mathbb{Z}_p$ and set $r = r_1 + \cdots + r_\ell + r_{\ell+1}$. The new components of the key for the identity tuple of length $(\ell + 1)$ are as follows.

$$K_{3,\ell+1} = r_{\ell+1} P_2,\ D_{\ell+1} = r_{\ell+1}(\mathsf{id}_i Q_{2,\ell+1} + \mathsf{ktag}_{\ell+1} W_2 + U_{2,\ell+1}) P_2.$$

Re-randomisation of the previous components is done as follows.

$K_1 \leftarrow K_1 + rV_2$, $K_2 \leftarrow K_2 + rV_2'$
$K_{3,i} \leftarrow K_{3,i} + r_i P_2$ for $i = 1, \ldots, \ell$,
$D_i \leftarrow D_i + r_i(\mathsf{id}_i Q_{2,i} + \mathsf{ktag}_i W_2 + U_{2,i})$ for $i = 1, \ldots, \ell$.

$\mathcal{HIBE6}$.Decrypt($\mathcal{C}, \mathsf{id}, \mathcal{SK}_{\mathsf{id}}, \mathcal{PP}$): Decryption succeeds only when $\mathsf{ctag}_i \ne \mathsf{ktag}_i$ for all $i \in \{1, \ldots, \ell\}$. Let $\vartheta_i = (\mathsf{ctag}_i - \mathsf{ktag}_i)^{-1}$ for $i = 1, \ldots, \ell$. First compute

$$A_1 = \prod_{i=1}^{\ell} e(\vartheta_i E_i, K_{3,i})$$

Then compute

$$A_2 = e\left(C_1, K_1 + \sum_{i=1}^{\ell} \vartheta_i D_i\right) e(C_2, K_2) e\left(C_3, \sum_{i=1}^{\ell} K_{3,i}\right)$$

Unmask the message as $M = (C_0 \cdot A_1)/A_2$.

5.7.2 Security Proof

We first define semi-functional ciphertexts and keys for the HIBE.

$\mathcal{HIBE6}$.SFEncrypt($\mathcal{C}', \mathcal{MSK}$): Let $\mathcal{C}' = (C_0', C_1', C_2', C_3', (E_j', \mathsf{ctag}_j)_{j=1}^{\ell})$ be a ciphertext normally generated by the Encrypt algorithm for message M and identity $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$. Let V_1' be an element of $\mathbb{G}_1$ such that $\mathsf{dlog}_{P_1} V_1' = \mathsf{dlog}_{P_2} V_2'$. Choose $\mu \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. The semi-functional ciphertext generation algorithm will modify the normal ciphertext as: $C_0 = C_0'$, $C_1 = C_1'$, $E_j = E_j'$ for $1 \le j \le \ell$ and

$$C_2 = C_2' + \mu P_1,\ C_3 = C_3' - \mu V_1'.$$

$\mathcal{HIBE6}$.SFKeyGen($\mathcal{SK}'_{\mathbf{id}}, \mathcal{MSK}$): Let $\mathcal{SK}'_{\mathbf{id}} = (K'_1, K'_2, K'_{3,j}, D'_j, \mathsf{ktag}_j)_{j=1}^h)$ be a normal secret key generated by the KeyGen algorithm for identity $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$. Pick $\gamma \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. The semi-functional key generation algorithm will modify the normal key as: $K_{3,j} = K'_{3,j}$, $D_j = D'_j$ for $1 \le j \le \ell$ and

$$K_1 = K'_1 - a\gamma P_2,\ K_2 = K'_2 + \gamma P_2.$$

For the security proof, we follow the path set out by [158]. Suppose that the attacker $\mathscr{A}$ makes at most q_R reveal queries and q_A create and delegate queries. Security is proved via a hybrid argument over a sequence of $q_R + 3$ games. Let G_{real} be the ind-cpa2 HIBE security game defined in Section 2.2.1.2. G_0 is similar to G_{real} except that the challenge ciphertext is semi-functional. In G_k (for $1 \le k \le q_R$), the first k keys are semi-functional and the rest are normal. During the reduction, the keys are changed from normal to semi-functional just before they are revealed. G_{final} is just like G_{q_R} except that the challenge ciphertext is a semi-functional encryption of a random message. Define $X_\square$ as the event that $\mathscr{A}$ wins in $\mathsf{G}_\square$. The following theorem summarizes the security guarantee obtained for $\mathcal{HIBE6}$.

Theorem 5.7.1. *If* $(\varepsilon_{\mathrm{DDH1}}, t_1)$-DDH1, $(\varepsilon_{\mathrm{DDH2v}}, t_2)$-DDH2v *and* $(\varepsilon_{\mathrm{DBDH}}, t_3)$-DBDH *assumptions hold in* $\mathbb{G}_1$, $\mathbb{G}_2$ *and* $\mathbb{G}_T$ *respectively, then* $\mathcal{HIBE6}$ *is* $(\varepsilon, t, q_R, q_A)$-IND-ID-CPA2*-secure where* $\varepsilon \le \varepsilon_{\mathrm{DDH1}} + q_R q_A \cdot \varepsilon_{\mathrm{DDH2v}} + \varepsilon_{\mathrm{DBDH}}$, $t_1 = t + O(q_R h\rho)$, $t_2 = t + O(q_R h\rho)$ *and* $t_3 = t + O(q_R h\rho)$. ρ *is the maximum time required for one scalar multiplication in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$.

Proof. Using lemmas 5.7.1, 5.7.2 and 5.7.3, we have for any polynomial time attacker $\mathscr{A}$,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ind\text{-}cpa2}}_{\mathcal{HIBE6}}(\mathscr{A}) &= |\Pr[X_{real}] - \Pr[X_{final}]| \\ &\le |\Pr[X_{real}] - \Pr[X_0]| + \sum_{k=1}^{q_R} (|\Pr[X_{k-1}] - \Pr[X_k]|) + |\Pr[X_{q_R}] - \Pr[X_{final}]| \\ &\le \varepsilon_{\mathrm{DDH1}} + q_R q_A \cdot \varepsilon_{\mathrm{DDH2v}} + \varepsilon_{\mathrm{DBDH}}\end{aligned}$$

□

In the sequel, let $\mathscr{B}_1$ denote a DDH1-solver, $\mathscr{B}_2$ a DDH2-solver and $\mathscr{B}_3$ is a DBDH-solver.

Lemma 5.7.1. $|\Pr[X_{real}] - \Pr[X_0]| \le \varepsilon_{\mathrm{DDH1}}$.

Proof. The algorithm $\mathscr{B}_1$ receives $(P_1, sP_1, aP_1, P_2, (as+\mu)P_1)$ as an instance of DDH1. We describe how it will simulate each phase in the security game.

Setup: $\mathscr{B}_1$ chooses random elements $\alpha, y_v, y'_v, y_{q_1}, \ldots, y_{q_n}, y_w, y_{u_1}, \ldots, y_{u_n}$ from $\mathbb{Z}_p$ and sets the parameters as follows.

$$P_1 = P_1, aP_1 = aP_1, P_2 = P_2, V_2 = y_v P_2, V'_2 = y'_v P_2$$

$$W_i = y_w P_i, Q_{i,j} = y_{q_j} P_i, U_{i,j} = y_{u_j} P_i \text{ for } i = 1, 2 \text{ and } 1 \le j \le h$$

This implicitly sets $\tau = y_v + a y'_v$. Using this, the element τP_1 can be computed as $y_v P_1 + y'_v(aP_1)$. The simulator computes the remaining parameters using α and gives the following public parameters to $\mathscr{A}$.

$$\mathcal{PP} = \{\mathcal{G}, P_1, P_2, aP_1, \tau P_1, Q_{1,1}, \ldots, Q_{1,h}, W_1, U_{1,1}, \ldots, U_{1,h}, e(P_1, P_2)^\alpha\}$$

Phase 1: $\mathscr{A}$ makes a number of key extract queries. $\mathscr{B}_1$ knows the master secret and using that it returns a normal key for every key extract query made by $\mathscr{A}$.

Challenge: $\mathscr{B}_1$ receives the target identity $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\hat{\ell}})$ and two messages M_0 and M_1 from $\mathscr{A}$. It chooses $\beta \xleftarrow{\text{U}} \{0,1\}$. To encrypt M_β, $\mathscr{B}_1$ chooses $\widehat{\mathsf{ctag}}_1, \ldots, \widehat{\mathsf{ctag}}_\ell \xleftarrow{\text{U}} \mathbb{Z}_p$ and computes the ciphertext elements as follows.

$$\begin{aligned}
&C_0 = M_\beta \cdot e(sP_1, P_2)^\alpha,\\
&C_1 = sP_1,\ C_2 = (as + \mu)P_1,\ C_3 = -y_v(sP_1) - y'_v(as + \mu)P_1 + y_w(sP_1),\\
&E_j = (\widehat{\mathsf{id}}_j y_{q_j} + \widehat{\mathsf{ctag}}_j y_w + y_{u_j})(sP_1) \ \text{ for } \ 1 \le j \le \ell.
\end{aligned}$$

$\mathscr{B}_1$ returns $\widehat{\mathcal{C}} = (C_0, C_1, C_2, C_3, (E_j, \widehat{\mathsf{ctag}}_j)_{j=1}^{h})$ to $\mathscr{A}$.

If $\mu = 0$ then the challenge ciphertext is normal; otherwise $\widehat{\mathcal{C}}$ is semi-functional.

Note that, to check whether $\widehat{\mathcal{C}}$ is semi-functional or not, $\mathscr{B}_1$ itself could try to decrypt it with a semi-functional key for $\widehat{\mathbf{id}}$. However since aP_2 is not known to $\mathscr{B}_1$, it cannot create such a key.

Phase 2: As in first phase, $\mathscr{B}_1$ returns a normal key for every query.

Guess: The adversary returns its guess β' to $\mathscr{B}_1$.

If $\beta = \beta'$, then $\mathscr{B}_1$ returns 1 and otherwise returns 0. We have,

$$\begin{aligned}
|\Pr[X_{real}] - \Pr[X_0]| &= |\Pr[\beta = \beta' | \mu = 0] = \Pr[\beta = \beta' | \mu \xleftarrow{\text{U}} \mathbb{Z}_p]|\\
&= |\Pr[\mathscr{B}_1 \text{ returns } 1 | \mu = 0] = \Pr[\mathscr{B}_1 \text{ returns } 1 | \mu \xleftarrow{\text{U}} \mathbb{Z}_p]|\\
&= \mathsf{Adv}_{\mathcal{G}}^{\text{DDH1}}(\mathscr{B}_1)\\
&\le \varepsilon_{\text{DDH1}}
\end{aligned}$$

□

Lemma 5.7.2. $|\Pr[X_{k-1}] - \Pr[X_k]| \le q_A \cdot \varepsilon_{\text{DDH2v}}$.

Proof. Let $(P_1, dP_1, dzP_1, zx_1P_1, P_2, dP_2, x_1P_1, x_2P_2, (x_1x_2 + \gamma)P_2)$ be the instance of DDH2v that $\mathscr{B}_2$ receives.

Setup: $\mathscr{B}_2$ chooses random elements $a, \alpha, (\lambda_j, \nu_j, y_{q_j}, y_{u_j})_{j=1}^{h}, y'_v, y_w \xleftarrow{\text{U}} \mathbb{Z}_p$ and sets the parameters as follows. $P_1 = P_1, P_2 = P_2$, $Q_{2,j} = -\lambda_j(dP_2) + y_{q_j}P_2$, $U_{2,j} = -\nu_j(dP_2) + y_{u_j}P_2$ for $1 \le j \le h$, $W_2 = dP_2 + y_wP_2$, $V_2 = -a(x_1P_2)$ and $V'_2 = x_1P_2 + y'_vP_2$ setting $\tau = ay'_v$ using which one can compute $\tau P_1 = ay'_vP_1$. The public parameters $Q_{1,1}, \ldots, Q_{1,h}, W_1, U_{1,1}, \ldots, U_{1,h}$ can be computed since $\mathscr{B}_2$ has dP_1. The remaining parameters required to provide $\mathcal{PP}$ to $\mathscr{A}$ are computed using a, α and other elements of the problem instance.

Phases 1 and 2: We first describe how create, delegate and reveal queries are handled. The simulator $\mathscr{B}_2$ chooses $\theta \in \{1, 2, \ldots, q_A\}$ at random and guesses that the k-th key revealed will be the θ-th key either created directly or by delegation. It then creates a counter $\jmath$ for the number of create/delegate queries and initializes it to zero. Recall that S denotes the set of created keys. $\mathscr{B}_2$ now considers two cases.

Case $\jmath \neq \theta$: If this is a create query for an identity **id** of depth ℓ then $\mathscr{B}_2$ chooses tags $\mathsf{ktag}_1, \ldots, \mathsf{ktag}_\ell \xleftarrow{\text{U}} \mathbb{Z}_p$ and associates them along with **id** to the $\jmath$-th member of the set S. Otherwise, if this is a delegate query with inputs **id** of depth $\ell - 1$ and an identity id_ℓ, then the simulator chooses one new tag $\mathsf{ktag}_\ell \xleftarrow{\text{U}} \mathbb{Z}_p$. The other $\ell - 1$ tags $\mathsf{ktag}_1, \ldots, \mathsf{ktag}_{\ell-1}$ are copied from the key we are delegating from. All these tags are associated with the $\jmath$-th element of S.

Case $\jmath = \theta$: If this is a create query for an identity $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ of depth ℓ then $\mathscr{B}_2$ chooses tags $\mathsf{ktag}_1, \ldots, \mathsf{ktag}_{\ell-1} \xleftarrow{\text{U}} \mathbb{Z}_p$, sets $\mathsf{ktag}_\ell = \lambda_\ell \mathsf{id}_\ell + \nu_\ell$ and associates these tags along with **id** to the $\jmath$-th member of the set S. Otherwise, if this is a delegate query with inputs **id** of depth $\ell - 1$ and an identity id_ℓ, then the simulator sets $\mathsf{ktag}_\ell = \lambda_\ell \mathsf{id}_\ell + \nu_\ell$. The other $\ell - 1$ tags $\mathsf{ktag}_1, \ldots, \mathsf{ktag}_{\ell-1}$ are copied from the key we are delegating from. $\mathscr{B}_2$ associates these tags with the $\jmath$-th element of S.

Each element of the set S is now associated with tag values but not a key. This is sufficient since the only elements that are not re-randomized during delegation are the tags. So the keys can be constructed just before revealing. Now consider the $\imath$-th reveal query for $\imath \in \{1, \ldots, q_R\}$ and suppose that this query is for the $\jmath$-th key to be revealed. For $\imath > k$ the simulator has to create a normal key and does so using the master secret and the tag values stored in the $\jmath$-th element of S. For $\imath < k$, $\mathscr{B}_2$ creates a semi-functional key by first creating a normal key using the tag values stored in the $\jmath$-th element of S and then modifies it using its knowledge of aP_2.

Now consider the case when $\imath = k$. If $\jmath \neq k$ then the simulator aborts and makes a random guess of the distribution of γ. Otherwise, using the master secret, it generates a normal key $\mathcal{SK}'_{\mathbf{id}}$ for the identity $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ with components $K'_1, K'_2, (K'_{3,j}, D'_j)_{j=1}^{\ell}$ and tag values already assigned. We know that $\mathsf{ktag}_\ell = \lambda_\ell \mathsf{id}_\ell + \nu_\ell$. Let $r'_1, \ldots, r'_\ell$ be the randomisers used with $r' = r'_1 + \cdots + r'_\ell$. $\mathscr{B}_2$ then chooses $\gamma \in \mathbb{Z}_p$ at random, sets

$$
\begin{aligned}
&K_1 = K'_1 - a(x_1 x_2 + \gamma)P_2,\ K_2 = K'_2 + (x_1 x_2 + \gamma)P_2 + y'_v(x_2 P_2), \\
&K_{3,j} = K'_{3,j},\ D_j = D'_j \text{ for } 1 \le j \le \ell - 1, \\
&K_{3,\ell} = K'_{3,\ell} + x_2 P_2, \\
&D_\ell = D'_\ell + (y_{q_\ell} \mathsf{id}_\ell + y_w \mathsf{ktag}_\ell + y_{u_\ell})(x_2 P_2),
\end{aligned}
$$

thus implicitly setting $r_\ell = r'_\ell + x_2$ and $r = r' + x_2$. It returns the key $\mathcal{SK}_{\mathbf{id}}$ consisting of $K_1, K_2, K_{3,1}, \ldots, K_{3,\ell}, D_1, \ldots, D_\ell, \mathsf{ktag}_1, \ldots, \mathsf{ktag}_\ell$ to $\mathscr{A}$. The choice of $\mathsf{ktag}_\ell = \lambda_\ell \mathsf{id}_\ell + \nu_\ell$ allows $\mathscr{B}_2$ to create D_ℓ. If $Z_2 = x_1 x_2 P_2$ then the key for **id** will be normal and otherwise it will be semi-functional with $\gamma = c$ where $Z_2 = (x_1 x_2 + c)P_2$. Note that a semi-functional ciphertext for **id** with any value of ctag_ℓ cannot be generated without the knowledge of $d x_1 z P_1$ which is neither available from the assumption nor can be computed by $\mathscr{B}_2$. This rules out the obvious way of checking whether the key for id_ℓ is semi-functional or not.

Challenge: $\mathscr{B}_2$ receives two messages M_0, M_1 and a challenge identity $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_\ell)$ during the challenge phase. It chooses $\beta \xleftarrow{\text{U}} \{0, 1\}$, generates a normal ciphertext with randomisers $s' \xleftarrow{\text{U}} \mathbb{Z}_p$ and $\widehat{\mathsf{ctag}}_j = \lambda_j \widehat{\mathsf{id}}_j + \nu_j$ for $1 \le j \le \ell$. Observe that the tags are chosen this way to be able to create the ciphertext elements $E_1, \ldots, E_\ell$. The simulator then changes the ciphertext elements as follows.

$C_0 = C_0' \cdot e(zx_1P_1, P_2)^\alpha$,
$C_1 = C_1' + x_1zP_1$, $C_2 = C_2' + a(zx_1P_1) + dzP_1$, $C_3 = C_3' - ay_v'(zx_1P_1) + y_w(zx_1P_1) - y_v'(dzP_1)$,
$E_j = E_j' + (y_{q_j}\widehat{\mathsf{id}}_j + \widehat{\mathsf{ctag}}_j y_w + y_{u_j})(zx_1P_1)$ for $1 \le j \le \ell$,

setting $s = s' + zx_1$ and $\mu = dz$.

If $\mathscr{A}$ wins i.e., $\beta = \beta'$, then $\mathscr{B}_2$ returns 1 and otherwise returns 0. When $\jmath \neq k$ then the guess is random and the probability that $\mathscr{A}$ wins is 1/2. So we need only consider the event $\jmath = k$ which happens with probability $1/q_A$. We therefore have,

$$\begin{aligned}|\Pr[X_{k-1}] - \Pr[X_k]| &= |\Pr[\beta = \beta'|\gamma = 0] - \Pr[\beta = \beta'|\gamma \xleftarrow{\text{U}} \mathbb{Z}_p]| \\ &= |\Pr[\mathscr{B}_2 \text{ returns } 1|\gamma = 0] - \Pr[\mathscr{B}_2 \text{ returns } 1|\gamma \xleftarrow{\text{U}} \mathbb{Z}_p]| \\ &= \mathsf{Adv}_{\mathcal{G}}^{\text{DLin2}}(\mathscr{B}_2) \\ &\le \varepsilon_{\text{DLin2}}\end{aligned}$$

□

Lemma 5.7.3. $|\Pr[X_{q_R}] - \Pr[X_{final}]| \le \varepsilon_{\text{DBDH}}$.

Proof. $\mathscr{B}_3$ receives $(P_1, xP_1, aP_1, sP_1, P_2, xP_2, aP_2, sP_2, e(P_1, P_2)^{xas+c})$ as an instance of the DBDH problem.

Setup: Sample $b, y_v, y_v', y_{q_1}, \ldots, y_{q_h}, y_w, y_{u_1}, \ldots, y_{u_h} \xleftarrow{\text{U}} \mathbb{Z}_p$, $\mathscr{B}_3$ sets the parameters as

$$P_1 = P_1, P_2 = P_2, aP_1 = aP_1, V_2 = y_vP_2, V_2' = y_v'P_2,\ \tau P_1 = y_vP_1 + y_v'(aP_1)$$

$$Q_{1,j} = y_{q_j}P_1, W_1 = y_wP_1, U_{1,j} = y_{u_j}P_1 \text{ for } 1 \le j \le h, e(P_1, P_2)^{b\alpha} = e(xP_1, aP_2)$$

implicitly setting $a = a$, $\alpha = xa$ and $\tau = y_v + ay_v'$. The remaining parameters can be computed easily. $\mathscr{B}_3$ returns $\mathcal{PP}$ to $\mathscr{A}$.

Phases 1 and 2: When $\mathscr{A}$ asks for the secret key for an identity vector $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, $\mathscr{B}_3$ chooses at random $r_1, \ldots, r_\ell, \mathsf{ktag}_1, \ldots, \mathsf{ktag}_\ell, \gamma' \in \mathbb{Z}_p$ with $r = r_1 + \cdots + r_\ell$ implicitly setting $\gamma' = x - \gamma$. It then computes a semi-functional key for $\mathbf{id}$ as follows.

$K_1 = \gamma'(aP_2) + rV_2 = xaP_2 - a\gamma P_2 + rV_2 = \alpha P_2 + rV_2 - a\gamma P_2$,
$K_2 = rV_2' - \gamma' P_2 = rV_2' + xP_2 + \gamma P_2 = rV_2' + \gamma P_2$,
$K_{3,j} = r_jP_2, D_j = r_j(\mathsf{id}_jQ_{2,j} + \mathsf{ktag}_jW_2 + U_{2,j})$ for $1 \le j \le \ell$.

Observe that $\mathscr{B}_3$ does not know α and hence cannot create a normal key.

Challenge: $\mathscr{B}_3$ receives the challenge identity $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_\ell)$ and two messages M_0 and M_1 from $\mathscr{A}$. It chooses $\beta \in \{0, 1\}$ and $\widehat{\mathsf{ctag}}_1, \ldots, \widehat{\mathsf{ctag}}_\ell, \mu', t \in \mathbb{Z}_p$ at random and generates a semi-functional challenge ciphertext as follows. Here $\mathscr{B}_3$ implicitly sets $\mu' = \mu + as$.

$C_0 = M_\beta \cdot e(P_1, P_2)^{xas+c}$,
$C_1 = sP_1, C_2 = \mu' P_1 = asP_1 + \mu P_1$,
$C_3 = -y_v(sP_1) - \mu' y_v' P_1 + y_w(sP_1) = -y_vsP_1 - asy_v'P_1 - \mu y_v'P_1 + sW_1 = -\tau sP_1 - \mu V_1' + sW_1$,
$E_j = (y_{q_j}\widehat{\mathsf{id}}_j + y_w\widehat{\mathsf{ctag}}_j + y_{u_j})(sP_1)$ for $1 \le j \le \ell$.

The challenge ciphertext $\widehat{\mathcal{C}} = (C_0, C_1, C_2, C_3, E_1, \ldots, E_\ell, \widehat{\mathsf{ctag}}_1, \ldots, \widehat{\mathsf{ctag}}_\ell)$ is returned to $\mathscr{A}$. If $c = 0$ then $\widehat{\mathcal{C}}$ will be a semi-functional encryption of M_β; otherwise $c \overset{\mathrm{U}}{\longleftarrow} \mathbb{Z}_p$ and $\widehat{\mathcal{C}}$ semi-functionally encrypts $Me(P_1, P_2)^c$ which is a random element of $\mathbb{G}_T$.

$\mathscr{B}_3$ returns 1 if $\beta = \beta'$; otherwise it returns 0. Hence,

$$\begin{aligned}
|\Pr[X_q] - \Pr[X_{final}]| &= |\Pr[\mathscr{A} \text{ wins in } \mathsf{G}_q] - \Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{final}]| \\
&= |\Pr[\beta = \beta' \text{ in } \mathsf{G}_q] - \Pr[\beta = \beta' \text{ in } \mathsf{G}_{final}]| \\
&= |\Pr[\beta = \beta' | c = 0] = \Pr[\beta = \beta' | c \overset{\mathrm{U}}{\longleftarrow} \mathbb{Z}_p]| \\
&= \mathsf{Adv}_{\mathcal{G}}^{\mathrm{DBDH}}(\mathscr{B}_3) \\
&\le \varepsilon_{\mathrm{DBDH}}
\end{aligned}$$

□

5.8 Broadcast Encryption

This section discusses Scheme $\mathcal{BE}6$ and its security. Refer to Definition 2.1.3 in Section 2.1.3 for a definition of broadcast encryption and Section 2.2.2 for a description of the corresponding security model.

5.8.1 Construction

$\mathcal{BE}6$.Setup(κ, n): A Type-3 pairing $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, F_1, F_2)$ is generated based on the security parameter κ. Let n be the total number of users and $\mathcal{N} = [1, n]$. Generators $P_1 \overset{\mathrm{U}}{\longleftarrow} \mathbb{G}_1$ and $P_2 \overset{\mathrm{U}}{\longleftarrow} \mathbb{G}_2$ are chosen. Also choose $Q_{1,1}, \ldots Q_{1,n}, W_1 \overset{\mathrm{U}}{\longleftarrow} \mathbb{G}_1$ and $Q_{2,1}, \ldots, Q_{2,n}, W_2 \longleftarrow \mathbb{G}_2$ such that $\mathsf{dlog}_{P_1}(Q_{1,1}, \ldots, Q_{1,n}, W_1) = \mathsf{dlog}_{P_2}(Q_{2,1}, \ldots, Q_{2,n}, W_2)$. Let α, a, v, v' be uniform random elements of $\mathbb{Z}_p$. Set $V_2 = vP_2$, $V_2' = v'P_2$ and $\tau = v + av'$ so that $\tau P_2 = V_2 + aV_2'$.

$\mathcal{PK}$: $(P_1, aP_1, \tau P_1, Q_{1,1}, \ldots, Q_{1,n}, W_1, e(P_1, P_2)^\alpha)$.
$\mathcal{SK}$: $(P_2, \alpha P_2, V_2, V_2', Q_{2,1}, \ldots, Q_{2,n}, W_2)$.

$\mathcal{BE}6$.Encrypt$(\mathcal{PK}, \mathcal{S} \subseteq \mathcal{N}, M)$: Choose random s from $\mathbb{Z}_p$; ciphertext $\mathcal{C}$ for the set $\mathcal{S}$ is (C_0, C_1, C_2, C_3, E) where the elements are defined as follows.

$C_0 = M \cdot e(P_1, P_2)^{\alpha s}$,
$C_1 = sP_1$, $C_2 = asP_1$, $C_3 = -\tau s P_1 + sW_1$, $E = s(\sum_{i \in \mathcal{S}} Q_{1,i})$.

$\mathcal{BE}6$.KeyGen$(\mathcal{SK}, j \in \mathcal{N})$: Choose random r from $\mathbb{Z}_p$; $\mathcal{SK}_j$ is $(K_1, K_2, K_3, D, \forall_{i \ne j} D_i)$ where the elements are defined as follows.

$K_1 = \alpha P_2 + rV_2$, $K_2 = rV_2'$, $K_3 = rP_2$,
$D = r(Q_{2,j} + W_2)$, $D_i = rQ_{2,i}$ for $i \in \mathcal{N}$ and $i \ne j$.

$\mathcal{BE}6$.Decrypt$(\mathcal{C}, \mathcal{S}, \mathcal{SK}_j)$: Decryption works only if $j \in \mathcal{S}$. Unmask the message as

$$M = \frac{C_0}{e(C_1, K_1 - D - \sum_{\substack{i \in \mathcal{S} \\ i \neq j}} D_i) e(C_2, K_2) e(C_3 + E, K_3)}.$$

5.8.2 Security Proof

The semi-functional ciphertexts and keys for the BE are defined as follows.

$\mathcal{BE}6$.SFEncrypt$(\mathcal{PP}, \mathcal{MSK}, \mathcal{C}')$: Let $\mathcal{C}' = (C_0', C_1', C_2', C_3', E)$ be a ciphertext normally generated by the Encrypt algorithm for message M and subset $\mathcal{S} \subseteq \mathcal{N}$ of users. Let V_1' be an element of $\mathbb{G}_1$ such that $V_1' \sim V_2'$. Choose $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$. The semi-functional ciphertext generation algorithm will modify the normal ciphertext as: $C_0 = C_0'$, $C_1 = C_1'$, $E = E'$ and

$$C_2 = C_2' + \mu P_1, \; C_3 = C_3' - \mu V_1'.$$

$\mathcal{BE}6$.SFKeyGen$(\mathcal{PP}, \mathcal{MSK}, \mathcal{SK}_j')$: Let $\mathcal{SK}_j' = (K_1', K_2', K_3', D', (D_i')_{i \in \mathcal{S}, i \neq j})$ be a secret key normally generated by the KeyGen algorithm for user j. The semi-functional key generation algorithm will choose $\gamma \xleftarrow{\text{U}} \mathbb{Z}_p$ and modify the normal key as $K_3 = K_3'$, $D = D'$, $D_i = D_i'$ for all $i \neq j$ and

$$K_1 = K_1' - a\gamma P_2, \; K_2 = K_2' + \gamma P_2.$$

Theorem 5.8.1. *If* $(\varepsilon_{\text{DDH1}}, t_1)$-DDH1, $(\varepsilon_{\text{DDH2v}}, t_2)$-DDH2v *and* $(\varepsilon_{\text{DBDH}}, t_3)$-DBDH *assumptions hold in* $\mathbb{G}_1$, $\mathbb{G}_2$ *and* $\mathbb{G}_T$ *respectively, then* $\mathcal{BE}6$ *is* (ε, t, q)-IND-B-CPA-*secure where* $\varepsilon \le \varepsilon_{\text{DDH1}} + qn \cdot \varepsilon_{\text{DDH2v}} + \varepsilon_{\text{DBDH}}$, $t_1 = t + O(qn\rho)$, $t_2 = t + O(qn\rho)$ *and* $t_3 = t + O(qn\rho)$. ρ *is the maximum time required for one scalar multiplication in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$.

Proof. The proof is organised as a sequence of games. G_{real} denotes the real BE security game ind-be-cpa as defined in Section 2.2.2. G_0 is similar to G_{real} except that the ciphertext encrypted to the challenge set is semi-functional. Suppose that the adversary queries for private keys of q users. In G_k for $1 \le k \le q$, the private keys returned for the first k queries are semi-functional and the rest are normal. G_{final} is just like G_q except that the challenge ciphertext is an encryption of a random message. Clearly, $\Pr[X_{final}] = 1/2$. Let $X_\square$ denote the event that $\mathscr{A}$ wins in $\mathsf{G}_\square$.

Using lemmas 5.8.1, 5.8.2 and 5.8.3, we have for any polynomial time attacker $\mathscr{A}$,

$$\begin{aligned}
\mathsf{Adv}^{\text{ind-be-cpa}}_{\mathcal{BE}6}(\mathscr{A}) &= |\Pr[X_{real}] - \frac{1}{2}| \\
&= |\Pr[X_{real}] - \Pr[X_{final}]| \\
&\le |\Pr[X_{real}] - \Pr[X_0]| + \sum_{k=1}^{q} (|\Pr[X_{k-1}] - \Pr[X_k]|) + |\Pr[X_q] - \Pr[X_{final}]| \\
&\le \varepsilon_{\text{DDH1}} + qn \cdot \varepsilon_{\text{DDH2v}} + \varepsilon_{\text{DBDH}}
\end{aligned}$$

□

Lemma 5.8.1. $|\Pr[X_{real}] - \Pr[X_0]| \le \varepsilon_{\text{DDH1}}$.

Proof. The algorithm $\mathscr{B}_1$ receives $(P_1, sP_1, aP_1, P_2, (as+\mu)P_1)$ as an instance of DDH1. We describe how it will simulate each phase in the security game.

Setup: $\mathscr{B}_1$ chooses random elements $\alpha, y_v, y'_v, y_{q_1}, \ldots, y_{q_n}, y_w \xleftarrow{\text{U}} \mathbb{Z}_p$ and sets the parameters as follows.

$$P_1 = P_1, aP_1 = aP_1, Q_{1,i} = y_{q_i}P_1 \text{ for } i \in [1,n], W_1 = y_wP_1$$

$$P_2 = P_2, V_2 = y_vP_2, V'_2 = y'_vP_2.$$

This implicitly sets $\tau = y_v + ay'_v$. Using this, the element τP_1 can be computed as $y_vP_1 + y'_v(aP_1)$. The simulator computes the remaining parameters using α and gives the public key $\mathcal{PK}$ to $\mathscr{A}$.

Key Extraction Phases 1 and 2: $\mathscr{A}$ issues a number of private key queries adaptively. For every query, $\mathscr{B}_1$ returns a normal key computed using $\mathcal{SK}$.

Challenge: $\mathscr{B}_1$ receives the challenge set $\widehat{\mathcal{S}}$ and two messages M_0 and M_1 from $\mathscr{A}$. It chooses $\beta \xleftarrow{\text{U}} \{0,1\}$ and computes the ciphertext elements as follows.

$C_0 = M_\beta \cdot e(sP_1, P_2)^\alpha$,
$C_1 = sP_1$, $C_2 = (as+\mu)P_1$, $C_3 = -y_v(sP_1) - y'_v(as+\mu)P_1 + y_w(sP_1)$,
$E = (\sum_{i\in\widehat{\mathcal{S}}} y_{q_i})(sP_1)$.

$\mathscr{B}_1$ returns $\widehat{\mathcal{C}} = (C_0, C_1, C_2, C_3, E)$ to $\mathscr{A}$. If $\mu = 0$ then the $\widehat{\mathcal{C}}$ is normal; otherwise $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$ and it is straightforward to verify that $\widehat{\mathcal{C}}$ is semi-functional.

Note that, to check whether $\widehat{\mathcal{C}}$ is semi-functional or not, $\mathscr{B}_1$ itself could try to decrypt it with a semi-functional key for some user $i \in \widehat{\mathcal{S}}$. However since no encoding of a in $\mathbb{G}_2$ is known to $\mathscr{B}_1$, it cannot create such a key.

Guess: The adversary returns its guess β' to $\mathscr{B}_1$.

$\mathscr{B}_1$ returns 1 if $\beta = \beta'$ and otherwise returns 0.

$$\begin{aligned}
|\Pr[X_{real}] - \Pr[X_0]| &= |\Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{real}] - \Pr[\mathscr{A} \text{ wins in } \mathsf{G}_0]| \\
&= |\Pr[\beta = \beta' \text{ in } \mathsf{G}_{real}] - \Pr[\beta = \beta' \text{ in } \mathsf{G}_0]| \\
&= |\Pr[\beta = \beta' | \mu = 0] - \Pr[\beta = \beta' | \mu \xleftarrow{\text{U}} \mathbb{Z}_p]| \\
&= \mathsf{Adv}_{\mathcal{G}}^{\text{DDH1}}(\mathscr{B}_1) \\
&\le \varepsilon_{\text{DDH1}}
\end{aligned}$$

□

Lemma 5.8.2. $|\Pr[X_{k-1}] - \Pr[X_k]| \le n \cdot \varepsilon_{\text{DDH2v}}$.

Proof. Let $(P_1, dP_1, dzP_1, zx_1P_1, P_2, dP_2, x_1P_1, x_2P_2, (x_1x_2+\gamma)P_2)$ be the instance of DDH2v that $\mathscr{B}_2$ receives.

Setup: $\mathscr{B}_2$ chooses random elements $a, \alpha, y'_v, y_{q_1}, \ldots, y_{q_n}, y_w \xleftarrow{\text{U}} \mathbb{Z}_p$ and sets the parameters as follows. $\mathscr{B}_2$ guesses a value $j' \in [1,n]$. $P_1 = P_1, P_2 = P_2$, $Q_{2,j'} = -dP_2 + y_{q_{j'}}P_2$, $Q_{2,i} = y_{q_i}P_2$ for $i \ne j'$, $W_2 = dP_2 + y_wP_2$, $V_2 = -a(x_1P_2)$ and $V'_2 = x_1P_2 + y'_vP_2$ setting $\tau = ay'_v$ using which one can compute $\tau P_1 = ay'_vP_1$. The public parameters $Q_{1,1}, \ldots, Q_{1,n}, W_1$ can be computed since $\mathscr{B}_2$ has dP_1. The remaining parameters required to provide $\mathcal{PK}$ to $\mathscr{A}$ are computed using a, α and other elements of the problem instance.

Private Key Query Phases 1 and 2: The queries before the k-th query are answered with a semi-functional key and for those after k-th query, normal keys are returned. Suppose that the k-th query is for the private key of user j.

When $j \neq j'$, the simulator $\mathscr{B}_2$ returns a normal key to $\mathscr{A}$ and randomly guesses whether $\gamma = 0$ or not. In such a case, the adversary has zero advantage in distinguishing between the games. Also $\mathscr{B}_2$ has zero advantage in solving the DDH2v problem.

Next we consider the case $j = j'$ which happens with probability at least $1/n$. For the k-th query (for user j) a normal key $\mathcal{SK}_j$ with elements $K_1', K_2', K_3', D, \forall_{i \neq j} D_i$ is generated using randomiser $r' \xleftarrow{\text{U}} \mathbb{Z}_p$ and then modified as:

$$\begin{aligned}
&K_1 = K_1' - a(x_1x_2 + \gamma)P_2,\ K_2 = K_2' + (x_1x_2 + \gamma)P_2 + y_v'(x_2P_2),\ K_3 = K_3' + x_2P_2,\\
&D = D' + (y_{q_j} + y_w)(x_2P_2),\ D_i = D_i' + y_{q_i}(x_2P_2) \text{ for all } i \neq j.
\end{aligned}$$

Since $j = j'$,

$$\begin{aligned}
D &= D' + (y_{q_j} + y_w)(x_2P_2)\\
&= r'(Q_{2,j} + W_2) + x_2(y_{q_j} + y_w)P_2\\
&= r'(-dP_2 + y_{q_j}P_2 + dP_2 + y_wP_2) + x_2(y_{q_j} + y_w)P_2\\
&= (r' + x_2)(y_{q_j} + y_w)P_2\\
&= (r' + x_2)(-d + y_{q_j} + d + y_w)P_2\\
&= (r' + x_2)(Q_{2,j} + W_2).
\end{aligned}$$

This implicitly sets $r = r' + x_2$. If $\gamma = 0$ then the key $\mathcal{SK}_j$ will be normal and otherwise it will be semi-functional with γ coming from the instance. Note that a semi-functional ciphertext for any set containing user j cannot be generated without V_1' and thus the obvious way of checking whether $\mathcal{SK}_j$ is semi-functional or not is ruled out.

Challenge: $\mathscr{B}_2$ receives two messages M_0, M_1 and a challenge set $\widehat{\mathcal{S}}$. It chooses $\beta \xleftarrow{\text{U}} \{0,1\}$, generates a normal ciphertext with elements $C_0', C_1', C_2', C_3', E'$ generated using the randomiser $s' \xleftarrow{\text{U}} \mathbb{Z}_p$ and changes the ciphertext elements as follows.

$$\begin{aligned}
&C_0 = C_0' \cdot e(zx_1P_1, P_2)^{\alpha},\\
&C_1 = C_1' + x_1zP_1,\ C_2 = C_2' + a(zx_1P_1) + dzP_1,\ C_3 = C_3' - ay_v'(zx_1P_1) + y_w(zx_1P_1) - y_v'(dzP_1),\\
&E = E' + (\textstyle\sum_{i \in \widehat{\mathcal{S}}} y_{q_i})(zx_1P_1),
\end{aligned}$$

setting $s = s' + zx_1$ and $\mu = dz$. It is easy to check that C_3 is well-formed.

Let abort denote the event that $\mathscr{B}_2$ aborts the game. Also, let Y_{real} and Y_{rand} denote the events that $\mathscr{B}_2$ returns 1 when $\gamma = 0$ and $\gamma \xleftarrow{\text{U}} \mathbb{Z}_p$ respectively. Note that $\mathscr{B}_2$ returns a randomly chosen bit in the event that it aborts the game. We now have

$$\begin{aligned}\mathsf{Adv}_{\mathcal{G}}^{\mathrm{DDH2v}}(\mathscr{B}_2) &= |\Pr[Y_{real}] - \Pr[Y_{rand}]| \\ &= |\Pr[Y_{real}|\mathsf{abort}]\Pr[\mathsf{abort}] + \Pr[Y_{real}|\overline{\mathsf{abort}}]\Pr[\overline{\mathsf{abort}}] \\ &\quad - \Pr[Y_{rand}|\mathsf{abort}]\Pr[\mathsf{abort}] - \Pr[Y_{rand}|\overline{\mathsf{abort}}]\Pr[\overline{\mathsf{abort}}]| \\ &= \left|\frac{1}{2}\left(1-\frac{1}{n}\right) + \Pr[X_{k-1}]\left(\frac{1}{n}\right) - \frac{1}{2}\left(1-\frac{1}{n}\right) - \Pr[X_k]\left(\frac{1}{n}\right)\right| \\ &= \frac{1}{n}\left|\Pr[X_{k-1}] - \Pr[X_k]\right|,\end{aligned}$$

from which the lemma follows. □

Lemma 5.8.3. $|\Pr[X_q] - \Pr[X_{final}]| \le \varepsilon_{\mathrm{DBDH}}$.

Proof. $\mathscr{B}_3$ receives $(P_1, xP_1, aP_1, sP_1, P_2, xP_2, aP_2, sP_2, e(P_1,P_2)^{xas+c})$ as an instance of the DBDH problem.

Setup: With $b, y_v, y_v', y_{q_1}, \ldots, y_{q_n}, y_w$ chosen at random from $\mathbb{Z}_p$, $\mathscr{B}_3$ sets the parameters as

$$P_1 = P_1, P_2 = P_2, aP_1 = aP_1, V_2 = y_vP_2, V_2' = y_v'P_2,\ \tau P_1 = y_vP_1 + y_v'(aP_1)$$

$$Q_{1,1} = y_{q_1}P_1, \ldots, Q_{1,n} = y_{q_n}P_1, W_1 = y_wP_1, e(P_1,P_2)^{b\alpha} = e(xP_1, aP_2)$$

implicitly setting $a = a$, $\alpha = xa$ and $\tau = y_v + ay_v'$. The remaining parameters can be computed easily. $\mathscr{B}_3$ returns $\mathcal{PP}$ to $\mathscr{A}$.

Private Key Query Phases 1 and 2: When $\mathscr{A}$ asks for the secret key for user j, $\mathscr{B}_3$ chooses $r, \gamma' \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ implicitly setting $\gamma' = x - \gamma$. It then computes a semi-functional key for j as follows.

$K_1 = \gamma'(aP_2) + rV_2 = xaP_2 - a\gamma P_2 + rV_2 = \alpha P_2 + rV_2 - a\gamma P_2$,
$K_2 = rV_2' - \gamma' P_2 = rV_2' + xP_2 + \gamma P_2 = rV_2' + \gamma P_2$,
$K_3 = rP_2$, $D = r(Q_{2,j} + W_2)$, $D_i = rQ_{2,i}$ for $i \neq j$.

Observe that $\mathscr{B}_3$ does not know α and hence cannot create a normal key.

Challenge: $\mathscr{B}_3$ receives the challenge set $\widehat{\mathcal{S}}$ and two messages M_0 and M_1 from $\mathscr{A}$. It pick $\beta \xleftarrow{\mathrm{U}} \{0,1\}$ and $\mu' \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and generates a semi-functional challenge ciphertext as follows. Here $\mathscr{B}_3$ implicitly sets $\mu' = \mu + as$.

$C_0 = M_\beta \cdot Z$,
$C_1 = sP_1, C_2 = \mu' P_1 = asP_1 + \mu P_1$,
$C_3 = -y_v(sP_1) - \mu' y_v' P_1 + y_w sP_1 = -y_v sP_1 - asy_v'P_1 - \mu y_v' P_1 + y_w sP_1 = -\tau sP_1 - \mu V_1' + sW_1$,
$E = (\sum_{i \in \widehat{\mathcal{S}}} y_{q,i})(sP_1)$.

The challenge ciphertext $\widehat{\mathcal{C}} = (C_0, C_1, C_2, C_3, E, \widehat{\mathsf{ctag}})$ is returned to $\mathscr{A}$. If $c = 0$ then $\widehat{\mathcal{C}}$ will be a semi-functional encryption of M_β to $\widehat{\mathcal{S}}$; otherwise $\widehat{\mathcal{C}}$ semi-functionally encrypts a random message given by $M_\beta \cdot e(P_1, P_2)^c$. We now have,

$$\begin{aligned} |\Pr[X_q] - \Pr[X_{final}]| &= |\Pr[\beta = \beta' \text{ in } \mathsf{G}_q] - \Pr[\beta = \beta' \text{ in } \mathsf{G}_{final}]| \\ &= |\Pr[\beta = \beta' | c = 0] - \Pr[\beta = \beta' | c \xleftarrow{\text{U}} \mathbb{Z}_p]| \\ &= \mathsf{Adv}_{\mathcal{G}}^{\text{DBDH}}(\mathscr{B}_1) \\ &\leq \varepsilon_{\text{DBDH}} \end{aligned}$$

□

Chapter 6

Constant-Size Ciphertext HIBE

In this section we present three new HIBE schemes that can be implemented using Type-3 pairings – $\mathcal{LW}$-$\mathcal{AHIBE}$, $\mathcal{JR}$-$\mathcal{AHIBE}$ and $\mathcal{JR}$-$\mathcal{HIBE}$ – all achieving constant-sized ciphertexts. The first two are anonymous and the last one is non-anonymous. $\mathcal{LW}$-$\mathcal{AHIBE}$ is obtained via a non-trivial extension of Lewko-Waters Type-3 pairing based IBE scheme [114]. $\mathcal{JR}$-$\mathcal{AHIBE}$ and $\mathcal{JR}$-$\mathcal{HIBE}$ are constructed from the IBE scheme of Jutla and Roy [103]. The literature already contains several different HIBE schemes. So, the question arises as to why new ones are needed? We argue below that previous direct constructions of HIBE schemes had one or more drawbacks related to either efficiency or security. At the time they were proposed, our schemes could overcome most of these issues. Further, $\mathcal{JR}$-$\mathcal{AHIBE}$ and $\mathcal{JR}$-$\mathcal{HIBE}$ are candidates of choice for any practical deployment.

The discussion in Section 1.2 suggests that from an efficiency point of view, HIBE schemes with *constant-size ciphertexts* that can be instantiated with *Type-3 pairings* would offer the best performances.

The first construction for CC-HIBE was given by Boneh, Boyen and Goh [24]. While the scheme itself is quite elegant, its proof of security was in a very restricted attack model, the so-called selective-identity model. This work introduced a way to hash identity vectors into the pairing groups. We refer to this as the BBG-hash. Almost all known CC-HIBE schemes that appeared later have either used this technique or a variant [49, 50, 114, 144, 63, 129, 108, 54]. Since we are interested in CC-HIBE, we do not consider the line of work [86, 22, 157, 47, 158, 136] where the length of the ciphertext depends on the length of the identity tuple. Another method for obtaining constant-size ciphertext HIBE is to specialise constructions of hierarchical inner product encryption. The resulting schemes are not efficient. We comment on this later.

The first construction of anonymous HIBE without random oracles was given by [36] with security in the selective-id model. Later constructions by [144, 63] could achieve security in the adaptive-id setting but were based on composite-order pairings. Two other constructions [70, 129] used asymmetric pairings but with security in the selective-id model.

6.1 Possible Approaches to the Construction of HIBE Schemes.

Scheme $\mathcal{LW}$-$\mathcal{AHIBE}$ was one of the first HIBE schemes to achieve most of the nice provable properties except for standard assumptions. Among the known CC-HIBE schemes, $\mathcal{JR}$-$\mathcal{AHIBE}$ and $\mathcal{JR}$-$\mathcal{HIBE}$ are the most suitable ones for practical deployment.

Extension from IBE. It is quite natural that the construction of a HIBE scheme will be based on an IBE scheme. Below we list other candidate IBE schemes and discuss why their extensions to HIBE schemes do not achieve the same security and efficiency as our constructions.

To start with, it is desirable to avoid a security degradation which is exponential in the depth of the HIBE. In the current state of the art, this means that one has to follow the dual-system approach. So, any attempt to construct a CC-HIBE should start with an IBE which has been proved secure using the dual-system technique. In the dual-system proof technique for both IBE and HIBE, ciphertext and key in the scheme itself are called *normal*. As part of the proof, alternate forms of ciphertext and key are defined. These are called *semi-functional*. In the proof, these are simulated using instances of some hard problem and the argument proceeds by showing that an adversary's ability to distinguish between normal and semi-functional components can be translated into an algorithm to solve the problem. During simulation, it is essential to ensure that all ciphertexts and keys given to the attacker including the semi-functional components (possibly generated using elements from the problem instance) have proper distributions i.e., as in the real construction. This requirement creates the main hurdle in extending the known IBE schemes with dual system proofs to CC-HIBE while retaining the security properties. We discuss this problem below in detail.

The IBE constructions of Waters [158] and its variants [136] do not have a structure that is suitable for extension to CC-HIBE. This is because both the ciphertext and keys have associated *tags* that are public and play a crucial role in dual system arguments. It is precisely these tags that cause the problem in extending these IBEs to CC-HIBEs. While extending to a CC-HIBE, sufficient information should be provided in either the public parameters or the keys to support rerandomisation during key delegation. The tags either cannot be rerandomised or the elements needed to enable their rerandomisation, when given out, lead to insecure schemes.

Lewko and Waters [114] presented a new variant of dual system technique by shifting the role of tags into the semi-functional components. This enabled them to obtain a CC-HIBE scheme over composite order pairing groups. They converted the IBE version of the scheme to the prime-order asymmetric pairing setting (referred to as LW-IBE) but not the HIBE scheme. Security of both their IBE schemes (for composite-order pairings as well as for Type-3 pairings) are based on static but non-standard assumptions.

Our first construction $\mathcal{LW}$-$\mathcal{AHIBE}$ is obtained by extending LW-IBE. $\mathcal{LW}$-$\mathcal{AHIBE}$ is anonymous and achieves security under static assumptions. The only drawback is that the assumptions are non-standard. A parallel work by [108] independently obtained a CC-HIBE scheme from Lewko-Waters' IBE in prime-order groups but with security based on different yet non-standard assumptions.

Another IBE scheme following the dual-system approach is due to Chen et al. [52]. This work uses *dual pairing vector spaces* (DPVSs) [119, 120]. These are algebraic structures that have properties found in composite order groups such as *cancelling* and *parameter-hiding* which are useful for dual system arguments [111]. The Chen et al. IBE can be seen as a translation of Lewko-

Waters' composite-order pairing-based IBE [114] to the setting of asymmetric pairing using DPVS. It is then natural to ask whether the Lewko-Waters composite-order CC-HIBE can be similarly translated using the DPVS-approach to a CC-HIBE. Unfortunately, such a transformation does not yield a CC-HIBE. This is due to the fact that for the proof to work, the dimension of the vector spaces becomes proportional to the HIBE depth. Since ciphertexts contain vectors from such spaces, the constant-size feature cannot be attained.

Our next two HIBE constructions are obtained by extending the IBE scheme of Jutla and Roy [103] (JR-IBE). This IBE is one of the most efficient schemes within the dual system framework achieving security under the standard SXDH assumption.

Chen and Wee [53] introduced new techniques for parameter-hiding in DPVS-based constructions proposed a new CC-HIBE scheme based on these techniques in the full version [55]. However, our approach is different from that of [55]. Moreover, we provide both an anonymous HIBE scheme ($\mathcal{JR}$-$\mathcal{AHIBE}$) and a non-anonymous HIBE scheme ($\mathcal{JR}$-$\mathcal{HIBE}$), whereas [55] provides only a non-anonymous HIBE scheme; and third, compared to the non-anonymous HIBE scheme in [55], $\mathcal{JR}$-$\mathcal{HIBE}$ has smaller ciphertexts and more efficient encryption and decryption algorithms.

Hierarchical Inner-Production Encryption. HIBE schemes can also be seen as special cases of hierarchical inner product encryption (HIPE) schemes. HIBE schemes obtained from two constructions of HIPE schemes by Okamoto and Takashima [122, 124] are comparable to our schemes in terms of provable properties. The scheme in [122] is not anonymous but achieves constant-size ciphertexts. The construction in [124] achieves anonymity and prefix decryption but not constant-size ciphertexts. Note that none of the two HIBE schemes achieve all three properties (anonymity, constant-size ciphertexts and prefix decryption) at the same time. Both schemes are based on dual pairing vector spaces over symmetric pairing groups and are shown to be secure under decisional linear (DLin) assumptions. Although constructions based on DPVSs can be extended to more sophisticated primitives such as attribute-based encryption, a drawback of using this approach is that the ciphertext size depends on the dimension of some DPVS (chosen during system setup). This restricts us from making any further optimisations on the size of the ciphertexts by "transferring" some structure to the keys. Another drawback is as follows. Since the HIBE is an instantiation of (zero) inner-product encryption, the length of the (attribute-)vector consisting of the identity has to be twice the maximum depth (h) of the hierarchy to enable cancellation. Moreover, to accommodate a dual system proof, the dimension of the vector space needs to be at least twice this length i.e., $4h$. This will result in larger keys. Needless to say, we may hope to obtain more efficient HIBE schemes through direct constructions.

Predicate Encryption. As mentioned in Section 3.3, recent works by Wee [161] and Attrapadung [8] provide generic constructions of predicate encryption schemes achieving full security via dual system techniques. Many primitives such as IBE, spatial encryption and inner product encryption can be viewed as specific cases of predicate encryption. In particular, a HIBE scheme is a PE scheme defined by the equality predicate over hierarchical identities. Here, the attributes associated with the ciphertexts and keys are nothing but the hierarchical identities. Both the works use composite order groups for their construction. In principle it should be possible to obtain HIBE schemes in Type-3 setting from these works. Doing this would require specialising a PE scheme to a HIBE scheme and also converting from composite-order setting to the prime-order setting possibly by using the tools from [111, 52]. It is not clear though that the resulting HIBE schemes will have efficiencies comparable to that of $\mathcal{LW}$-$\mathcal{AHIBE}$, $\mathcal{JR}$-$\mathcal{AHIBE}$ or $\mathcal{JR}$-$\mathcal{HIBE}$. We believe that HIBE is an important enough primitive to warrant research on obtaining direct and efficient constructions of such schemes.

6.2 Anonymous HIBE from LW-IBE

The starting point of our work are the IBE schemes in [114]. Two IBE schemes are given in [114] where the first one is in the setting of composite order groups and the second one is in the Type-3 setting. The IBE in the composite order setting is not anonymous (shown in [63]) due to the following reason – the identity-hash in both the ciphertext and key live in the same subgroup; moreover, elements used to create the hash are public thus providing a test for the recipient identity for any ciphertext. On the other hand, the Type-3 variant, which we refer to as "LW-IBE", is anonymous. This is because ciphertexts live in $\mathbb{G}_1$, keys in $\mathbb{G}_2$ and the elements required to create the hash in $\mathbb{G}_2$ are kept secret. Hence there would be no way to test whether a given ciphertext is encrypted to a particular identity or not. However, there has been no proof of anonymity in any follow-up work. The first contribution of the current work is to show that the LW-IBE is anonymous. Two static (though non-standard) computational assumptions (which we denote as LW1 and LW2) along with decision bilinear Diffie-Hellman (DBDH) assumption are used in [114] to show the security of LW-IBE. For proving anonymity, we need to introduce a new computational assumption, called A1, which is again static, but, non-standard.

The second contribution of this paper is to extend the LW-IBE to a constant-size ciphertext HIBE. At a very basic level, the idea for obtaining constant-size ciphertexts is to use the identity hashing technique suggested in [24] over existing IBE schemes. We will refer to this as BBG-hash or BBG-extension. We do not take the path of converting the composite-order pairing based HIBE of [114]. Techniques for such conversions have been proposed by Freeman [73] and Lewko [111]. The latter uses *dual pairing vector spaces* (DPVSs) constructed over pairing groups to simulate features of composite order pairings. But it seems hard to retain the constant size of ciphertexts using these conversion techniques. Instead, we start with LW-IBE and extend it to a CC-HIBE by plugging in the BBG-hash. One complication in doing so arises. In the dual-system technique, two kinds of ciphertexts and keys are defined – *normal* and *semi-functional*. Semi-functional components are required only for proving security and are generated using some secret elements during simulation. The main elements of a dual system proof would be appropriately defining semi-functional components and generating them using a problem instance in the reduction ensuring correct distribution of all elements provided to the attacker. Extending the decryption key of LW-IBE to the decryption key of a HIBE in a straightforward manner does not retain the structure required for a dual-system proof. Our way of tackling this is to add additional components to the decryption key. On the face of it, this complicates the key generation and delegation mechanisms. However, somewhat counter-intuitively, adding this extra level of complication allows the security reductions to go through.

An offshoot of the extension is that the scheme becomes anonymous. This is because in LW-IBE, the semi-functional space (for both ciphertexts and keys) is created using some secret elements (part of the master secret). The same elements are implicitly used in creating ciphertexts and keys.

In case of a direct extension to HIBE, all these elements may have to be revealed in the public parameters to facilitate re-randomisation during delegation of keys. This makes the scheme non-anonymous but at the same time affects dual system arguments for which keeping the elements secret is essential. The way out is to make the scheme anonymous. We also provide a proof of anonymity based on a static assumption.

The computational assumptions required to obtain CPA-security are those used in [114] along with the new assumption required to show that the LW-IBE is anonymous. The last assumption is used to prove the anonymity of the HIBE scheme.

6.2.1 Lewko-Waters IBE and its Security

This section reviews the asymmetric pairing-based IBE construction of Lewko-Waters [114]. The description in [114] consists of the usual ciphertexts and keys as well as the so-called semi-functional ciphertexts and keys. We use a compact notation to denote normal and semi-functional ciphertexts and keys. The group elements shown in curly brackets { } are the semi-functional components. To get the scheme itself, these components should be ignored.

Let $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$ be an asymmetric pairing. Pick $Q_1, U_1 \xleftarrow{\text{U}} \mathbb{G}_1$ and $Q_2, U_2 \in \mathbb{G}_2$ be such that $\mathsf{dlog}_{P_2}(Q_2, U_2) = \mathsf{dlog}_{P_1}(Q_1, U_1)$. Choose $F_2 \xleftarrow{\text{U}} \mathbb{G}_2^{\times}$, $a, v, v' \xleftarrow{\text{U}} \mathbb{Z}_p$ and define $V_2 = vF_2, V_2' = v'F_2$. Let $\tau = v + av'$ so that $\tau F_2 = V_2 + aV_2'$. Identities are elements of $\mathbb{Z}_p$. The public parameters and master secret are given by

$\mathcal{PP}$: $(P_1, aP_1, \tau P_1, Q_1, aQ_1, \tau Q_1, U_1, aU_1, \tau U_1, e(P_1, P_2)^{\alpha})$
$\mathcal{MSK}$: $(\alpha P_2, P_2, V_2, V_2', Q_2, U_2, F_2)$.

The randomisers for the ciphertext and key are s and w, r_1, r_2 respectively. These are elements of $\mathbb{Z}_p$. For the semi-functional components, μ, σ and γ, π are chosen at random from $\mathbb{Z}_p$. Elements $V_1', F_1 \in \mathbb{G}_1$ are such that $\mathsf{dlog}_{P_1}(F_1, V_1') = \mathsf{dlog}_{P_2}(F_2, V_2')$.

Ciphertext:

$C_0 = M \cdot e(P_1, P_2)^{\alpha s}$
$C_{1,1} = s(\mathsf{id}Q_1 + U_1)$, $C_{1,2} = as(\mathsf{id}Q_1 + U_1)\{+\mu\sigma F_1\}$,
$C_{1,3} = -\tau s(\mathsf{id}Q_1 + U_1)\{-\mu\sigma V_1'\}$
$C_{2,1} = sP_1$, $C_{2,2} = asP_1\{+\mu F_1\}$, $C_{2,3} = -\tau s P_1\{-\mu V_1'\}$

Key:

$K_{1,1} = wP_2 + r_1V_2\{-a\gamma F_2\}$, $K_{1,2} = r_1V_2'\{+\gamma F_2\}$, $K_{1,3} = r_1F_2$
$K_{2,1} = \alpha P_2 + w(\mathsf{id}Q_2 + U_2) + r_1V_2\{-a\gamma\pi F_2\}$,
$K_{2,2} = r_2V_2'\{+\gamma\pi F_2\}$, $K_{2,3} = r_2F_2$

Lewko and Waters show that this scheme is adaptively secure without random oracles under three non-standard but static assumptions – LW1, LW2 and DBDH-3. Since the elements Q_2, U_2 are in the master secret there seems to be no way to check whether a given ciphertext is encrypted to a particular identity or not. In other words, this scheme is anonymous. We provide a proof in the ANO-IND-ID-CPA model (described in Section 2.2.1.3) which encompasses both CPA-security and anonymity. For the proof of anonymity, a new static assumption named A1 (defined in Section 2.3.2.1) is introduced. A discussion is provided below.

6.2.2 Discussion on Assumption A1

We introduce assumption A1 to show anonymity of Lewko-Waters' IBE [114] as well as our HIBE scheme $\mathcal{LW}$-$\mathcal{AHIBE}$ described in Section 6.2.3. Given $(\mathcal{G}, F_1, zF_1, dzF_1, azF_1, adzF_1, szF_1, F_2, zF_2, aF_2, xF_2, (dz - ax)F_2)$, the task in A1 is to decide whether $Z_1 = sdzF_1$ or random. Note that the challenge is an element $Z_1 \in \mathbb{G}_1$. Suppose we can successfully create $e(F_1, F_2)^{sdz\delta}$ (for some δ such that δF_2 is given in the instance) using elements in the instance, then the problem becomes easy to solve – just check for equality with $e(Z_1, \delta F_2)$. If they are equal then Z_1 is real; otherwise Z_1 is random. Since s and d appear in separate elements in $\mathbb{G}_1$, the only possible way is to compute $e(Z_1, zF_2)$ and compare it to $e((dz - ax)F_2, szF_1)$ after cancelling out $e(F_1, F_2)^{axsz}$. But this extra term cannot be cancelled since a and x appear in separate elements of $\mathbb{G}_2$. So our assumption is meaningful and there does not seem to be any way of efficiently solving A1.

The problem LW1 contains an embedded instance of DDH1. The elements sF_1 and ab^2F_1 are provided in the instance and it is required to determine whether Y_1 equals ab^2sF_1 or Y_1 is random. Similarly, LW2 contains an embedded instance of DDH2: the elements bF_2 and cF_2 are provided in the instance and it is required to determine whether Y_2 equals bcF_2 or Y_2 is random. As a result, an algorithm to solve DDH1 (resp. LW1) implies an algorithm to solve LW1 (resp. LW2) so that we can say that LW1 (resp. LW2) is no harder than DDH1 (resp. DDH2). The other direction, however, is not clear and it is due to this reason that the assumptions are considered non-standard.

Similar to the above, the problem A1 contains an embedded instance of DDH1. If $P_1 = zF_1$, $P_2 = zF_2$, then the elements $P_1, dP_1, sP_1, Z_1, P_2$ (present in the A1-instance) will form a proper DDH1 instance where it is required to determine whether $Z_1 = sdP_1 = sdzF_1$ or not. Hence a DDH1 solver can be used to solve A1. On the other hand, the converse is not known to hold.

Theorem 6.2.1. *If the* $(\varepsilon_{\mathrm{LW1}}, t')$-LW1*,* $(\varepsilon_{\mathrm{LW2}}, t')$-LW2*,* $(\varepsilon_{\mathrm{DBDH\text{-}3}}, t')$-DBDH-3 *and* $(\varepsilon_{\mathrm{A1}}, t')$-A1 *assumptions hold, then* LW-IBE *is* (ε, t, q)-ANO-IND-ID-CPA *secure where*

$$\varepsilon \leq \varepsilon_{\mathrm{LW1}} + q\varepsilon_{\mathrm{LW2}} + \varepsilon_{\mathrm{DBDH\text{-}3}} + \varepsilon_{\mathrm{A1}}$$

and $t = t' - O(q\rho)$*, where* ρ *is an upper bound on the time required for one scalar multiplication in* $\mathbb{G}_1$ *or* $\mathbb{G}_2$*.*

Proof. Let $\mathscr{A}$ be any t-time adversary against LW-IBE in the ano-ind-cpa. The proof follows a hybrid argument over a sequence of $q+4$ games – $\mathsf{G}_{real}, \mathsf{G}_0, \mathsf{G}_1, \ldots, \mathsf{G}_q, \mathsf{G}_{M\text{-}rand}, \mathsf{G}_{final}$ – between $\mathscr{A}$ and a simulator $\mathscr{B}$, where the games are defined as follows.

- G_{real}: the real security game ano-ind-cpa.
- G_0: challenge ciphertext is semi-functional.
- G_k $(1 \leq k \leq q)$: first k keys returned to the adversary are semi-functional and the rest are normal.

- $\mathsf{G}_{M\text{-}rand}$: the challenge ciphertext encrypts a random message under one of the challenge identities.
- G_{final}: both message and challenge identity are random in the challenge ciphertext.

Let X_{real}, X_k, $X_{M\text{-}rand}$ and X_{final} denote the events that the adversary wins in G_{real}, G_k, $\mathsf{G}_{M\text{-}rand}$ and G_{final} for $0 \leq k \leq q$ respectively. Note that, in G_{final}, the challenge ciphertext is an encryption of a random message under a random identity vector. Hence β is statistically hidden from the adversary's view implying that $\Pr[X_{final}] = 1/2$. From [114], we know that $|\Pr[X_{real}] - \Pr[X_0]| \leq \varepsilon_{\text{LW1}}$, $|\Pr[X_{k-1}] - \Pr[X_k]| \leq \varepsilon_{\text{LW2}}$ and $|\Pr[X_q] - \Pr[X_{M\text{-}rand}]| \leq \varepsilon_{\text{DBDH-3}}$.

We now show that $\Pr[X_{M\text{-}rand}] - \Pr[X_{final}] \leq \varepsilon_{\text{A1}}$. Consider a simulator $\mathscr{B}$ playing the game $\mathsf{ano\text{-}ind\text{-}cpa}$ with $\mathscr{A}$. At this stage all keys are semi-functional and the message encrypted in the challenge ciphertext is random. Let $(\mathcal{G}, F_1, zF_1, dzF_1, azF_1, adzF_1, szF_1, F_2, zF_2, aF_2, xF_2, (dz - ax)F_2, Z_1)$ be the instance of A1 provided to $\mathscr{B}$. Let $Z_1 = c \cdot sdzF_1$. $\mathscr{B}$ has to determine whether $c = 1$ or $c \xleftarrow{\text{U}} \mathbb{Z}_p$. The game is simulated as follows.

Set-Up: Pick $\alpha, v, v', y, u \xleftarrow{\text{U}} \mathbb{Z}_p$ and set the parameters as

$P_1 = zF_1$, $V_2 = vF_2$, $V_2' = v'F_2$, $Q_1 = y(dzF_1)$, $U_1 = u(dzF_1)$,
$aP_1 = azP_1$, $aQ_1 = y(adzF_1)$, $aU_1 = u(adzF_1)$,

Similarly compute the elements τP_1, τQ_1 and τU_1. Compute $e(P_1, P_2)^\alpha = e(zF_1, zF_2)^\alpha$. $\mathscr{B}$ returns $\mathcal{PP}$ to $\mathscr{A}$. $\mathscr{B}$ knows $P_2 = zF_2$ and α but not Q_2 and U_2.

Key Extraction Phases 1 and 2: $\mathscr{B}$ picks $w, r_1, r_2 \xleftarrow{\text{U}} \mathbb{Z}_p$, $\gamma \xleftarrow{\text{U}} \mathbb{Z}_p^\times$ and $\pi' \xleftarrow{\text{U}} \mathbb{Z}_p$. It then computes the key for the k-th identity id_k as follows.

$K_{1,1} = w(zF_2) + r_1V_2 - \gamma aF_2$, $K_{1,2} = r_1V_2' + \gamma F_2$, $K_{1,3} = r_1F_2$
$K_{2,1} = \alpha zF_2 + w(y\mathsf{id}_k + u)(dz - ax)F_2 + r_2V_2 - \gamma\pi'(aF_2)$,
$K_{2,2} = r_2V_2' + w(y\mathsf{id}_k + u)xF_2 + \gamma\pi'F_2$, $K_{2,3} = r_2F_2$,

setting $\pi = \pi' + \gamma^{-1}w(y\mathsf{id}_k + u)x$. Since $\gamma^{-1}w(y\mathsf{id}_k + u)x$ is additively randomised by π', π has the correct distribution in $\mathscr{A}$'s view. $\mathscr{B}$ returns $\mathcal{SK}_{\mathsf{id}_k} = ((K_{1,i}, K_{2,i})_{i=1,2,3})$ to $\mathscr{A}$. The following calculation shows that $K_{2,1}$ and $K_{2,2}$ are well-formed.

The following calculation shows that $K_{2,1}$ and $K_{2,2}$ are well-formed.

$$\begin{aligned}
K_{2,1} &= \alpha zF_2 + w(y\mathsf{id}_i + u)(dz - ax)F_2 + r_2V_2 - \gamma\pi'(aF_2)\\
&= \alpha P_2 + w(y\mathsf{id}_i + u)(dz - ax)F_2 + r_2V_2 - \gamma(\pi - \gamma^{-1}w(y\mathsf{id}_i + u)x)(aF_2)\\
&= \alpha P_2 + w(y\mathsf{id}_i + u)dzF_2 - w(y\mathsf{id}_i + u)axF_2 + r_2V_2 - a\gamma\pi F_2 + w(y\mathsf{id}_i + u)(axF_2)\\
&= \alpha P_2 + w(\mathsf{id}_iQ_2 + U_2) + r_2V_2 - a\gamma\pi F_2\\
K_{2,2} &= r_2V_2' + w(y\mathsf{id}_i + u)xF_2 + \gamma\pi'F_2\\
&= r_2V_2' + w(y\mathsf{id}_i + u)(xF_2) + \gamma(\pi - \gamma_1^{-1}w(y\mathsf{id}_i + u)x)F_2\\
&= r_2V_2' + w(y\mathsf{id}_i + u)(xF_2) + \gamma\pi F_2 - w(y\mathsf{id}_i + u)xF_2\\
&= r_2V_2' + \gamma\pi F_2.
\end{aligned}$$

Challenge: $\mathscr{B}$ receives two pairs of messages and identities $(M_0, \widehat{\mathsf{id}}_0)$ and $(M_1, \widehat{\mathsf{id}}_1)$ from $\mathscr{A}$. It chooses $\beta \xleftarrow{\text{U}} \{0,1\}$ and $a', \xi \xleftarrow{\text{U}} \mathbb{Z}_p$ at random and generates a semi-functional challenge ciphertext as follows.

$C_0 \xleftarrow{\mathrm{U}} \mathbb{G}_T$
$C_{1,1} = (y\widehat{\mathsf{id}}_\beta + u)Z_1,\ C_{1,2} = a'(y\widehat{\mathsf{id}}_\beta + u)Z_1 + \xi F_1,$
$C_{1,3} = -v(y\widehat{\mathsf{id}}_\beta + u))Z_1 - v'a'(y\widehat{\mathsf{id}}_\beta + u)Z_1 - v'\xi F_1,$
$C_{2,1} = szF_1,\ C_{2,2} = a'szF_1,\ C_{2,3} = -v(szF_1) - v'a'(szF_1),$

where $a' = a + \mu'$, $\mu = \mu' sz$ and $\xi = \mu\sigma'$. The challenge ciphertext $\widehat{\mathcal{C}} = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ is returned to $\mathscr{A}$. The computations below illustrate that $\widehat{\mathcal{C}}$ is a semi-functional encryption with $\sigma = \sigma' + cd(y\widehat{\mathsf{id}}_\beta + u)$.

$$\begin{aligned}
C_{1,2} &= a'(y\widehat{\mathsf{id}}_\beta + u)Z_1 + \xi F_1 \\
&= (a + \mu')(y\widehat{\mathsf{id}}_\beta + u)csdzF_1 + \mu\sigma' F_1 \\
&= a(y\widehat{\mathsf{id}}_\beta + u)csdzF_1 + \mu' h(\widehat{\mathbf{id}}_\beta)csdzF_1 + \mu\sigma' F_1 \\
&= as(\widehat{\mathsf{id}}_\beta Q_1 + U_1) + (\mu' sz)(cd(y\widehat{\mathsf{id}}_\beta + u))F_1 + \mu\sigma' F_1 \\
&= as(\widehat{\mathsf{id}}_\beta Q_1 + U_1) + \mu(cd(y\widehat{\mathsf{id}}_\beta + u))F_1 + \mu\sigma' F_1 \\
&= as(\widehat{\mathsf{id}}_\beta Q_1 + U_1) + \mu\sigma F_1
\end{aligned}$$

Observe that $C_{1,1} = s(\widehat{\mathsf{id}}_\beta Q_1 + U_1) = (c \cdot (y\widehat{\mathsf{id}}_\beta + u))(sdzF_1)$. If $c = 1$, then $\sigma = \sigma' + d(y\widehat{\mathsf{id}}_\beta + u)$ and $\widehat{\mathcal{C}}$ is encrypted under $\mathbf{id}_\beta$. Otherwise, c is random, causing $(y\widehat{\mathsf{id}}_\beta + u)$ and consequently the target identity and σ to be random quantities.

Guess: $\mathscr{A}$ returns its guess β' of β.

If the algorithm $\mathscr{B}$ returns 1 when $\beta = \beta'$ and 0 otherwise, it can solve the A1 instance with advantage

$$\begin{aligned}
\mathsf{Adv}_{\mathcal{G}}^{\mathrm{A1}}(\mathscr{B}) &= |\Pr[\beta = \beta' | Z_1 \text{ is real}] - \Pr[\beta = \beta' | Z_1 \text{ is random}]| \\
&= |\Pr[X_{\textit{M-rand}}] - \Pr[X_{\textit{final}}]|.
\end{aligned}$$

Now, $\mathscr{A}$'s advantage in winning the game is given by

$$\begin{aligned}
\mathsf{Adv}_{\mathrm{LW\text{-}IBE}}^{\textsf{ano-ind-cpa}}(\mathscr{A}) &= \left|\Pr[X_{real}] - \frac{1}{2}\right| \\
&= \left|\Pr[X_{real}] - \Pr[X_{final}]\right| \\
&\le |\Pr[X_{real}] - \Pr[X_0]| + \sum_{k=1}^{q} (|\Pr[X_{k-1}] - \Pr[X_k]|) \\
&\quad + |\Pr[X_{q,1}] - \Pr[X_{\textit{M-rand}}]| + |\Pr[X_{\textit{M-rand}}] - \Pr[X_{final}]| \\
&\le \varepsilon_{\mathrm{LW1}} + q\varepsilon_{\mathrm{LW2}} + \varepsilon_{\text{DBDH-3}} + \varepsilon_{\mathrm{A1}}
\end{aligned}$$

□

6.2.3 Extension to the Hierarchical Setting

In this section, we present our HIBE scheme, $\mathcal{LW\text{-}AHIBE}$, resulting from a BBG-type extension of the LW IBE scheme. A straightforward BBG-type extension would lead to problems in adopting the dual system methodology. We introduce some new elements to overcome this problem. The construction is based on a Type-3 prime-order pairing with group order p. Identities are variable length tuples of elements from $\mathbb{Z}_p^\times$ with maximum length h.

The first step towards obtaining constant-size ciphertexts is to add elements $(Q_{1,j})_{j\in[1,h]}, U_1 \in \mathbb{G}_1$ to the public parameters. These are used to create the identity hash – for an identity $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, the hash is given by $\sum_{j=1}^{\ell} \mathsf{id}_j Q_{1,j} + U_1$. This replaces the hash in LW-IBE without affecting the number of elements in the ciphertext. To facilitate key extraction, the corresponding elements in $\mathbb{G}_2$ also are provided. We introduce some notation here: the tuple $(P_1, (Q_{1,j})_{j\in[1,h]}, U_1)$ is denoted $\mathbf{Q}_1$ and let its $\mathbb{G}_2$ counterpart be $\mathbf{Q}_2$. Also present in the master secret of LW-IBE are the elements V_2, V_2', F_2 that provide cancellation analogous to the composite order setting. In the HIBE setting, these elements along with $\mathbf{Q}_2$, must be made public to assist in re-randomisation during delegation. Once these are made public, nothing is kept secret except for α. This acts as a stumbling block against a dual system proof. In a proof within the dual system framework, some secret elements are needed to create the so-called semi-functional components that are central to this proof methodology. In the composite order setting, this is achieved by keeping one subgroup hidden from the attacker which essentially forms the semi-functional space. Similarly, schemes based on dual pairing vector spaces have some vectors in the dual bases hidden that assist in generating the semi-functional space. But the strategy for HIBE extension of LW-IBE chalked out above, requires everything to be made public (except α), which in turn limits our ability to define a semi-functional space.

Our solution to this problem is to keep $\mathbf{Q}_2$ in the master secret. In a way, some elements of the group $\mathbb{G}_2$ are hidden and provide the basis for generating semi-functional components. To support delegation, suitably randomised copies of the key components are provided in the key itself. This technique was introduced by Boyen and Waters [36] to construct an anonymous HIBE scheme. V_2, V_2', F_2 are public to help in re-randomisation during delegation; this ensures proper distribution of the delegated key. Note that $\mathbf{Q}_2$ contains precisely the elements required to check whether a ciphertext is encrypted to a particular identity or not. A by-product of keeping this tuple secret is anonymity. Thus our scheme is secure in the ANO-IND-ID-CPA security model (refer to Section 2.2.1.3).

We now present the scheme $\mathcal{LW}\text{-}\mathcal{AHIBE}$. A discussion on the security of $\mathcal{LW}\text{-}\mathcal{AHIBE}$ can be found in Section 6.2.4.

Construction

$\mathcal{LW}\text{-}\mathcal{AHIBE}$.Setup($\kappa$): Let h denote the maximum depth of the HIBE. Choose random generators $P_1 \in \mathbb{G}_1$ and $P_2 \in \mathbb{G}_2$; elements $Q_{1,1}, \ldots, Q_{1,h}, U_1 \xleftarrow{\text{U}} \mathbb{G}_1$ and $Q_{2,1}, \ldots, Q_{2,h}, U_2 \in \mathbb{G}_2$ such that $Q_{2,j} \sim Q_{1,j}$ for all $1 \le j \le h$ and $U_2 \sim U_1$. Let $F_2 \in \mathbb{G}_2$ be chosen at random and v, v' be chosen randomly from $\mathbb{Z}_p$. Set $V_2 = vF_2$, $V_2' = v'F_2$. Pick α, a at random from $\mathbb{Z}_p$. Set $\tau = v + av'$ so that $\tau F_2 = V_2 + aV_2'$.

$\mathcal{PP}$: $(P_1, aP_1, \tau P_1, U_1, aU_1, \tau U_1, (Q_{1,j}, aQ_{1,j}, \tau Q_{1,j})_{j\in[1,h]},$
$V_2, V_2', F_2, e(P_1, P_2)^\alpha)$.

$\mathcal{MSK}$: $(\alpha P_2, P_2, Q_{2,1}, \ldots, Q_{2,h}, U_2)$.

$\mathcal{LW}\text{-}\mathcal{AHIBE}$.Encrypt($M, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{PP}$): Choose $s \xleftarrow{\text{U}} \mathbb{Z}_p$. Let $\mathcal{H}_i(\mathbf{id}) = \mathsf{id}_1 Q_{i,1} + \cdots + \mathsf{id}_\ell Q_{i,\ell} +$

U_i for $i = 1, 2$. The ciphertext is given by $\mathcal{C} = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ where the elements are computed as follows.

$C_0 = M \times e(P_1, P_2)^{\alpha s}$,
$C_{1,1} = s\mathcal{H}_1(\mathbf{id})$, $C_{1,2} = as\mathcal{H}_1(\mathbf{id})$, $C_{1,3} = -\tau s\mathcal{H}_1(\mathbf{id})$
$C_{2,1} = sP_1$, $C_{2,2} = asP_1$, $C_{2,3} = -\tau sP_1$

$\mathcal{LW}$-$\mathcal{AHIBE}$.KeyGen($\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{MSK}, \mathcal{PP}$): Choose $w_1, w_2, r_1, r_2, r_3, r_4, (z_{1,j}, z_{2,j})_{j\in[\ell+1,h]} \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. The key consists of $6(n - \ell + 2)$ group elements computed as follows.

$K_{1,1} = w_1P_2 + r_1V_2$, $K_{1,2} = r_1V_2'$, $K_{1,3} = r_1F_2$
$K_{2,1} = \alpha P_2 + w_1\mathcal{H}_2(\mathbf{id}) + r_2V_2$, $K_{2,2} = r_2V_2'$, $K_{2,3} = r_2F_2$
$D_{j,1} = w_1Q_{2,j} + z_{1,j}V_2$, $D_{j,2} = z_{1,j}V_2'$, $D_{j,3} = z_{1,j}F_2$ for $\ell + 1 \le j \le h$

$J_{1,1} = w_2P_2 + r_3V_2$, $J_{1,2} = r_3V_2'$, $J_{1,3} = r_3F_2$
$J_{2,1} = w_2\mathcal{H}_2(\mathbf{id}) + r_4V_2$, $J_{2,2} = r_4V_2'$, $J_{2,3} = r_4F_2$
$E_{j,1} = w_2Q_{2,j} + z_{2,j}V_2$, $E_{j,2} = z_{2,j}V_2'$, $E_{j,3} = z_{2,j}F_2$ for $\ell + 1 \le j \le h$.

The secret key for $\mathbf{id}$ is given by $\mathcal{SK}_{\mathbf{id}} = (\mathcal{S}_1, \mathcal{S}_2)$, where $\mathcal{S}_1 = (K_{1,i}, K_{2,i}, D_{j,i})_{j\in[\ell+1,h],i=1,2,3}$ and $\mathcal{S}_2 = (J_{1,i}, J_{2,i}, E_{j,i})_{j\in[\ell+1,h],i=1,2,3}$. Notice that $\mathcal{S}_2$-components are almost same as $\mathcal{S}_1$-components except that the secret α is not embedded in $\mathcal{S}_2$. The set $\mathcal{S}_2$ is exclusively used for re-randomisation.

$\mathcal{LW}$-$\mathcal{AHIBE}$.Delegate($\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathsf{id}_{\ell+1}, \mathcal{PP}$): Let $\mathbf{id} : \mathsf{id}_{\ell+1}$ denote the $\ell + 1$-length identity vector $(\mathsf{id}_1, \ldots, \mathsf{id}_\ell, \mathsf{id}_{ell+1})$ obtained by appending $\mathsf{id}_{\ell+1}$ to $\mathbf{id}$. Choose $r_1', r_2', r_3', r_4', (z_{1,j}', z_{2,j}')_{j\in[\ell+2,h]} \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and $w_1', w_2' \xleftarrow{\mathrm{U}} \mathbb{Z}_p^\times$. The components of the key for the identity $\mathbf{id} : \mathsf{id}_{\ell+1}$ are computed as follows.

$K_{1,1} \leftarrow K_{1,1} + w_1'J_{1,1} + r_1'V_2$ $\quad$ $K_{2,1} \leftarrow K_{2,1} + \mathsf{id}_{\ell+1}D_{\ell+1,1} + w_1'(J_{2,1} + \mathsf{id}_{\ell+1}E_{\ell+1,1}) + r_2'V_2$
$K_{1,2} \leftarrow K_{1,2} + w_1'J_{1,2} + r_1'V_2'$ $\quad$ $K_{2,2} \leftarrow K_{2,2} + \mathsf{id}_{\ell+1}D_{\ell+1,2} + w_1'(J_{2,2} + \mathsf{id}_{\ell+1}E_{\ell+1,2}) + r_2'V_2'$
$K_{1,3} \leftarrow K_{1,3} + w_1'J_{1,3} + r_1'F_2$ $\quad$ $K_{2,3} \leftarrow K_{2,3} + \mathsf{id}_{\ell+1}D_{\ell+1,3} + w_1'(J_{2,3} + \mathsf{id}_{\ell+1}E_{\ell+1,3}) + r_2'F_2$

$J_{1,1} \leftarrow w_2'J_{1,1} + r_3'V_2$ $\quad$ $J_{2,1} \leftarrow w_2'(J_{2,1} + \mathsf{id}_{\ell+1}E_{\ell+1,1}) + r_4'V_2$
$J_{1,2} \leftarrow w_2'J_{1,2} + r_3'V_2'$ $\quad$ $J_{2,2} \leftarrow w_2'(J_{2,2} + \mathsf{id}_{\ell+1}E_{\ell+1,2}) + r_4'V_2'$
$J_{1,3} \leftarrow w_2'J_{1,3} + r_3'F_2$ $\quad$ $J_{2,3} \leftarrow w_2'(J_{2,3} + \mathsf{id}_{\ell+1}E_{\ell+1,3}) + r_4'F_2$

For $j = \ell + 2, \ldots, h$,
$D_{j,1} \leftarrow D_{j,1} + w_1'E_{j,1} + z_{1,j}'V_2$ $\quad$ $D_{j,2} \leftarrow D_{j,2} + w_1'E_{j,2} + z_{1,j}'V_2'$ $\quad$ $D_{j,3} \leftarrow D_{j,3} + w_1'E_{j,3} + z_{1,j}'F_2$
$E_{j,1} \leftarrow w_2'E_{j,1} + z_{2,j}'V_2$ $\quad$ $E_{j,2} \leftarrow w_2'E_{j,2} + z_{2,j}'V_2'$ $\quad$ $E_{j,3} \leftarrow w_2'E_{j,3} + z_{2,j}'F_2$

The above procedure essentially re-randomises all components of the key. As a result the distribution of a key obtained using delegation is the same as the distribution of a key obtained using the key generation procedure. To note the re-randomisation consider the following change of scalars for the modified key.

$w_1 \leftarrow w_1 + w_1'w_2$; $\quad$ $w_2 \leftarrow w_2'w_2$;
$r_1 \leftarrow r_1 + r_1' + w_1'r_3$; $\quad$ $r_3 \leftarrow w_2'r_3 + r_3'$;
$r_2 \leftarrow r_2 + r_2' + \mathsf{id}_{\ell+1}z_{1,\ell+1} + w_2'(r_4 + \mathsf{id}_{\ell+1}z_{2,\ell+1})$; $\quad$ $r_4 \leftarrow w_2'(r_4 + \mathsf{id}_{\ell+1}z_{2,\ell+1}) + r_4'$;
$z_{1,j} \leftarrow z_{1,j} + z_{1,j}' + w_1'z_{2,j+1}$ for $j = \ell + 2, \ldots, h$ $\quad$ $z_{2,j} \leftarrow w_2'z_{2,j} + z_{2,j}'$ for $j = \ell + 2, \ldots, h$

These new randomisers are properly distributed by the choice of $w_1', w_2', r_1', r_2', r_3', r_4', (z_{1,j}'), (z_{2,j}')$.

$\mathcal{LW}\text{-}\mathcal{AHIBE}.\mathsf{Decrypt}(\mathcal{C}, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathcal{PP})$: Decryption is done as follows.

$$M = C_0 \times \frac{e(C_{1,1}, K_{1,1})e(C_{1,2}, K_{1,2})e(C_{1,3}, K_{1,3})}{e(C_{2,1}, K_{2,1})e(C_{2,2}, K_{2,2})e(C_{2,3}, K_{2,3})} \tag{6.1}$$

Correctness of decryption of the HIBE scheme follows directly from that of LW-IBE since the decryption procedure remains the same – the additional delegation components do not play any role in decryption. Observe that computing the ratio of pairings in Equation (6.1) using $J_{1,i}, J_{2,i}$ ($i = 1, 2, 3$) instead of $K_{1,i}, K_{2,i}$ results in 1_T (the identity of $\mathbb{G}_T$).

6.2.4 Security of $\mathcal{LW}\text{-}\mathcal{AHIBE}$

We first provide some basic intuition underlying the proof with respect to different stages of security analysis (within the dual system framework), highlighting the similarities and differences with LW-IBE security proof. Then, a detailed security analysis of $\mathcal{LW}\text{-}\mathcal{AHIBE}$ is presented in Section 6.2.4.2.

6.2.4.1 Ideas Underlying the Security Proof

The first step is to define semi-functional (sf) ciphertexts and keys. The definition of sf-ciphertext remains the same as LW-IBE. The keys of $\mathcal{LW}\text{-}\mathcal{AHIBE}$ are significantly different from LW-IBE. We formulate the definition of sf-keys on the basis of the following observations.

- Sf-components for $(K_{1,i}, K_{2,i})_{i=1,2}$ are identical to LW-IBE since only these components participate in decryption.
- It is required to define sf-components for $(D_{j,1}, D_{j,2})_{j\in[\ell+1,h]}$ though they are only used during delegation to create the identity-hash. This is because they share the randomiser w_1 with $K_{1,i}, K_{2,i}$ and this randomiser comes from a problem instance in the reductions.
- Once sf-components are defined for $\mathcal{S}_1$, it is natural to ask: is it necessary to define sf-parts for $\mathcal{S}_2$? The answer is yes since otherwise the fourth reduction fails, where $P_2, U_2, (Q_{2,j})$ are masked by a quantity that forces the keys to be semi-functional. We have already seen this in the context of LW-IBE (see Theorem 6.2.1).

We would like to emphasise that the definition of semi-functional components (in both ciphertexts and keys), complexity assumptions and the reductions are all inter-linked. Changing the structure of sf-keys may determine the assumption required or affect simulation in some reduction. Also, for the reductions to go through, the sf-components may have to be defined in a particular way. The structure of sf-components we have is in a sense, optimal, subject to assumptions and simulations we provide.

An outline of the four main reductions in the augmented security proof (including anonymity) of LW-IBE is as follows.

First reduction: The goal of this reduction is to show that an attacker cannot distinguish between a normal ciphertext and an sf-ciphertext. It is achieved via a reduction from the LW1 problem. An LW1 instance is embedded in the challenge ciphertext attempting to exploit the adversary's ability to detect the change in order to solve the problem.

Second reduction: In this reduction, it is shown that if the adversary can decide whether the response to the k-th key extraction query is normal or semi-functional, then LW2 problem can be solved. The k-th key is constructed from an instance of LW2 problem in such a way that the key is normal if the instance is 'real' and semi-functional otherwise.

Third reduction: Here, the message that the challenge ciphertext encrypts, is changed to a random element of $\mathbb{G}_T$. It is shown that solving the DBDH-3 problem is no harder than distinguishing between an sf-encryption of the real message from an sf-encryption of a random element of $\mathbb{G}_T$.

Fourth reduction: Challenge ciphertext encrypts a random message under a random identity. The identity-hash is created using the challenge in an instance of A1 problem thus making it real or random according to the distribution of the challenge.

This strategy does not directly extend to the hierarchical setting. Several challenges emerge as we try to prove security of $\mathcal{LW}$-$\mathcal{AHIBE}$.

The first and the third reductions for $\mathcal{LW}$-$\mathcal{AHIBE}$ are the closest to the corresponding reductions for LW-IBE appearing in [114]. In these reductions, the simulations of the public parameters; the ciphertext elements; and the components of the key which are present in LW-IBE; are exactly the same as for LW-IBE. The only technicality is to ensure that the extra components of the key can be properly simulated without changing the corresponding assumptions (LW1 for the first reduction and DBDH-3 for the third reduction).

The second reduction presents some technical novelty. We need to extend the dual-system technique to handle this reduction. In this reduction, it is shown that the adversary cannot decide whether the response to the k-th key extraction query is normal or semi-functional. Compared to the LW-IBE, the key has additional components which are required for delegation and re-randomisation; moreover, these have semi-functional parts. A new technique is required to handle these simulations.

Partial semi-functionality: Consider the second reduction where the k-th key is made semi-functional. LW-IBE reduction embeds a pairwise independent function in the k-th key as well as the challenge ciphertext to ensure independent distribution of the scalars involved in the respective sf-components. This function is determined by the parameters used to create the identity-hash. An attempt to use the same strategy for $\mathcal{LW}$-$\mathcal{AHIBE}$, however, causes a problem. The reason is that the identity-hash is now present in three places – challenge ciphertext, $\mathcal{S}_1$ and $\mathcal{S}_2$. In addition, all these have sf-components. One possible way to deal with this is to embed a 3-wise independent function i.e., a degree-2 polynomial in the identity. As result the one extra group element is required in $\mathcal{PP}$ as well as $\mathcal{MSK}$. Also, encryption and key generation would each require an extra scalar multiplication and a squaring in the underlying field. The other way to get around the problem is to use two separate instances to generate the two hashes in the key. We follow the latter approach since the efficiency of the scheme remains unaffected although the degradation is increased by a factor of 2. The key is changed from normal to semi-functional in two steps – first make $\mathcal{S}_1$ semi-functional followed by $\mathcal{S}_2$. We call a key partial semi-functional if $\mathcal{S}_1$ is semi-functional and $\mathcal{S}_2$ is normal. Also, let $\mathsf{PSFKeyGen}$ denote the algorithm that generates a partial semi-functional key.

The second step of the dual-system technique changes the key in the k-th response from normal to semi-functional (without the adversary noticing this). In our case, this is done in two sub-steps – the first step changes from normal to partial semi-functional and the second step changes from partial semi-functional to semi-functional. This leads to a slight degradation in the security bound by a factor of 2.

The fourth reduction is to show anonymity of the HIBE scheme. This is almost the same as the reduction that we have provided to show the anonymity of the LW-IBE. The only difference is that the extra elements of the key have to properly simulated.

Discussion. It is natural to ask whether it is at all required to define semi-functional terms for $\mathcal{S}_2$ components of a key that do not play any role in decryption. The answer is yes and the reason is as follows. Since all the elements required to create the id-hash in $\mathbb{G}_2$ are hidden, there is no way to test the identity to which a ciphertext is encrypted. The scheme seems to be anonymous but to prove it, we need to ensure that a semi-functional encryption to a target identity is indistinguishable from a semi-functional encryption to a random identity vector. (We need semi-functionality in order to deal with the key extraction queries.)

Normally, the K-components of the key are used for decrypting a ciphertext. When these are paired with the ciphertext components we obtain the blinding factor for the message that only depends on α and the randomiser s. Instead if we try decrypting using J-components of the key (which do not have any α terms), we get 1_T, the identity of $\mathbb{G}_T$. Hence the J-components help in testing whether the ciphertext is indeed encrypted under **id** or not. The presence of such a test does not help in proving anonymity property. Therefore, it is essential to make $\mathcal{S}_2$-components of all keys semi-functional before arguing about anonymity.

6.2.4.2 Detailed Proof

As is typical in the dual-system technique, we first describe semi-functional ciphertexts and keys. These are required only in the reductions and not in the actual scheme.

$\mathcal{LW}$-$\mathcal{AHIBE}$.$\mathsf{SFEncrypt}(\mathcal{PP}, \mathcal{MSK}, \mathcal{C}')$: Let $\mathcal{C}' = (C'_0, C'_{1,1}, C'_{1,2}, C'_{1,3}, C'_{2,1}, C'_{2,2}, C'_{2,3})$ be a ciphertext normally generated by the $\mathsf{Encrypt}$ algorithm for message M and identity **id**. Let V'_1, F_1 be elements of $\mathbb{G}_1$ such that $V'_1 \sim V'_2$ and $F_1 \sim F_2$. Choose $\mu, \sigma \xleftarrow{\text{U}} \mathbb{Z}_p$. Modify the normal ciphertext as: $C_0 = C'_0$, $C_{1,1} = C'_{1,1}$, $C_{2,1} = C'_{2,1}$ and

$$C_{1,2} = C'_{1,2} + \mu\sigma F_1,\ C_{1,3} = C'_{1,3} - \mu\sigma V'_1, C_{2,2} = C'_{2,2} + \mu F_1,\ C_{2,3} = C'_{2,3} - \mu V'_1.$$

$\mathcal{LW}$-$\mathcal{AHIBE}$.$\mathsf{SFKeyGen}(\mathcal{PP}, \mathcal{MSK}, \mathcal{SK}'_{\mathbf{id}})$: Let $\mathcal{SK}'_{\mathbf{id}} = (\mathcal{S}_1, \mathcal{S}_2)$ be the secret key generated by the KeyGen algorithm for identity $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ with $\mathcal{S}_1 = (K_{1,i}, K_{2,i}, D_{j,i})_{j\in[\ell+1,h], i=1,2,3}$, $\mathcal{S}_2 = (J_{1,i}, J_{2,i}, E_{j,i})_{j\in[\ell+1,h], i=1,2,3}$. Let $\gamma_1, \pi, \gamma_2, \eta, (\pi_j, \eta_j)_{j\in[\ell+1,h]} \xleftarrow{\text{U}} \mathbb{Z}_p$. The semi-functional key generation algorithm will modify the normal key as:

$$\begin{aligned} &K_{1,1} = K_{1,1} - a\gamma_1 F_2,\ K_{1,2} = K_{1,2} + \gamma_1 F_2, && J_{1,1} = J_{1,1} - a\gamma_2 F_2,\ J_{1,2} = J_{1,2} + \gamma_2 F_2,\\ &K_{2,1} = K_{2,1} - a\gamma_1 \pi F_2,\ K_{2,2} = K_{2,2} + \gamma_1 \pi F_2, && J_{2,1} = J_{2,1} - a\gamma_2 \eta F_2,\ J_{2,2} = J_{2,2} + \gamma_2 \eta F_2,\end{aligned}$$

For $j = \ell + 1, \ldots, h$

$$D_{j,1} = D_{j,1} - a\gamma_1 \pi_j F_2,\ D_{j,2} = D_{j,2} + \gamma_1 \pi_j F_2, \qquad E_{j,1} = E_{j,1} - a\gamma_2 \eta_j F_2,\ E_{j,2} = E_{j,2} + \gamma_2 \eta_j F_2.$$

The rest of the components remain unchanged.

$\mathcal{LW}$-$\mathcal{AHIBE}$.$\mathsf{PSFKeyGen}(\mathcal{PP}, \mathcal{MSK}, \mathcal{SK}'_{\mathbf{id}} = (\mathcal{S}_1, \mathcal{S}_2))$: In a partial semi-functional key, $\mathcal{S}_2$ is normal and $\mathcal{S}_1$ is semi-functional.

Note that definitions are similar to [114] except for the delegation and re-randomisation components. Since decryption is not affected by these components of the key, all the requirements for semi-functional keys and ciphertexts are satisfied. A pair of semi-functional ciphertext and key is called *nominally semi-functional* if $\sigma = \pi$ (condition that makes decryption successful).

Structure of the Proof. We consider the security model defined in Section 2.2.1.3. The proof is organised as a hybrid over a sequence of $2q + 4$ games defined as follows.

G_{real}: $\mathsf{ano\text{-}ind\text{-}cpa}$ game defined in Section 2.2.1.3.

$\mathsf{G}_{0,1}$: the challenge ciphertext is semi-functional and all the keys returned to the adversary are normal.

$\mathsf{G}_{k,0}$ (for $1 \le k \le q$): k-th key is partial semi-functional, the first $k-1$ keys are semi-functional; the rest of the keys are normal.

$\mathsf{G}_{k,1}$ (for $1 \le k \le q$): similar to $\mathsf{G}_{k,0}$ except that the k-th key is (fully) semi-functional.

$\mathsf{G}_{M\text{-}rand}$: all keys are semi-functional and the challenge ciphertext encrypts a random message to the challenge identity.

G_{final}: similar to $\mathsf{G}_{M\text{-}rand}$ except that the challenge ciphertext now encrypts to a random identity vector.

These games are ordered as G_{real}, $\mathsf{G}_{0,1}$, $\mathsf{G}_{1,0}, \mathsf{G}_{1,1}, \ldots, \mathsf{G}_{q,0}, \mathsf{G}_{q,1}$, $\mathsf{G}_{M\text{-}rand}$, G_{final} in our hybrid argument. Let $X_\square$ be events that $\mathscr{A}$ wins in $\mathsf{G}_\square$.

For the proof it will be convenient to use the following short-hand: denote by $h(\mathbf{id})$ the sum $\sum_{j=1}^{\ell} y_j \mathsf{id}_j + u$ and by $g(\mathbf{id})$ the sum $\sum_{j=1}^{\ell} \lambda_j \mathsf{id}_j + \nu$, where $y_1, \ldots, y_n, u, \lambda_1, \ldots, \lambda_n, \nu$ are elements of $\mathbb{Z}_p$ to be chosen in the proofs.

Theorem 6.2.2. *If the* $(\varepsilon_{\text{LW1}}, t')$-LW1, $(\varepsilon_{\text{LW2}}, t')$-LW2, $(\varepsilon_{\text{DBDH-3}}, t')$-DBDH-3 *and* $(\varepsilon_{\text{A1}}, t')$-A1 *assumptions hold, then* $\mathcal{LW}$-$\mathcal{AHIBE}$ *is* (ε, t, q)-$\mathsf{ANO\text{-}IND\text{-}ID\text{-}CPA}$ *secure where*

$$\varepsilon \le \varepsilon_{\text{LW1}} + 2q\varepsilon_{\text{LW2}} + \varepsilon_{\text{DBDH-3}} + \varepsilon_{\text{A1}}$$

and $t = t' - O(q\rho)$, *where* ρ *is an upper bound on the time required for one scalar multiplication in* $\mathbb{G}_1$ *or* $\mathbb{G}_2$.

Proof. For any t-time adversary $\mathscr{A}$ against $\mathcal{LW}$-$\mathcal{AHIBE}$ in the $\mathsf{ano\text{-}ind\text{-}cpa}$, its advantage in winning the game is given by

$$\mathsf{Adv}^{\mathsf{ano\text{-}ind\text{-}cpa}}_{\mathcal{LW}\text{-}\mathcal{AHIBE}}(\mathscr{A}) = \left|\Pr[X_{real}] - \frac{1}{2}\right|.$$

We know that $\Pr[X_{final}] = \frac{1}{2}$ and hence we have

$$\begin{aligned}
\mathsf{Adv}^{\text{ano-ind-cpa}}_{\mathcal{LW}\text{-}\mathcal{AHIBE}}(\mathscr{A}) &= \left|\Pr[X_{real}] - \Pr[X_{final}]\right| \\
&\leq |\Pr[X_{real}] - \Pr[X_0]| + \sum_{k=1}^{q}\left(|\Pr[X_{k-1,1}] - \Pr[X_{k,0}]|\right) + \sum_{k=1}^{q}\left(|\Pr[X_{k,0}] - \Pr[X_{k,1}]|\right) \\
&\quad + |\Pr[X_{q,1}] - \Pr[X_{M\text{-}rand}]| + |\Pr[X_{M\text{-}rand}] - \Pr[X_{final}]| \\
&\leq \varepsilon_{\text{LW1}} + 2q\varepsilon_{\text{LW2}} + \varepsilon_{\text{DBDH-3}} + \varepsilon_{\text{A1}}
\end{aligned}$$

The last inequality follows from the lemmas 6.2.1, 6.2.2, 6.2.3, 6.2.4 and 6.2.5. In all the lemmas, $\mathscr{A}$ is a t-time adversary against $\mathcal{LW}\text{-}\mathcal{AHIBE}$ and $\mathscr{B}$ is an algorithm running in time t' that interacts with $\mathscr{A}$ and solves one of the three problems LW1, LW2, DBDH-3 or A1. □

Lemma 6.2.1. $|\Pr[X_{real}] - \Pr[X_{0,1}]| \leq \varepsilon_{\text{LW1}}$.

Proof. The algorithm $\mathscr{B}$ receives the following instance of LW1

$$(F_1, bsF_1, sF_1, aF_1, ab^2F_1, bF_1, b^2F_1, asF_1, b^2sF_1, b^3F_1, b^3sF_1, F_2, bF_2, Z_1).$$

$\mathscr{B}$ has to determine whether $Z_1 = ab^2sF_1$ or $Z_1 \xleftarrow{\text{U}} \mathbb{G}_1$. We will call Z_1 "real" in the former case and "random" otherwise. $\mathscr{B}$ simulates the security game as described below.

Set-Up: $\mathscr{B}$ chooses $\alpha, y, v', (y_j, \lambda_j)_{j\in[1,h]}, u, \nu \xleftarrow{\text{U}} \mathbb{Z}_p$ and sets the parameters.

$$P_1 = b^2F_1 + yF_1, Q_{1,j} = \lambda_j(b^2F_1) + y_jF_1 \text{ for } 1 \leq j \leq h, U_1 = \nu(b^2F_1) + uF_1$$

$$V_2 = bF_2, V_2' = v'F_2.$$

This implicitly sets $P_2 = (b^2 + y)F_2$, $v = b$ and $\tau = b + av'$. Compute $aP_1 = ab^2F_1 + y(aF_1)$ and $\tau P_1 = b^3F_1 + v'(ab^2F_1) + y(bF_1) + yv'(aF_1)$. The elements $(aQ_{1,j}, \tau Q_{1,j})_{j\in[1,h]}, aU_1, \tau U_1$ are constructed similarly. Set $e(P_1, P_2)^\alpha = (e(b^3F_1 + y(bF_1), bF_2)e(P_1, yF_2))^\alpha$. The simulator gives the following public parameters to $\mathscr{A}$.

$$\mathcal{PP} = (P_1, Q_{1,1}, \ldots, Q_{1,h}, U_1, aP_1, aQ_{1,1}, \ldots, aQ_{1,h}, aU_1, \tau P_1, \tau Q_{1,1}, \ldots, \tau Q_{1,h}, \tau U_1, e(P_1, P_2)^\alpha).$$

Phases 1 and 2: $\mathscr{A}$ makes a number of key extract queries. $\mathscr{B}$ does not know $P_2, Q_{2,j}, U_2$ which are part of the master secret. The secret key for a query on **id** is constructed as follows. $\mathscr{B}$ chooses $r_1', r_2', r_3', r_4', (z_{1,j}', z_{2,j}')_{j\in[\ell+1,h]}, w_1, w_2 \in \mathbb{Z}_p$ at random and computes

$K_{1,1} = w_1yF_2 + r_1'(bF_2),\ K_{1,3} = r_1'F_2 - w(bF_2),\ K_{1,2} = v'K_{1,3},,$
$K_{2,1} = \alpha yF_2 + r_2'(bF_2) + wh(\mathbf{id})F_2,\ K_{2,3} = r_2'F_2 - (w_1g(\mathbf{id}) + \alpha)(bF_2),\ K_{2,2} = v'K_{2,3},,$
$D_{j,1} = w_1y_jF_2 + z_{1,j}'(bF_2),\ D_{j,3} = z_{1,j}'F_2 - w_1\lambda_j(bF_2),\ D_{j,2} = v'D_{j,3}$ for $\ell + 1 \leq j \leq h$,

$J_{1,1} = w_2yF_2 + r_3'(bF_2),\ J_{1,3} = r_3'F_2 - w_2(bF_2),\ J_{1,2} = v'J_{1,3},$
$J_{2,1} = r_4'(bF_2) + w_2h(\mathbf{id})F_2,\ J_{2,3} = r_4'F_2 - w_2g(\mathbf{id})(bF_2),\ J_{2,2} = v'J_{2,3},$
$E_{j,1} = w_2y_jF_2 + z_{2,j}'(bF_2),\ E_{j,3} = z_{2,j}'F_2 - w_2\lambda_j(bF_2),\ E_{j,2} = v'E_{j,3}$ for $\ell + 1 \leq j \leq h$,

implicitly setting

$r_1 = r'_1 - w_1 b,\ r_2 = r'_2 - (w_1 g(\mathbf{id}) + \alpha)b,$
$r_3 = r'_3 - w_2 b,\ r_4 = r'_4 - w_2 g(\mathbf{id})b,$
$z_{1,j} = z'_{1,j} - w_1\lambda_j b,\ z_{2,j} = z'_{2,j} - w_2\lambda_j b$ for $j = \ell+1, \ldots, h$.

The following computation shows that the components are well-formed.

$$\begin{aligned} K_{1,1} &= wyF_2 + r'_1(bF_2) \\ &= wyF_2 + (r_1 + wb)bF_2 \\ &= w(yF_2 + b^2F_2) + r_1(bF_2) \\ &= wP_2 + r_1V_2 \end{aligned} \qquad \begin{aligned} K_{2,1} &= \alpha yF_2 + r'_2(bF_2) + wh(\mathbf{id})F_2 \\ &= \alpha yF_2 + (r_2 + (wg(\mathbf{id}) + \alpha)b)bF_2 + wh(\mathbf{id})F_2 \\ &= \alpha(yF_2 + b^2F_2) + r_2(bF_2) + w(g(\mathbf{id})b^2F_2 + h(\mathbf{id})F_2) \\ &= \alpha P_2 + w\mathscr{H}_2(\mathbf{id}) + r_2V_2 \end{aligned}$$

$$\begin{aligned} D_{1,j} &= wy_jF_2 + z'_j(bF_2) \\ &= wy_jF_2 + (z_{1,j} + w\lambda_j b)(bF_2) \\ &= w(y_jF_2 + \lambda_j b^2F_2) + z_{1,j}(bF_2) \\ &= wQ_{2,j} + z_{1,j}V_2 \end{aligned}$$

Following the same logic, it can be verified that $J_{1,1}, J_{2,1}, E_{j,1}$ are well-formed. Remaining components clearly have the right form.

Challenge: $\mathscr{B}$ receives two pairs $(M_0, \widehat{\mathbf{id}}_0)$ and $(M_1, \widehat{\mathbf{id}}_1)$ from $\mathscr{A}$. It chooses $\beta \in \{0,1\}$ at random. $\mathscr{B}$ computes the ciphertext for M_β under $\widehat{\mathbf{id}}_\beta$ as follows.

$C_0 = M_\beta \cdot \left(e(b^3sF_1 + y(bsF_1), bF_2)e(b^2sF_1 + y(sF_1), yF_2)\right)^\alpha = M_\beta \cdot e(P_1, P_2)^{\alpha s}$
$C_{1,1} = g(\widehat{\mathbf{id}}_\beta)(b^2sF_1) + h(\widehat{\mathbf{id}}_\beta)(sF_1),\ C_{1,2} = g(\widehat{\mathbf{id}}_\beta)Z_1 + h(\widehat{\mathbf{id}}_\beta)(asF_1)$
$C_{1,3} = -g(\widehat{\mathbf{id}}_\beta)(b^3sF_1) - h(\widehat{\mathbf{id}}_\beta)(bsF_1) - v'g(\widehat{\mathbf{id}}_\beta)Z_1 - v'h(\widehat{\mathbf{id}}_\beta)(asF_1)$
$C_{2,1} = b^2sF_1 + y(sF_1),\ C_{2,2} = Z_1 + y(asF_1)$
$C_{2,3} = -b^3sF_1 - y(bsF_1) - v'Z_1 - v'(asF_1).$

$\mathscr{B}$ returns $\widehat{\mathcal{C}} = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ to $\mathscr{A}$.

If $Z_1 = ab^2sF_1$, then it is easy to see that $\widehat{\mathcal{C}}$ is normal. Otherwise, $Z_1 = (ab^2s + \mu)F_1$ for some $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$ and $\widehat{\mathcal{C}}$ is semi-functional with $\mu = \mu$ and $\sigma = g(\widehat{\mathbf{id}}_\beta)$. The calculation below shows that $C_{1,2}$ is a properly formed (semi-functional) component.

$$\begin{aligned} C_{1,2} &= g(\widehat{\mathbf{id}}_\beta)Z_1 + h(\widehat{\mathbf{id}}_\beta)(asF_1) \\ &= g(\widehat{\mathbf{id}}_\beta)(ab^2s + \mu)F_1 + h(\widehat{\mathbf{id}}_\beta)(asF_1) \\ &= as(g(\widehat{\mathbf{id}}_\beta)b^2F_1 + h(\widehat{\mathbf{id}}_\beta)F_1) + \mu g(\widehat{\mathbf{id}}_\beta)F_1 \\ &= as\mathscr{H}_1(\widehat{\mathbf{id}}_\beta) + \mu\sigma F_1 \end{aligned}$$

Verification of the well-formedness of $C_{1,3}$, $C_{2,2}$ and $C_{2,3}$ follows the same pattern. Scalars $(\lambda_j)_{j\in[1,h]}, \nu$ are information theoretically hidden from $\mathscr{A}$'s view and hence $\sigma = g(\mathbf{id})$ appears to be uniformly and independently distributed with respect to all other information provided to $\mathscr{A}$.

Note that, to check whether $\widehat{\mathcal{C}}$ is semi-functional or not, $\mathscr{B}$ itself could try to decrypt it with a semi-functional key for $\widehat{\mathbf{id}}_\beta$. Any such attempt will fail due to the following reason – aF_2 is unavailable to $\mathscr{B}$; it could try to cancel out $-a\gamma F_2$ in $K_{1,1}$ or γF_2 in $K_{1,2}$ with some other elements;

but we do not see how to achieve this keeping the link between $K_{1,1}$ and $K_{1,2}$ (via γ) intact, without knowing aF_2.

Guess: The adversary returns its guess β' to $\mathscr{B}$.

If Z_1 is real, $\widehat{\mathcal{C}}$ is normal and hence $\mathscr{B}$ simulates G_{real}. Otherwise, Z_1 is random and $\widehat{\mathcal{C}}$ is semi-functional in which case, $\mathscr{B}$ simulates $\mathsf{G}_{0,1}$. Suppose that $\mathscr{B}$ returns 1 if $\beta = \beta'$ and 0 otherwise. Then it can solve the LW1 problem with advantage

$$\mathsf{Adv}_{\mathcal{G}}^{\mathrm{LW1}}(\mathscr{B}) = |\Pr[\beta = \beta' | Z_1 \text{ is real}] - \Pr[\beta = \beta' | Z_1 \text{ is random}]| = |\Pr[X_{real}] - \Pr[X_{0,1}]| \le \varepsilon_{\mathrm{LW1}}.$$

□

Lemma 6.2.2. $|\Pr[X_{k-1,1}] - \Pr[X_{k,0}]| \le \varepsilon_{\mathrm{LW2}}$ *for* $1 \le k \le q$.

Proof. Let $(F_1, dF_1, d^2F_1, bxF_1, dbxF_1, d^2xF_1, F_2, dF_2, bF_2, cF_2, Z_2)$ be the instance of LW2 that $\mathscr{B}$ receives. Let $Z_2 = (bc + \gamma)F_2$. $\mathscr{B}$'s task is to decide whether $\gamma = 0$ (Z_2 is real) or $\gamma \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ (Z_2 is random).

Set-Up: $\mathscr{B}$ chooses $\alpha, a, y_v, y_1, \dots, y_h, u, \lambda_1, \dots, \lambda_h, \nu \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and computes parameters as follows. $P_1 = dF_1$, $Q_{1,j} = \lambda_j(dF_1) + y_jF_1$ for $1 \le j \le h$, $U_1 = \nu(dF_1) + uF_1$, $V_2 = -a(bF_2) + dF_2 + y_vF_2$ and $V_2' = bF_2$ setting $v = -ab + d + y_v$, $v' = b$ and $\tau = d + y_v$. The element τP_1 can be computed as $\tau P_1 = d^2F_1 + y_v(dF_1)$. The parameters $\tau Q_{1,j}$ for $1 \le j \le h$ and τU_1 are given by $\tau Q_{1,j} = \lambda_j(d^2F_1) + y_j(dF_1) + y_v\lambda_j(dF_1) + y_vy_jF_1$ and $\tau U_1 = \nu(d^2F_1) + u(dF_1) + y_v\nu(dF_1) + y_vuF_1$. The remaining parameters required to provide $\mathcal{PP}$ to $\mathscr{A}$ are computed using a, α and elements of the problem instance. Elements of the master secret key can also be obtained from the instance and randomisers chosen at setup.

Phases 1 and 2: The key extraction queries for identities $\mathbf{id}_1, \dots, \mathbf{id}_q$ are answered in the following way. If $i < k$, a semi-functional key is returned and if $i > k$ a normal key is returned. $\mathscr{B}$ creates semi-functional keys using the master secret, a and F_2.

For $i = k$, $\mathscr{B}$ computes of $\mathcal{S}_1$ using the problem instance in the following manner. Let $\mathbf{id}_k = (\mathsf{id}_1, \dots, \mathsf{id}_\ell)$. $\mathscr{B}$ chooses $w_1', r_2', z_{1,\ell+1}', \dots, z_{1,h}' \xleftarrow{\mathrm{U}} \mathbb{Z}_p$.

$K_{1,1} = w_1'P_2 - aZ_2 + y_v(cF_2)$, $K_{1,2} = Z_2$, $K_{1,3} = cF_2$
$K_{2,1} = \alpha P_2 + w_1'(g(\mathbf{id}_k)(dF_2) + h(\mathbf{id}_k)F_2) + r_2'V_2 - ag(\mathbf{id}_k)Z_2 + y_vg(\mathbf{id}_k)(cF_2) - h(\mathbf{id}_k)cF_2$
$K_{2,2} = r_2'V_2' + g(\mathbf{id}_k)Z_2$, $K_{2,3} = r_2'F_2 + g(\mathbf{id}_k)(cF_2)$

and for $j = \ell + 1, \dots, h$, set

$D_{j,1} = w_1'Q_{2,j} + z_{1,j}'V_2 - y_j(cF_2) - a\lambda_jZ_2 + y_v\lambda_j(cF_2)$
$D_{j,2} = z_{1,j}'V_2' + \lambda_jZ_2$, $D_{j,3} = z_{1,j}'F_2 + \lambda_j(cF_2)$

thus implicitly setting $w_1 = w_1' - c$, $r_1 = c$, $r_2 = r_2' + g(\mathbf{id}_k)c$ and $z_{1,j} = z_{1,j}' + \lambda_jc$ for $\ell + 1 \le j \le h$.

Let $\mathcal{S}_1 = (K_{1,i}, K_{2,i}, D_{j,i})_{j\in[\ell+1,h],i=1,2,3}$. The second set $\mathcal{S}_2 = (J_{1,i}, J_{2,i}, E_{j,i})_{j\in[\ell+1,h],i=1,2,3}$ is created normally. $\mathscr{B}$ returns $\mathcal{SK}_{\mathbf{id}_k} = (\mathcal{S}_1, \mathcal{S}_2)$ as the key for $\mathbf{id}_k$. If $Z_2 = bcF_2$ then the key for $\mathbf{id}_k$ is normal.

We show that $K_{2,1}$ is well-formed. Verifying the remaining parts can be done analogously.

$$\begin{aligned}
K_{2,1} &= \alpha P_2 + w_1'(g(\mathbf{id}_k)(dF_2) + h(\mathbf{id}_k)F_2) + r_2'V_2 - ag(\mathbf{id}_k)Z_2 + y_v g(\mathbf{id}_k)(cF_2) - h(\mathbf{id}_k)cF_2 \\
&= \alpha P_2 + (w_1 + c)\mathcal{H}_2(\mathbf{id}_k) + (r_2 - g(\mathbf{id}_k)c)(-a(bF_2) + dF_2 + y_v F_2) \\
&\qquad + g(\mathbf{id}_k)(-abcF_2 + y_v cF_2) - h(\mathbf{id}_k)cF_2 \\
&= \alpha P_2 + (w_1 + c)\mathcal{H}_2(\mathbf{id}_k) + r_2 V_2 - g(\mathbf{id}_k)cdF_2 + g(\mathbf{id}_k)(abcF_2 - cy_v F_2) \\
&\qquad - g(\mathbf{id}_k)(abcF_2 - y_v cF_2) - h(\mathbf{id}_k)cF_2 \\
&= \alpha P_2 + (w_1 + c)\mathcal{H}_2(\mathbf{id}_k) + r_2 V_2 - g(\mathbf{id}_k)cdF_2 - h(\mathbf{id}_k)cF_2 \\
&= \alpha P_2 + (w_1 + c)\mathcal{H}_2(\mathbf{id}_k) + r_2 V_2 - c\mathcal{H}_2(\mathbf{id}_k) \\
&= \alpha P_2 + w_1 \mathcal{H}_2(\mathbf{id}_k) + r_2 V_2.
\end{aligned}$$

If $Z_2 = (bc + \gamma)F_2$ the key will be partial semi-functional with $\gamma_1 = \gamma$, $\pi = g(\mathbf{id}_k)$ and $\pi_j = \lambda_j$ for $\ell + 1 \le j \le h$. It is straightforward to check that $\mathcal{SK}_{\mathbf{id}_k}$ is a properly formed partial sf-key. Also, since $(\lambda_j)_{j\in[1,h]}, \nu$ are information theoretically hidden from the adversary, $\pi, (\pi_j)_{j\in[\ell+1,h]}$ are uniformly and independently distributed in $\mathscr{A}$'s view.

$\mathscr{B}$ could attempt checking whether $\mathcal{SK}_{\mathbf{id}_k}$ is semi-functional by creating a sf-ciphertext for $\mathbf{id}_k$. Since $V_1' = bF_1$ is not available to $\mathscr{B}$, the only way of doing this will lead to σ being the same as π (challenge ciphertext is created via this method). The ciphertext-key pair will be nominally semi-functional and thus provides no information to $\mathscr{B}$.

Challenge: $\mathscr{A}$ provides two message-identity pairs, $(M_0, \widehat{\mathbf{id}}_0)$ and $(M_1, \widehat{\mathbf{id}}_1)$ to $\mathscr{B}$. It chooses $\beta \xleftarrow{\text{U}} \{0,1\}$, generates the challenge ciphertext as shown below.

$C_0 = M_\beta \cdot e(dbxF_1, dF_2)^\alpha$
$C_{1,1} = g(\widehat{\mathbf{id}}_\beta)(dbxF_1) + h(\widehat{\mathbf{id}}_\beta)(bxF_1)$
$C_{1,2} = ag(\widehat{\mathbf{id}}_\beta)(dbxF_1) + ah(\widehat{\mathbf{id}}_\beta)(bxF_1) - g(\widehat{\mathbf{id}}_\beta)(d^2xF_1)$
$C_{1,3} = -y_v g(\widehat{\mathbf{id}}_\beta)(dbxF_1) - h(\widehat{\mathbf{id}}_\beta)(dbxF_1) - y_v h(\widehat{\mathbf{id}}_\beta)(bxF_1)$
$C_{2,1} = dbxF_1$, $C_{2,2} = a(dbxF_1) - d^2xF_1$, $C_{2,3} = -y_v(dbxF_1)$.

This sets $s = bx$, $\mu = -d^2x$ and $\sigma = g(\widehat{\mathbf{id}}_\beta)$. Since $\lambda_1, \ldots, \lambda_h$ and ν are chosen uniformly at random from $\mathbb{Z}_p$, $\lambda_1 X_1 + \cdots + \lambda_h X_h + \nu$ is a pairwise independent function for variables $X_1, \ldots, X_h$ over $\mathbb{Z}_p$. As a result, $\pi = \lambda_1 \mathsf{id}_1 + \cdots + \lambda_\ell \mathsf{id}_\ell + \nu$ and $\sigma = \lambda_1 \widehat{\mathsf{id}}_1 + \cdots + \lambda_{\widehat{\ell}} \widehat{\mathsf{id}}_{\widehat{\ell}} + \nu$ are independent and uniformly distributed. $\mathscr{B}$ returns $\widehat{\mathcal{C}} = (C_{1,i}, C_{2,i})_{i=1,2,3}$.

To show that $\widehat{\mathcal{C}}$ is indeed distributed properly, we show that $C_{1,3}$ is well-formed. Along the same lines, one can check the well-formedness of $C_{1,2}$, $C_{2,2}$ and $C_{2,3}$.

$$\begin{aligned}
C_{1,3} &= -y_v g(\widehat{\mathbf{id}}_\beta)(dbxF_1) - h(\widehat{\mathbf{id}}_\beta)(dbxF_1) - y_v h(\widehat{\mathbf{id}}_\beta)(bxF_1) \\
&= -y_v g(\widehat{\mathbf{id}}_\beta)(dbxF_1) - y_v h(\widehat{\mathbf{id}}_\beta)(bxF_1) - h(\widehat{\mathbf{id}}_\beta)(dbxF_1) - g(\widehat{\mathbf{id}}_\beta)d^2bxF_1 + g(\widehat{\mathbf{id}}_\beta)d^2bxF_1 \\
&= -y_v bx\mathcal{H}_1(\widehat{\mathbf{id}}_\beta) - dbx\mathcal{H}_1(\widehat{\mathbf{id}}_\beta) + g(\widehat{\mathbf{id}}_\beta)d^2x(bF_1) \\
&= -\tau\mathcal{H}_1(\widehat{\mathbf{id}}_\beta) + \sigma\mu V_1'
\end{aligned}$$

Guess: $\mathscr{A}$ returns a bit β' as its guess for β.

When the instance is real, $\mathscr{B}$ simulates $\mathsf{G}_{k-1,1}$ and otherwise simulates $\mathsf{G}_{k,0}$. $\mathscr{B}$ returns 1 if $\mathscr{A}$ wins the game i.e., $\beta = \beta'$; otherwise it returns 0. Hence, $\mathscr{B}$ can solve the LW2 instance with advantage

$$\mathsf{Adv}_{\mathcal{G}}^{\mathrm{LW2}}(\mathscr{B}) = |\Pr[\beta = \beta' | Z_2 \text{ is real}] - \Pr[\beta = \beta' | Z_2 \text{ is random}]| = |\Pr[X_{k-1,1}] - \Pr[X_{k,0}]|.$$

from which the statement of the lemma follows. □

Lemma 6.2.3. $|\Pr[X_{k,0}] - \Pr[X_{k,1}]| \le \varepsilon_{\mathrm{LW2}}$ *for* $1 \le k \le q$.

The proof is reminiscent of Lemma 6.2.2. The reason is as follows: the structure of $\mathcal{S}_2$ is identical to $\mathcal{S}_1$ if the αP_2 term is removed from $K_{2,1}$. Moreover, the simulator chooses α and creates αP_2 independent of the instance. Hence the simulation will be similar except that the instance is now embedded in $\mathcal{S}_2$ and $\mathcal{S}_1$ is made semi-functional independent of the instance.

Lemma 6.2.4. $|\Pr[X_{q,1}] - \Pr[X_{M\text{-}rand}]| \le \varepsilon_{\text{DBDH-3}}$.

Proof. $\mathscr{B}$ receives $(F_1, aF_1, bF_1, sF_1, F_2, aF_2, bF_2, sF_2, Z_T)$ as an instance of the DBDH-3 problem where $Z_T = e(F_1, F_2)^{abs}$ (real) or $Z_T \xleftarrow{\mathrm{U}} \mathbb{G}_T$ (random).

Set-Up: With $y, v, v', y_1, \ldots, y_h, u$ chosen at random from $\mathbb{Z}_p$, $\mathscr{B}$ sets the parameters as

$$P_1 = yF_1, P_2 = yF_2, aP_1 = y(aF_1), V_2 = vF_2, V_2' = v'F_2,\ \tau P_1 = yvF_1 + yv'(aF_1)$$

$$Q_{1,j} = y_jP_1 = y_jyF_1 \text{ for } 1 \le j \le h, U_1 = uP_1 = uyF_1, e(P_1, P_2)^{\alpha} = e(aF_1, bF_2)^{y^2}$$

implicitly setting $\alpha = ab$ and $\tau = v + av'$. The remaining parameters can be computed easily. $\mathscr{B}$ returns $\mathcal{PP}$ to $\mathscr{A}$.

Phases 1 and 2: When $\mathscr{A}$ asks for the secret key for the i'th identity $\mathbf{id}_i = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, $\mathscr{B}$ chooses at random $w_1, w_2, r_1, r_2, r_3, r_4, (z_{1,j}, z_{2,j})_{j=1}^{h}$ and $\gamma_1', \gamma_1, \gamma_2, (\pi_j)_{j=1}^{h}, \eta, (\eta_j)_{j=1}^{h}$ from $\mathbb{Z}_p$ and computes a semi-functional key for $\mathbf{id}_i$ as follows.

$K_{1,1} = w_1P_2 + r_1V_2 - \gamma_1(aF_2),\ K_{1,2} = r_1V_2' + \gamma_1F_2,\ K_{1,3} = r_1F_2$
$K_{2,1} = \gamma_1'(aF_2) + w_1h(\mathbf{id}_i)(P_2) + r_2V_2,\ K_{2,2} = r_2V_2' + y(bF_2) - \gamma_1'F_2,\ K_{2,3} = r_2F_2,$
$D_{j,1} = w_1Q_{2,j} + z_{1,j}V_2 - \gamma_1\pi_j(aF_2),\ D_{j,2} = z_{1,j}V_2' + \gamma_1\pi_jF_2,\ D_{j,3} = z_{1,j}F_2$ for $\ell + 1 \le j \le h$.

$J_{1,1} = w_2P_2 + r_3V_2 - \gamma_2(aF_2),\ J_{1,2} = r_3V_2' + \gamma_2F_2,\ J_{1,3} = r_3F_2$
$J_{2,1} = w_2h(\mathbf{id}_i)(P_2) + r_4V_2 - \gamma_2\eta(aF_2),\ J_{2,2} = r_4V_2' + \gamma_2\eta F_2,\ J_{2,3} = r_4F_2,$
$E_{j,1} = w_2Q_{2,j} + z_{2,j}V_2 - \gamma_2\eta_j(aF_2),\ E_{j,2} = z_{2,j}V_2' + \gamma\eta_jF_2,\ E_{j,3} = z_{2,j}F_2$ for $\ell + 1 \le j \le h$.

Here the relation a$\gamma_1' = by - \gamma_1\pi$ is implicitly set by the simulator. Calculations provided below justify that $K_{2,1}$ and $K_{2,2}$ have the correct distribution. Other elements have the correct form and distribution.

$$\begin{aligned} K_{2,1} &= \gamma_1'(aF_2) + w_1h(\mathbf{id}_i)(P_2) + r_2V_2 \\ &= (by - \gamma_1\pi)(aF_2) + w_1h(\mathbf{id}_i)(P_2) + r_2V_2 \\ &= ab(yF_2) + w_1h(\mathbf{id}_i)(P_2) + r_2V_2 - a\gamma_1\pi F_2 \\ &= \alpha P_2 + w_1h(\mathbf{id}_i)(P_2) + r_2V_2 - a\gamma_1\pi F_2. \end{aligned} \qquad \begin{aligned} K_{2,2} &= r_2V_2' + y(bF_2) - \gamma_1'F_2 \\ &= r_2V_2' + y(bF_2) - (by - \gamma_1\pi)F_2 \\ &= r_2V_2' + y(bF_2) - byF_2 + \gamma_1\pi F_2 \\ &= r_2V_2' + \gamma_1\pi F_2. \end{aligned}$$

Observe that $\mathscr{B}$ does not know α or αF_2 and hence cannot create a normal key.

Challenge: $\mathscr{B}$ receives two pairs $(M_0, \widehat{\mathbf{id}}_0)$ and $(M_1, \widehat{\mathbf{id}}_1)$ from $\mathscr{A}$. It samples $\beta \xleftarrow{\mathrm{U}} \{0,1\}$, $\mu' \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and generates a semi-functional challenge ciphertext as follows.

$C_0 = M_\beta \times Z_T$
$C_{1,1} = yh(\widehat{\mathbf{id}}_\beta)sF_1,\ C_{1,2} = h(\widehat{\mathbf{id}}_\beta)\mu' F_1,\ C_{1,3} = -vyh(\widehat{\mathbf{id}}_\beta)(sF_1) - v'h(\widehat{\mathbf{id}}_\beta)\mu' F_1$
$C_{2,1} = y(sF_1),\ C_{2,2} = \mu' F_1,\ C_{2,3} = -yv(sF_1) - v'\mu' F_1$

with $\mu' = asy + \mu$ and $\sigma = h(\widehat{\mathbf{id}}_\beta)$. The challenge ciphertext $\widehat{\mathcal{C}} = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ is returned to $\mathscr{A}$.

Guess: $\mathscr{A}$ returns its guess β' of β.

It is clear that $\widehat{\mathcal{C}}$ is a semi-functional encryption of M_β when $Z_T = e(P_1, P_2)^{abs}$. And when $Z_T \xleftarrow{\text{U}} \mathbb{G}_T$ $\widehat{\mathcal{C}}$ would be a semi-functional encryption of a random message. Hence $\mathscr{B}$ simulates $\mathsf{G}_{q,1}$ or G_{final} according to Z_T being real or random respectively. If the algorithm $\mathscr{B}$ returns 1 when $\beta = \beta'$ and 0 otherwise, it can solve the DBDH-3 instance with advantage

$$\mathsf{Adv}_{\mathcal{G}}^{\text{DBDH-3}}(\mathscr{B}) = |\Pr[\beta = \beta' | Z_T \text{ is real}] - \Pr[\beta = \beta' | Z_T \text{ is random}]| = |\Pr[X_{q,1}] - \Pr[X_{final}]|.$$

□

Lemma 6.2.5. $|\Pr[X_{M\text{-}rand}] - \Pr[X_{final}]| \leq \varepsilon_{\text{A1}}$.

Proof. Let $(\mathcal{G}, F_1, zF_1, dzF_1, azF_1, adzF_1, szF_1, F_2, zF_2, aF_2, xF_2, (dz - ax)F_2, Z_1)$ be the instance of A1 provided to $\mathscr{B}$. Let $Z_1 = c \cdot sdzF_1$. $\mathscr{B}$ has to determine whether $c = 1$ or $c \xleftarrow{\text{U}} \mathbb{Z}_p$. The game is simulated as follows.

Set-Up: Pick $\alpha, v, v', y_1, \ldots, y_h, u \xleftarrow{\text{U}} \mathbb{Z}_p$ and set the parameters as

$$P_1 = zF_1,\ V_2 = vF_2,\ V_2' = v'F_2,\ Q_{1,j} = y_j(dzF_1),\ U_1 = u(dzF_1),$$

$$aP_1 = azP_1,\ aQ_{1,j} = y_j(adzF_1),\ aU_1 = u(adzF_1),$$

where $j = 1, \ldots, h$ and similarly the elements τP_1, $\tau Q_{1,j}$ and τU_1. Compute $e(P_1, P_2)^\alpha = e(zF_1, zF_2)^\alpha$. $\mathscr{B}$ returns $\mathcal{PP}$ to $\mathscr{A}$. $\mathscr{B}$ knows $P_2 = zF_2$ and α but not $Q_{2,j}$'s and U_2. The main idea is to mask the components required to create identity-hash in $\mathbb{G}_2$ by a scalar multiple of aF_2 so that only semi-functional keys can be created.

Key Extraction Phases 1 and 2: $\mathscr{B}$ picks $w_1, w_2, r_1, r_2, r_3, r_4, (z_{1,j}, z_{2,j})_{j=1}^h \xleftarrow{\text{U}} \mathbb{Z}_p$, $\gamma_1, \gamma_2 \xleftarrow{\text{U}} \mathbb{Z}_p^\times$ and $\pi', (\pi_j')_{j=1}^h, \eta', (\eta_j')_{j=1}^h \xleftarrow{\text{U}} \mathbb{Z}_p$. It then computes the key for the i-th identity vector $\mathbf{id}_i = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ as follows.

$K_{1,1} = w_1(zF_2) + r_1V_2 - \gamma_1 aF_2,\ K_{1,2} = r_1V_2' + \gamma_1 F_2,\ K_{1,3} = r_1F_2$
$K_{2,1} = \alpha zF_2 + w_1h(\mathbf{id}_i)(dz - ax)F_2 + r_2V_2 - \gamma_1\pi'(aF_2),\ K_{2,2} = r_2V_2' + w_1h(\mathbf{id}_i)xF_2 + \gamma_1\pi' F_2,\ K_{2,3} = r_2F_2,$

$J_{1,1} = w_2(zF_2) + r_3V_2 - \gamma_2 aF_2,\ J_{1,2} = r_3V_2' + \gamma_2 F_2,\ J_{1,3} = r_3F_2$
$J_{2,1} = w_2h(\mathbf{id}_i)(dz - ax)F_2 + r_4V_2 - \gamma_2\eta'(aF_2),\ J_{2,2} = r_4V_2' + w_2h(\mathbf{id}_i)xF_2 + \gamma_2\eta' F_2,\ J_{2,3} = r_4F_2,$

For $\ell + 1 \leq j \leq h$,
$D_{j,1} = w_1y_j(dz - ax)F_2 + z_{1,j}V_2 - \gamma_1\pi_j'(aF_2),\ D_{j,2} = z_{1,j}V_2' + w_1y_j(xF_2) + \gamma_1\pi_j' F_2,\ D_{j,3} = z_{1,j}F_2$
$E_{j,1} = w_2y_j(dz - ax)F_2 + z_{2,j}V_2 - \gamma_2\eta_j'(aF_2),\ E_{j,2} = z_{2,j}V_2' + w_2y_j(xF_2) + \gamma_2\eta_j' F_2,\ E_{j,3} = z_{2,j}F_2$

setting $\pi = \pi' + \gamma_1^{-1} w_1 h(\mathbf{id}_i)x$, $\pi_j = \pi'_j + \gamma_1^{-1} w_1 y_j x$, $\eta = \eta' + \gamma_2^{-1} w_2 h(\mathbf{id}_i)x$ and $\eta_j = \eta'_j + \gamma_2^{-1} w_2 y_j x$. Since all these scalars are additively randomised they remain properly distributed in the adversary's view. We show that $D_{j,1}, D_{j,2}$ are well-formed; the rest can be verified in a similar fashion.

$$\begin{aligned}
D_{j,1} &= w_1 y_j (dz - ax)F_2 + z_{1,j} V_2 - \gamma_1 \pi'_j (aF_2) \\
&= w_1 y_j dz F_2 - w_1 y_j ax F_2 + z_{1,j} V_2 - \gamma_1 (\pi_j - \gamma_1^{-1} w_1 y_j x)(aF_2) \\
&= w_1 y_j dz F_2 - w_1 y_j ax F_2 + z_{1,j} V_2 - a\gamma_1 \pi_j F_2 + w_1 y_j ax F_2 \\
&= w_1 y_j dz F_2 + z_{1,j} V_2 - a\gamma_1 \pi_j F_2 \\
D_{j,2} &= z_{1,j} V'_2 + w_1 y_j (xF_2) + \gamma_1 \pi'_j F_2 \\
&= z_{1,j} V'_2 + w_1 y_j (xF_2) + \gamma_1 (\pi_j - \gamma_1^{-1} w_1 y_j x) F_2 \\
&= z_{1,j} V'_2 + w_1 y_j x F_2 + \gamma_1 \pi_j F_2 - w_1 y_j x F_2 \\
&= z_{1,j} V'_2 + \gamma_1 \pi_j F_2
\end{aligned}$$

Challenge: $\mathscr{B}$ receives two pairs of messages and identity vectors $(M_0, \widehat{\mathbf{id}}_0)$ and $(M_1, \widehat{\mathbf{id}}_1)$ from $\mathscr{A}$. It chooses $\beta \xleftarrow{\text{U}} \{0,1\}$ and $a', \xi \xleftarrow{\text{U}} \mathbb{Z}_p$ at random and generates a semi-functional challenge ciphertext as follows.

$C_0 \xleftarrow{\text{U}} \mathbb{G}_T$
$C_{1,1} = h(\widehat{\mathbf{id}}_\beta) Z_1$, $C_{1,2} = a' h(\widehat{\mathbf{id}}_\beta) Z_1 + \xi F_1$, $C_{1,3} = -v h(\widehat{\mathbf{id}}_\beta) Z_1 - v' a' h(\widehat{\mathbf{id}}_\beta) Z_1 - v' \xi F_1$,
$C_{2,1} = szF_1$, $C_{2,2} = a' szF_1$, $C_{2,3} = -v(szF_1) - v'a'(szF_1)$,

where $a' = a + \mu'$, $\mu = \mu' sz$ and $\xi = \mu\sigma'$. The challenge ciphertext $\widehat{\mathcal{C}} = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ is returned to $\mathscr{A}$. The computations below illustrate that $\widehat{\mathcal{C}}$ is a semi-functional encryption with $\sigma = \sigma' + cdh(\widehat{\mathbf{id}}_\beta)$.

$$\begin{aligned}
C_{1,2} &= a' h(\widehat{\mathbf{id}}_\beta) Z_1 + \xi F_1 \\
&= (a + \mu') h(\widehat{\mathbf{id}}_\beta) csdz F_1 + \mu\sigma' F_1 \\
&= a h(\widehat{\mathbf{id}}_\beta) csdz F_1 + \mu' h(\widehat{\mathbf{id}}_\beta) csdz F_1 + \mu\sigma' F_1 \\
&= as\mathcal{H}_1(\widehat{\mathbf{id}}_\beta) + (\mu' sz)(cdh(\widehat{\mathbf{id}}_\beta))F_1 + \mu\sigma' F_1 \\
&= as\mathcal{H}_1(\widehat{\mathbf{id}}_\beta) + \mu(cdh(\widehat{\mathbf{id}}_\beta))F_1 + \mu\sigma' F_1 \\
&= as\mathcal{H}_1(\widehat{\mathbf{id}}_\beta) + \mu\sigma F_1
\end{aligned}$$

Observe that $C_{1,1} = s\mathcal{H}_1(\widehat{\mathbf{id}}_\beta) = (c \cdot h(\widehat{\mathbf{id}}_\beta))(sdzF_1)$. If $c = 1$, then $\sigma = \sigma' + dh(\widehat{\mathbf{id}}_\beta)$ and $\widehat{\mathcal{C}}$ is encrypted under $\widehat{\mathbf{id}}_\beta$. Otherwise, c is random, causing $h(\widehat{\mathbf{id}}_\beta)$ and consequently the target identity and σ to be random quantities.

Guess: $\mathscr{A}$ returns its guess β' of β.

If the algorithm $\mathscr{B}$ returns 1 when $\beta = \beta'$ and 0 otherwise, it can solve the A1 instance with advantage

$$\mathsf{Adv}_{\mathcal{G}}^{\text{A1}}(\mathscr{B}) = |\Pr[\beta = \beta' | Z_1 \text{ is real}] - \Pr[\beta = \beta' | Z_1 \text{ is random}]| = |\Pr[X_{M\text{-}rand}] - \Pr[X_{final}]|.$$

□

6.3 Extending JR-IBE to CC-HIBE

Schemes $\mathcal{JR}$-$\mathcal{AHIBE}$ and $\mathcal{JR}$-$\mathcal{HIBE}$ extend the JR-IBE to anonymous and non-anonymous CC-HIBEs respectively. At a top level, the identity-hashing technique of Boneh-Boyen-Goh [24] (BBG-hash) is applied on JR-IBE. We work in the setting of asymmetric pairings where ciphertext components are elements of $\mathbb{G}_1$ and key components are elements of $\mathbb{G}_2$. BBG-hash of the identity is required to be computed in both $\mathbb{G}_1$ and $\mathbb{G}_2$. During encryption, the BBG-hash is required to be computed in $\mathbb{G}_1$ and this requires adding some elements of $\mathbb{G}_1$ to the public parameters.

In previous CC-HIBE schemes in the prime-order setting within the dual system framework [108, 137], anonymity appears as a by-product of the HIBE extension. The basic difficulty in making it non-anonymous was due to the following dichotomy concerning key delegation. The BBG-hash for the key is computed in $\mathbb{G}_2$. The hash is defined using certain elements of $\mathbb{G}_2$. During key delegation, the hash has to be rerandomised and so the elements should be publicly available. On the other hand, information about these elements must not be leaked because they form the source of randomness used to generate the semi-functional components during simulation.

The problem described above does not arise in case of JR-IBE. The feature of JR-IBE that makes extension to the non-anonymous CC-HIBE $\mathcal{JR}$-$\mathcal{HIBE}$ possible is as follows. The master secret consists of two elements whose linear combination is used to mask the message during encryption. This is unlike previous (H)IBE schemes where a single element was used for the purpose. The two elements would be information theoretically hidden from an attacker's view. So the secret randomness for the semi-functional ciphertext space is provided by one of the two elements.

Anonymity is achieved by keeping the elements required to compute the BBG-hash in $\mathbb{G}_2$ to be secret and instead provide suitably randomised copies of these elements in the user keys. Problems then arise while defining *semi-functional* components and arguing about their well-formedness during simulation. Fortunately, it turns out that the problems can be handled by using appropriate algebraic relations. The technique of keeping certain elements hidden and providing their randomised version in the user keys closely follow the ideas introduced in [36] to obtain anonymity. In $\mathcal{JR}$-$\mathcal{AHIBE}$ the elements that are kept hidden are exactly the ones required to create the BBG-hash in $\mathbb{G}_2$. As a result, an adversary is unable to create an identity hash in $\mathbb{G}_2$ and cancel it out with the BBG-hash of the same identity in $\mathbb{G}_1$. This naturally leads to the scheme $\mathcal{JR}$-$\mathcal{AHIBE}$ being anonymous.

We note that a single-level instantiation of $\mathcal{JR}$-$\mathcal{HIBE}$ provides a non-anonymous variant of the JR-IBE with rerandomisable keys.

6.3.1 Jutla-Roy IBE with Ciphertexts in $\mathbb{G}_1$

In the IBE scheme of Jutla-Roy [103] (JR-IBE), ciphertext consists of elements in $\mathbb{G}_2$ and keys contain elements from $\mathbb{G}_1$. For Type-3 pairings, elements of $\mathbb{G}_1$ have a shorter representation compared to the elements of $\mathbb{G}_2$. To reduce the length of the ciphertext, one has to interchange the roles of the two groups. In contrast, for a signature scheme, it would be advantageous to have the signature to consist of elements from $\mathbb{G}_1$. Since the JR-IBE is obtained from non-interactive zero knowledge (NIZK) proofs via the idea of signatures, the scheme results in ciphertext elements being in $\mathbb{G}_2$.

This section describes a "dual" of the Jutla-Roy [103] (JR-IBE-D) where ciphertexts live in $\mathbb{G}_1$ and keys in $\mathbb{G}_2$. We use a compact notation to denote normal and semi-functional ciphertexts and keys. The group elements shown in curly brackets { } are the semi-functional components. To get the scheme itself, these components should be ignored.

Parameters: Choose $P_1 \xleftarrow{\text{U}} \mathbb{G}_1^{\times}$, $P_2 \xleftarrow{\text{U}} \mathbb{G}_2^{\times}$, $\Delta_1, \Delta_2, \Delta_3, \Delta_4, c, d, u, e \xleftarrow{\text{U}} \mathbb{Z}_p$, $b \xleftarrow{\text{U}} \mathbb{Z}_p^{\times}$, and set $U_1 = (-\Delta_1 b + d)P_1$, $V_1 = (-\Delta_2 b + e)P_1$, $W_1 = (-\Delta_3 b + c)P_1$, $g_T = e(P_1, P_2)^{-\Delta_4 b + u}$. The parameters are given by

$\mathcal{PP} : (P_1, bP_1, U_1, V_1, W_1, g_T)$
$\mathcal{MSK} : (P_2, cP_2, \Delta_1, \Delta_2, \Delta_3, \Delta_4, d, u, e)$ 7

Ciphertext: Consists of $(C_0, C_1, C_2, C_3, \mathsf{tag})$ where

$\mathsf{tag}, s \xleftarrow{\text{U}} \mathbb{Z}_p$, $\{\mu \xleftarrow{\text{U}} \mathbb{Z}_p\}$
$C_0 = m \cdot (g_T)^s \{e(P_1, P_2)^{u\mu}\}$,
$C_1 = sP_1\{+\mu P_1\}$, $C_2 = sbP_1$, $C_3 = s(U_1 + \mathsf{id}V_1 + \mathsf{tag}W_1)\{+\mu(d + \mathsf{id} \cdot e + \mathsf{tag} \cdot c)P_1\}$.

Key: Contains five elements $(K_1, \ldots, K_5)$ defined as follows.

$r \xleftarrow{\text{U}} \mathbb{Z}_p$, $\{\gamma, \pi \xleftarrow{\text{U}} \mathbb{Z}_p\}$
$K_1 = rP_2$, $K_2 = rcP_2\{+\gamma P_2\}$, $K_3 = (u + r(d + \mathsf{id}e))\,P_2\{+\gamma\pi P_2\}$,
$K_4 = -r\Delta_3 P_2\{-\frac{\gamma}{b}P_2\}$, $K_5 = (-\Delta_4 - r(\Delta_1 + \mathsf{id}\Delta_2))\,P_2\{-\frac{\gamma\pi}{b}P_2\}$.

Note 6.3.1. In JR-IBE [103], b is mentioned to be an element of $\mathbb{Z}_p$. This is an oversight and b should be an element of $\mathbb{Z}_p^{\times}$ as we have mentioned above. This is because if b is zero, then division by b and consequently the definitions of the semi-functional components will not be meaningful.

The original JR-IBE scheme in [103] was proved to be secure based on the SXDH assumption. Straightforward modifications of proof in [103] will also show the security of the variant JR-IBE-D under the same assumption. For the sake of completeness, we state the security theorem JR-IBE-D.

Theorem 6.3.1. *If* $(\varepsilon_{\text{DDH1}}, t_1)$-DDH1 *and* $(\varepsilon_{\text{DDH2}}, t_2)$-DDH2 *assumptions hold in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$ *respectively, then* JR-IBE-D *is* (ε, t)-$\mathsf{ANO\text{-}IND\text{-}ID\text{-}CPA}$-*secure where* $\varepsilon \leq \varepsilon_{\text{DDH1}} + q \cdot \varepsilon_{\text{DDH2}} + (q/p)$, $t_1 = t + O(\rho)$ *and* $t_2 = t + O(\rho)$. ρ *is the maximum time required for one scalar multiplication in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$.

A note on notation and proof technique. We have used the JR-IBE [103] as the basic building block and consequently, our notation and proofs build on that of [103]. This makes it easier for a reader to see the connections between our work and the IBE construction in [103]. We note, though, that we have provided all the relevant details and it is possible to directly verify, with a bit of work, all the claims in this paper without referring to [103]. Frameworks for presenting dual-system constructions and proofs have been proposed [111, 71]. Neither the JR-IBE nor the constructions in the present work appear to fall within these frameworks.

6.4 CC-HIBE Constructions

Both schemes $\mathcal{JR}\text{-}\mathcal{AHIBE}$ and $\mathcal{JR}\text{-}\mathcal{HIBE}$ are based on a Type-3 prime-order pairing with group order p. Identities are variable length tuples of elements from $\mathbb{Z}_p^\times$ with maximum length h.

As is typical with BBG-type extensions the element V_1 is replaced with h elements $V_{1,1}, \ldots, V_{1,h}$ – one for each level of an identity. The set $U_1, (V_{1,j})_{j\in[1,h]}$ is used to create the identity hash – for an identity $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, the hash is given by $U_1 + \sum_{j=1}^{\ell} \mathsf{id}_j V_{1,j}$. Element W_1 will be retained to append the tag-component to the hash. This replaces the hash in JR-IBE-D ciphertext without affecting the number of elements in the ciphertext. Moreover, since the hash is embedded in a single ciphertext component, only one tag is required. Note that the keys in JR-IBE-D have two sub-hashes that when combined during decryption cancels with the hash of the ciphertext.

In JR-IBE-D, each of U_1, V_1, W_1 is split into two components kept as part of the master secret. The two sets of components determine the sub-hashes required in generating keys. Similarly, for the HIBE, we need to split $V_{1,j}$ for all $j \in [1,h]$ as $V_{1,j} = b\Delta_{2,j} + e_j$ where $\Delta_{1,j}, e_j \xleftarrow{\text{U}} \mathbb{Z}_p$. So the sub-hashes are determined by the vectors $\mathbf{v}_1 = (d, e_1, \ldots, e_h)$ and $\mathbf{v}_2 = (\Delta_1, \Delta_{2,1}, \ldots, \Delta_{2,h})$. Rerandomisation of keys during delegation can be done in two possible ways – make the encodings of vectors $\mathbf{v}_1, \mathbf{v}_2$ along with Δ_3, c in $\mathbb{G}_2$ public; or provide appropriately randomised copies of these elements in the key.

The second method retains the anonymity property leading to the scheme $\mathcal{JR}\text{-}\mathcal{AHIBE}$. This is because the vectors $\mathbf{v}_1, \mathbf{v}_2$ can be used to test whether a given ciphertext is encrypted to a particular identity or not. Keeping them secret naturally leads to anonymity. The former method leads to the scheme $\mathcal{JR}\text{-}\mathcal{HIBE}$ that has shorter keys and faster algorithms compared to $\mathcal{JR}\text{-}\mathcal{AHIBE}$. But the efficiency comes at the cost of losing anonymity. Due to space constraints we only describe $\mathcal{JR}\text{-}\mathcal{AHIBE}$ and discuss its security. A description of $\mathcal{JR}\text{-}\mathcal{HIBE}$ followed by an outline of its security is provided in Appendices 6.5.1 and 6.5.2.

6.4.1 Scheme $\mathcal{JR}\text{-}\mathcal{AHIBE}$

The definition of various algorithms of $\mathcal{JR}\text{-}\mathcal{AHIBE}$ = ($\mathcal{JR}\text{-}\mathcal{AHIBE}$.Setup, $\mathcal{JR}\text{-}\mathcal{AHIBE}$.Encrypt, $\mathcal{JR}\text{-}\mathcal{AHIBE}$.KeyGen, $\mathcal{JR}\text{-}\mathcal{AHIBE}$.Delegate, $\mathcal{JR}\text{-}\mathcal{AHIBE}$.Decrypt) is provided below.

$\mathcal{JR}\text{-}\mathcal{AHIBE}$.Setup($\kappa$): Generate a Type-3 pairing $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, F_1, F_2)$ based on the security parameter κ. Compute parameters as follows.

$$P_1 \xleftarrow{\text{U}} \mathbb{G}_1^\times,\ P_2 \xleftarrow{\text{U}} \mathbb{G}_2^\times$$
$$\Delta_1, \Delta_3, \Delta_4, c, d, u, (\Delta_{2,j}, e_j)_{j=1}^h \xleftarrow{\text{U}} \mathbb{Z}_p,\ b \xleftarrow{\text{U}} \mathbb{Z}_p^\times,$$

$$U_1 = (-\Delta_1 b + d)P_1,\ V_{1,j} = (-\Delta_{2,j} b + e_j)P_1 \text{ for } j = 1, \ldots, h,\ W_1 = (-\Delta_3 b + c)P_1,$$
$$g_T = e(P_1, P_2)^{-\Delta_4 b + u},$$

$$\mathcal{PP} : (P_1, bP_1, U_1, (V_{1,j})_{j=1}^h, W_1, g_T)$$
$$\mathcal{MSK} : (P_2, cP_2, \Delta_1, \Delta_3, \Delta_4, d, u, (\Delta_{2,j}, e_j)_{j=1}^h)$$

$\mathcal{JR}\text{-}\mathcal{AHIBE}$.Encrypt($\mathcal{PP}, M, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$): Pick $\mathsf{tag}, s \xleftarrow{\text{U}} \mathbb{Z}_p$ and set the ciphertext $\mathcal{C} = (C_0, C_1, C_2, C_3, \mathsf{tag})$ where

$C_0 = M \cdot (g_T)^s$, $C_1 = sP_1$, $C_2 = sbP_1$, $C_3 = s(U_1 + \sum_{j=1}^{\ell} \mathsf{id}_j V_{1,j} + \mathsf{tag} W_1)$.

$\mathcal{JR}$-$\mathcal{AHIBE}$.$\mathsf{KeyGen}(\mathcal{MSK}, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell))$: Pick $r_1, r_2 \xleftarrow{\text{U}} \mathbb{Z}_p$ and compute the secret key $\mathcal{SK}_{\mathbf{id}} = (\mathcal{S}_1, \mathcal{S}_2)$ for $\mathbf{id}$, with $\mathcal{S}_1 = ((K_i)_{i \in [1,5]}, (D_{1,j}, E_{1,j})_{j \in [\ell+1,h]})$ and $\mathcal{S}_2 = ((J_i)_{i \in [1,5]}, (D_{2,j}, E_{2,j})_{j \in [\ell+1,h]})$ where

$$\begin{aligned}
&K_1 = r_1 P_2,\ K_2 = r_1 c P_2,\ K_3 = \left(u + r_1(d + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j e_j)\right) P_2,\\
&K_4 = -r_1 \Delta_3 P_2,\ K_5 = \left(-\Delta_4 - r_1(\Delta_1 + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j})\right) P_2,\\
&D_{1,j} = r_1 e_j P_2,\ E_{1,j} = -r_1 \Delta_{2,j} P_2 \text{ for } j = \ell+1, \ldots, h,
\end{aligned}$$

$$\begin{aligned}
&J_1 = r_2 P_2,\ J_2 = r_2 c P_2,\ J_3 = r_2 \left(d + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j e_j\right) P_2,\\
&J_4 = -r_2 \Delta_3 P_2,\ J_5 = -r_2(\Delta_1 + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j}) P_2,\\
&D_{2,j} = r_2 e_j P_2,\ E_{2,j} = -r_2 \Delta_{2,j} P_2 \text{ for } j = \ell+1, \ldots, h
\end{aligned}$$

$\mathcal{JR}$-$\mathcal{AHIBE}$.$\mathsf{Delegate}(\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathsf{id}_{\ell+1})$: Let $\mathbf{id} : \mathsf{id}_{\ell+1} = (\mathsf{id}_1, \ldots, \mathsf{id}_{\ell+1})$. $\mathcal{SK}_{\mathbf{id}:\mathsf{id}_{\ell+1}}$ is generated from $\mathcal{SK}_{\mathbf{id}}$ as follows.

$$\tilde{r}_1, \tilde{r}_2 \xleftarrow{\text{U}} \mathbb{Z}_p^{\times},$$

$$\begin{aligned}
&K_1 \leftarrow K_1 + \tilde{r}_1 J_1,\ \ K_2 \leftarrow K_2 + \tilde{r}_1 J_2,\ \ K_3 \leftarrow (K_3 + \mathsf{id}_{\ell+1} D_{1,\ell+1}) + \tilde{r}_1 (J_3 + \mathsf{id}_{\ell+1} D_{2,\ell+1}),\\
&K_4 \leftarrow K_4 + \tilde{r}_1 J_4,\ \ K_5 \leftarrow (K_5 + \mathsf{id}_{\ell+1} E_{1,\ell+1}) + \tilde{r}_1 (J_5 + \mathsf{id}_{\ell+1} E_{2,\ell+1}),\\
&D_{1,j} \leftarrow D_{1,j} + \tilde{r}_1 D_{2,j},\ \ E_{1,j} \leftarrow E_{1,j} + \tilde{r}_1 E_{2,j} \text{ for } j = \ell+2, \ldots, h,
\end{aligned}$$

$$\begin{aligned}
&J_1 \leftarrow \tilde{r}_2 J_1,\ \ J_2 \leftarrow \tilde{r}_2 J_2,\ \ J_3 \leftarrow \tilde{r}_2 (J_3 + \mathsf{id}_{\ell+1} D_{2,\ell+1}),\\
&J_4 \leftarrow \tilde{r}_2 J_4,\ \ J_5 \leftarrow \tilde{r}_2 (J_5 + \mathsf{id}_{\ell+1} E_{2,\ell+1}),\\
&D_{2,j} \leftarrow \tilde{r}_2 D_{2,j},\ \ E_{2,j} \leftarrow \tilde{r}_2 E_{2,j} \text{ for } j = \ell+2, \ldots, h,
\end{aligned}$$

setting $r_1 \leftarrow r_1 + \tilde{r}_1 r_2$ and $r_2 \leftarrow \tilde{r}_2 r_2$. Note that the new values of r_1 and r_2 have uniform and independent distribution over $\mathbb{Z}_p$ given that $r_1, r_2 \xleftarrow{\text{U}} \mathbb{Z}_p$ and $\tilde{r}_1, \tilde{r}_2 \xleftarrow{\text{U}} \mathbb{Z}_p^{\times}$. Hence the distribution of $\mathcal{SK}_{\mathbf{id}:\mathsf{id}_{\ell+1}}$ is same as that of a freshly generated key for $\mathbf{id} : \mathsf{id}_{\ell+1}$ via the $\mathcal{JR}$-$\mathcal{AHIBE}$.KeyGen algorithm.

$\mathcal{JR}$-$\mathcal{AHIBE}$.$\mathsf{Decrypt}(\mathcal{C}, \mathcal{SK}_{\mathbf{id}})$: Return M' computed as: $M' = \dfrac{C_0 \cdot e(C_3, K_1)}{e(C_1, \mathsf{tag} K_2 + K_3) e(C_2, \mathsf{tag} K_4 + K_5)}$.

Correctness: For all messages M, for all $1 \leq \ell \leq h$, for all identities $\mathbf{id}$ of length ℓ, for all $\mathcal{C}$ and $\mathcal{SK}_{\mathbf{id}}$ such that $\mathcal{C} \leftarrow \mathcal{JR}$-$\mathcal{AHIBE}$.$\mathsf{Encrypt}(M, \mathbf{id})$, $\mathcal{SK}_{\mathbf{id}} \leftarrow \mathcal{JR}$-$\mathcal{AHIBE}$.$\mathsf{KeyGen}(\mathcal{MSK}, \mathbf{id})$ and $M' = \mathcal{JR}$-$\mathcal{AHIBE}$.$\mathsf{Decrypt}(\mathcal{C}, \mathcal{SK}_{\mathbf{id}})$, it holds that $M' = M$. The following computation substantiates this claim. Let $(\mathcal{C} = (C_0, C_1, C_2, C_3)) = \mathcal{JR}$-$\mathcal{AHIBE}$.$\mathsf{Encrypt}(M, \mathbf{id}; s)$ and $(\mathcal{SK}_{\mathbf{id}} = (\mathcal{S}_1, \mathcal{S}_2)) = \mathcal{JR}$-$\mathcal{AHIBE}$.$\mathsf{KeyGen}(\mathcal{MSK}, \mathbf{id}; r_1, r_2)$ with $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$. We show the computation in steps.

Let $h_1 = d + \sum_{j=1}^{\ell} \mathsf{id}_j e_j + \mathsf{tag} \cdot c$ and $h_2 = \Delta_1 + \sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j} + \mathsf{tag} \cdot \Delta_3$.

$$\begin{aligned}
e(C_1, \mathsf{tag}K_2 + K_3) &= e(sP_1, \mathsf{tag} \cdot r_1 c P_2 + (u + r_1(d + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j e_j))P_2) \\
&= e(sP_1, uP_2 + r_1(d + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j e_j + \mathsf{tag} \cdot c)P_2) \\
&= e(P_1, P_2)^{su} e(P_1, P_2)^{r_1 s h_1} \\
e(C_2, \mathsf{tag} \cdot K_4 + K_5) &= e(sbP_1, -\mathsf{tag} \cdot r_1 \Delta_3 P_2 - (\Delta_4 + r_1(\Delta_1 + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j}))P_2) \\
&= e(sbP_1, -\Delta_4 P_2 - r_1(\Delta_1 + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j} + \mathsf{tag} \cdot \Delta_3)P_2) \\
&= e(P_1, P_2)^{-sb\Delta_4} e(P_1, P_2)^{-r_1 sbh_2} \\
e(C_3, K_1) &= e(s(U_1 + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j V_{1,j} + \mathsf{tag} \cdot W_1), r_1 P_2) \\
&= e((-\Delta_1 b + d)P_1 + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j(-\Delta_{2,j} b + e_j)P_1 + \mathsf{tag} \cdot (-\Delta_3 b + c)P_1, P_2)^{r_1 s} \\
&= e(-(\Delta_1 + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j} + \mathsf{tag} \cdot \Delta_3)bP_1, P_2)^{r_1 s} e((d + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j e_j + \mathsf{tag} \cdot c)P_1, P_2)^{r_1 s} \\
&= e(P_1, P_2)^{-r_1 sbh_2} e(P_1, P_2)^{r_1 s h_1}
\end{aligned}$$

Then, the message M' obtained after decryption is given by

$$\begin{aligned}
M' &= \frac{C_0 \cdot e(C_3, K_1)}{e(C_1, \mathsf{tag} \cdot K_2 + K_3) e(C_2, \mathsf{tag} \cdot K_4 + K_5)} \\
&= \frac{M \cdot g_T^s \cdot e(P_1, P_2)^{-r_1 sbh_2} e(P_1, P_2)^{r_1 s h_1}}{e(P_1, P_2)^{su} e(P_1, P_2)^{r_1 s h_1} e(P_1, P_2)^{-sb\Delta_4} e(P_1, P_2)^{-r_1 sbh_2}} \\
&= \frac{M \cdot e(P_1, P_2)^{(-\Delta_4 b + u)s}}{e(P_1, P_2)^{s(-\Delta_4 b + u)}} \\
&= M,
\end{aligned}$$

as required.

The above holds as well for all $\mathcal{SK}_{\mathbf{id}}$ derived from secret keys for higher level identities through the $\mathcal{JR}$-$\mathcal{AHIBE}$.Delegate algorithm. This is because a derived key have the same distribution as a key generated by a fresh call to the $\mathcal{JR}$-$\mathcal{AHIBE}$.KeyGen algorithm which has been pointed out in the description of the $\mathcal{JR}$-$\mathcal{AHIBE}$.Delegate algorithm.

From a dual system perspective. One can see in Section 6.5 that the scalar u, along with scalars $d, c, e_{j_{j \in [1,h]}}$, define the semi-functional ciphertext space for $\mathcal{JR}$-$\mathcal{AHIBE}$. These scalars provide the secret information for simulating semi-functional components. A crucial requirement for a dual system proof is that these scalars are statistically hidden from the adversary. Observe that the element g_T in the public parameters, information theoretically hides the element u. Similarly, elements $U_1, V_{1,j}, W_1$ hide the scalars d, e_j, c respectively. Further intuition with respect to the dual-system proof and a sketch of how the various scalars interact is provided in Section 6.5.

6.5 Security of $\mathcal{JR}$-$\mathcal{AHIBE}$

The scheme $\mathcal{JR}$-$\mathcal{AHIBE}$ is proved secure in the sense of ANO-IND-ID-CPA (described in Section 2.2.1.3) following the dual system methodology introduced by Waters [158]. We first provide algorithms $\mathcal{JR}$-$\mathcal{AHIBE}$.SFEncrypt and $\mathcal{JR}$-$\mathcal{AHIBE}$.SFKeyGen that generate semi-functional ciphertexts and keys (respectively) required for a dual system proof. In addition, we need an algorithm PSFKeyGen that generates *partial semi-functional keys* ([137]). These are required only in the security proof of $\mathcal{JR}$-$\mathcal{AHIBE}$ and not $\mathcal{JR}$-$\mathcal{HIBE}$.

$\mathcal{JR}$-$\mathcal{AHIBE}$.SFEncrypt$(\mathcal{MSK},\mathcal{C})$: Let $(\mathcal{C} = (C_0,C_1,C_2,C_3)) \leftarrow$ $\mathcal{JR}$-$\mathcal{AHIBE}$.Encrypt$(m,\mathbf{id} = (\mathsf{id}_1,\ldots,\mathsf{id}_\ell))$. Pick $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$ and modify the components of $\mathcal{C}$ as follows.

$$C_0 \leftarrow C_0 \cdot e(P_1,P_2)^{u\mu}, \quad C_1 \leftarrow C_1 + \mu P_1, \quad C_2 \leftarrow C_2, \quad C_3 \leftarrow C_3 + \mu(d + \textstyle\sum_{j=1}^{\ell} \mathsf{id}_j e_j + \mathsf{tag}\cdot c)P_1.$$

Return the modified ciphertext $\mathcal{C} = (C_0,C_1,C_2,C_3)$.

$\mathcal{JR}$-$\mathcal{AHIBE}$.SFKeyGen$(\mathcal{MSK},\mathcal{SK}_{\mathbf{id}})$: This algorithm takes in a normal secret key $\mathcal{SK}_{\mathbf{id}} = (\mathcal{S}_1,\mathcal{S}_2)$ for identity $\mathbf{id} = (\mathsf{id}_1,\ldots,\mathsf{id}_\ell)$ and generates a semi-functional key as follows.

$$\gamma_1,\gamma_2,\pi,\sigma,(\pi_j,\sigma_j)_{j=1}^{h} \xleftarrow{\text{U}} \mathbb{Z}_p,$$

$$K_1 \leftarrow K_1,\ K_2 \leftarrow K_2 + \gamma_1 P_2,\ K_3 \leftarrow K_3 + \gamma_1\pi P_2,\ K_4 \leftarrow K_4 - \left(\frac{\gamma_1}{b}\right)P_2,\ K_5 \leftarrow K_5 - \left(\frac{\gamma_1\pi}{b}\right)P_2,$$
$$D_{1,j} \leftarrow D_{1,j} + \gamma_1\pi_j P_2,\ E_{1,j} \leftarrow E_{1,j} - \left(\frac{\gamma_1\pi_j}{b}\right)P_2 \ \text{ for } j = \ell+1,\ldots,h,$$

$$J_1 \leftarrow J_1,\ J_2 \leftarrow J_2 + \gamma_2 P_2,\ J_3 \leftarrow J_3 + \gamma_2\sigma P_2,\ J_4 \leftarrow J_4 - \left(\frac{\gamma_2}{b}\right)P_2,\ J_5 \leftarrow J_5 - \left(\frac{\gamma_2\sigma}{b}\right)P_2,$$
$$D_{2,j} \leftarrow D_{2,j} + \gamma_2\sigma_j P_2,\ E_{2,j} \leftarrow E_{2,j} - \left(\frac{\gamma_2\sigma_j}{b}\right)P_2 \ \text{ for } j = \ell+1,\ldots,h,$$

The resulting key $\mathcal{SK}_{\mathbf{id}} = (\mathcal{S}_1,\mathcal{S}_2)$ is returned.

PSFKeyGen$(\mathcal{MSK},\mathcal{SK}_{\mathbf{id}})$: Returns a key $\mathcal{SK}_{\mathbf{id}}$ for identity $\mathbf{id}$ with $\mathcal{S}_1$-components having semi-functional terms (generated according to $\mathcal{JR}$-$\mathcal{AHIBE}$.SFKeyGen algorithm) and $\mathcal{S}_2$-components being normal (as returned by $\mathcal{JR}$-$\mathcal{AHIBE}$.KeyGen algorithm).

The basic ideas underlying the security proof (Section 6.2.4.1) of $\mathcal{LW}$-$\mathcal{AHIBE}$ are applicable even in case of $\mathcal{JR}$-$\mathcal{AHIBE}$. It is straightforward to see that decryption of a semi-functional ciphertext by a normal key or that of a normal ciphertext with a semi-functional key succeeds. When both ciphertext and key are semi-functional, decryption results in an extra masking factor of $e(P_1,P_2)^{\gamma\mu(\mathsf{tag}+\pi)}$ on the message. Decryption is only successful if $\pi = -\mathsf{tag}$ whence the ciphertext and key become *nominally semi-functional.*

The following theorem states precisely the security guarantee we obtain for $\mathcal{JR}$-$\mathcal{AHIBE}$.

Theorem 6.5.1. *If* $(\varepsilon_{\text{DDH1}},t_1)$-DDH1 *and* $(\varepsilon_{\text{DDH2}},t_2)$-DDH2 *assumptions hold in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$ *respectively, then* $\mathcal{JR}$-$\mathcal{AHIBE}$ *is* (ε,t)-ANO-IND-ID-CPA-*secure where* $\varepsilon \le \varepsilon_{\text{DDH1}} + 2q\cdot\varepsilon_{\text{DDH2}} + (2q/p)$, $t_1 = t + O(h\rho)$ *and* $t_2 = t + O(h\rho)$. ρ *is the maximum time required for one scalar multiplication in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$.

Proof Sketch. Fix any t-time adversary $\mathscr{A}$. Let G_{real} denote the HIBE security game ano-ind-cpa (described in Section 2.2.1.3) and G_{final} be a game where all keys are semi-functional and the challenge ciphertext is a semi-functional encryption of a random message to a random identity vector. The probability that $\mathscr{A}$ wins in G_{final} is 1/2. To prove the theorem, we need to show a bound on $\mathsf{Adv}^{\text{ano-ind-cpa}}_{\mathcal{JR}\text{-}\mathcal{AHIBE}}(\mathscr{A}) = |\Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{real}] - (1/2)|$ which is equivalent to bounding $|\Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{real}] - \Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{final}]|$. In order to obtain this bound, we first define a sequence of games starting from G_{real} and making small changes until we reach G_{final}. Define $\mathsf{G}_{k,0}$, $1 \le k \le q$ similar to G_{real} except that challenge ciphertext is semi-functional, first $k-1$ keys are semi-functional and k-th key is partial semi-functional. In $\mathsf{G}_{k,1}$, $0 \le k \le q$, the challenge ciphertext is semi-functional and first k keys are semi-functional. The game sequence is G_{real}, $\mathsf{G}_{0,1}$, $(\mathsf{G}_{k,0}, \mathsf{G}_{k,1})_{k=1}^{q}$, G_{final}. The advantage of $\mathscr{A}$ in winning G_{real} can now be bounded in terms of its advantage in distinguishing between successive games. This is done via reductions from the SXDH problem to the task of distinguishing between successive games. Essentially, there are two kinds of reductions - first and second. In the first reduction, we show that $\mathscr{A}$'s ability to distinguish between G_{real} and $\mathsf{G}_{0,1}$ can be used to solve a DDH1 instance. The second reduction shows that an algorithm $\mathscr{A}$ that can distinguish between $\mathsf{G}_{k-1,1}$ and $\mathsf{G}_{k,0}$ for some $k \in [1, q]$, can be used to construct an algorithm $\mathscr{B}_2$ solving DDH2. Similar arguments hold for all values of k and also for the transition from $\mathsf{G}_{k,0}$ to $\mathsf{G}_{k,1}$. The final transition i.e, $\mathsf{G}_{q,1}$ to G_{final} is done just by changing the way information provided to $\mathscr{A}$ is generated so that the distribution of $\mathscr{A}$'s view in the two games are statistically indistinguishable except with probability $2q/p$.. We now provide an outline of each stage in the proof.

First Reduction: Suppose that $\mathscr{B}_1$ is a DDH1-solver. $\mathscr{B}_1$ simulates the game using a DDH1 instance $(\mathcal{G}, P_1, bP_1, sbP_1, P_2, (s+\mu)P_1)$. The element b of the instance corresponds to the scalar b of the scheme. $\mathscr{B}_1$ sets up the system normally since it has all information required to do so. The master secret is also known since none of its components depend on b. Furthermore, it cannot create semi-functional keys as no encoding of b in $\mathbb{G}_2$ is provided. All the key extract queries are answered normally. $\mathscr{B}_1$ sets the randomiser for the challenge ciphertext $\widehat{\mathcal{C}}$ to be s (from the instance). $\widehat{\mathcal{C}}$ will be normal or semi-functional depending on whether the instance is real i.e., $\mu = 0$, or random ($\mu \xleftarrow{\text{U}} \mathbb{Z}_p$).

Second Reduction: The DDH2-solver $\mathscr{B}_2$ obtains an instance $(\mathcal{G}, P_1, P_2, rP_2, cP_2, (rc+\gamma)P_2)$. Here c corresponds to the scalar c in $\mathcal{MSK}$. Elements $d, (e_j)_{j\in[1,h]}$ are set to random degree-1 polynomials in c. Scalar b is chosen randomly from $\mathbb{Z}_p^{\times}$. Let $\mathbf{y} = (d, e_1, \ldots, e_h)$. The public parameters are created differently since $\mathbf{y}$ is not known. Only its encoding in $\mathbb{G}_2$ i.e, $\mathbf{y}P_2$ is known. Specifically $U_1, V_{1,j}, W_1$ are chosen at random from $\mathbb{G}_1$. Depending on these and $\mathbf{y}$, the corresponding Δ's are implicitly set. Encodings of Δ's can be computed only in $\mathbb{G}_2$. This enables normal key generation as well as semi-functional key generation. In its response to the k-th key extract query, $\mathscr{B}_2$ maps r from the instance to the randomiser r_1 in the key. Accordingly it generates the key choosing r_2 at random. If $\gamma = 0$, the key will be normal. Otherwise the key is partial semi-functional and γ corresponds to the randomiser γ_1 in the semi-functional part. Moreover, a linear polynomial $f(\mathbf{id}_k)$ in $\mathbf{id}_k$-components is embedded in the semi-functional scalar π. This polynomial is determined by the co-efficients of c in $\mathbf{y}$. The coefficients of c in e_j also determine π_j respectively. For the challenge ciphertext, $\mathscr{B}_2$ has to create semi-functional components which depend on $\mathbf{y}$. But $\mathbf{y}$ depends on c and encoding of c in $\mathbb{G}_1$ is not known. The only way out is to set $\mathsf{tag} = -f(\widehat{\mathbf{id}}_\beta)$ so that terms depending on c vanish. A consequence is that $\mathscr{B}_2$ can only generate nominally semi-functional ciphertext for $\mathbf{id}_k$. We then argue that the simulation is perfect.

Final Transition: It is required to show that $\mathsf{G}_{q,1}$ and G_{final} are statistically indistinguishable from the attacker's point of view except for probability at most $2q/p$. The generation of public parameters and keys provided to $\mathscr{A}$ are changed ensuring that their form is equivalent to that in $\mathsf{G}_{q,1}$ and they are independent of the scalars $u, d, (e_j)_{j\in[1,h]}$. Consequently the challenge ciphertext is the only place where these scalars come into play, especially in those components that consist of the identity-hash and the message. Basically, the message and the id-hash are masked by random quantities so that G_{final} is simulated.

Proof. Proof of Theorem 6.5.1

Consider a sequence of games G_{real}, $\mathsf{G}_{0,1}$, $(\mathsf{G}_{k,0}, \mathsf{G}_{k,1})_{k=1}^{q}$, G_{final} between an adversary $\mathscr{A}$ and a challenger with the games defined as follows.

- G_{real}: the actual HIBE security game ano-ind-cpa (described in Section 2.2.1.3).
- $\mathsf{G}_{k,0}$, $1 \le k \le q$: challenge ciphertext is semi-functional; first $k-1$ keys are semi-functional and k-th key is partial semi-functional.
- $\mathsf{G}_{k,1}$, $0 \le k \le q$: challenge ciphertext is semi-functional; first k keys are semi-functional.
- G_{final}: challenge ciphertext is a semi-functional encryption of a random message under a random identity vector; all keys are semi-functional.

Let $X_\square$ denote the event that $\mathscr{A}$ wins in $\mathsf{G}_\square$. Clearly, the bit β is statistically hidden from the attacker in G_{final}, which means that $\Pr[X_{final}] = 1/2$.

In Lemmas 6.5.1, 6.5.2, 6.5.3 and 6.5.4, we show that

- $|\Pr[X_{real}] - \Pr[X_{0,1}]| \le \varepsilon_{\text{DDH1}}$,
- $|\Pr[X_{k-1,1}] - \Pr[X_{k,0}]| \le \varepsilon_{\text{DDH2}}$,
- $|\Pr[X_{k,0}] - \Pr[X_{k,1}]| \le \varepsilon_{\text{DDH2}}$,
- $|\Pr[X_{q,1}] - \Pr[X_{final}]| \le 2q/p$.

The advantage of $\mathscr{A}$ in breaking the security of $\mathcal{JR}$-$\mathcal{AHIBE}$ is thus given by

$$\begin{aligned}
\mathsf{Adv}^{\text{ano-ind-cpa}}_{\mathcal{JR}\text{-}\mathcal{AHIBE}}(\mathscr{A}) &= |\Pr[X_{real}] - \frac{1}{2}| \\
&= |\Pr[X_{real}] - \Pr[X_{final}]| \\
&\le |\Pr[X_{real}] - \Pr[X_{0,1}]| + \sum_{k=1}^{q}(|\Pr[X_{k-1,1}] - \Pr[X_{k,0}]| + |\Pr[X_{k,0}] - \Pr[X_{k,1}]|) \\
&\quad + |\Pr[X_{q,1}] - \Pr[X_{final}]| \\
&\le \varepsilon_{\text{DDH1}} + 2q\varepsilon_{\text{DDH2}} + \frac{2q}{p}.
\end{aligned}$$

□

In the sequel, $\mathscr{B}_1$ (resp. $\mathscr{B}_2$) is a DDH1-solver (resp. DDH2-solver). We argue that $\mathscr{B}_1$, using the adversary's ability to distinguish between G_{real} and $\mathsf{G}_{0,1}$, can solve DDH1. Similarly, $\mathscr{A}$'s power to distinguish between $\mathsf{G}_{k-1,1}$ and $\mathsf{G}_{k,0}$ (or $\mathsf{G}_{k,0}$ and $\mathsf{G}_{k,1}$) for $k \in [1,q]$, can be leveraged to build a DDH2-solver $\mathscr{B}_2$.

Lemma 6.5.1. $|\Pr[X_{real}] - \Pr[X_{0,1}]| \le \varepsilon_{\text{DDH1}}$.

Proof. Let $(\mathcal{G}, P_1, bP_1, sbP_1, P_2, (s+\mu)P_1)$ be the instance of DDH1 that $\mathscr{B}_1$ has to solve i.e., decide whether $\mu = 0$ or $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$. The phases of the game are simulated by $\mathscr{B}_1$ as described below.

Setup: Choose $c, d, u, \Delta_1, \Delta_3, \Delta_4, (e_j, \Delta_{2,j})_{j=1}^{h} \xleftarrow{\text{U}} \mathbb{Z}_p$ and set parameters as:

$U_1 = -\Delta_1(bP_1) + dP_1$, $V_{1,j} = -\Delta_{2,j}(bP_1) + e_jP_1$ for $j = 1, \dots, h$, $W_1 = -\Delta_3(bP_1) + cP_1$,
$g_T = e(bP_1, P_2)^{-\Delta_4} e(P_1, P_2)^u$
$\mathcal{PP} : (P_1, bP_1, U_1, (V_{1,j})_{j=1}^{h}, W_1, g_T)$

All the secret scalars present in the $\mathcal{MSK}$ are known. $\mathscr{B}_1$ can thus create normal keys. However, $\mathscr{B}_1$'s lack of knowledge of the scalar b or its encoding in $\mathbb{G}_2$ does not allow it to create semi-functional keys.

Key Generation Phases 1 & 2: $\mathscr{B}_1$ answers all of $\mathscr{A}$'s queries with normal keys generated by the $\mathscr{JR}$-$\mathscr{AHIBE}$.KeyGen algorithm.

Challenge: $\mathscr{A}$ sends two message-identity pairs $(m_0, \widehat{\mathbf{id}}_0), (m_1, \widehat{\mathbf{id}}_1)$. $\mathscr{B}_1$ chooses $\beta \xleftarrow{\text{U}} \{0,1\}$, encrypts M_β under $\widehat{\mathbf{id}}_\beta$ and sends the resulting ciphertext $\widehat{\mathcal{C}} = (\widehat{C}_0, \widehat{C}_1, \widehat{C}_2, \widehat{C}_3, \widehat{\mathsf{tag}})$ to $\mathscr{A}$. Let $\widehat{\mathbf{id}}_\beta = (\widehat{\mathsf{id}}_1, \dots, \widehat{\mathsf{id}}_{\widehat{\ell}})$. $\widehat{\mathcal{C}}$ is computed as:

$$
\begin{aligned}
&\widehat{\mathsf{tag}} \xleftarrow{\text{U}} \mathbb{Z}_p,\\
&\widehat{C}_0 = M_\beta \cdot e(sbP_1, P_2)^{-\Delta_4} e((s+\mu)P_1, P_2)^u = M_\beta \cdot g_T^s e(P_1, P_2)^{u\mu},\\
&\widehat{C}_1 = (s+\mu)P_1 = sP_1 + \mu P_1,\\
&\widehat{C}_2 = sbP_1,\\
&\widehat{C}_3 = (-\Delta_1 - \textstyle\sum_{j=1}^{\widehat{\ell}} \Delta_{2,j}\widehat{\mathsf{id}}_j - \widehat{\mathsf{tag}} \cdot \Delta_3)(sbP_1) + (d + \sum_{j=1}^{\widehat{\ell}} e_j \widehat{\mathsf{id}}_j + \widehat{\mathsf{tag}} \cdot c)(s+\mu)P_1\\
&\quad = (-\Delta_1 b + d + \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j(-\Delta_{2,j}b + e_j) + \widehat{\mathsf{tag}}(-\Delta_3 b + c))(sP_1) + (d + \sum_{j=1}^{\widehat{\ell}} e_j\widehat{\mathsf{id}}_j + \widehat{\mathsf{tag}} \cdot c)(\mu P_1)\\
&\quad = s(U_1 + \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j V_{1,j} + \widehat{\mathsf{tag}} W_1) + \mu(d + \sum_{j=1}^{\widehat{\ell}} e_j \widehat{\mathsf{id}}_j + \widehat{\mathsf{tag}} \cdot c)P_1.
\end{aligned}
$$

Observe that $\widehat{\mathcal{C}}$ is normal if $\mu = 0$ and semi-functional when $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$.

Guess: $\mathscr{A}$ outputs its guess β' and halts.

$\mathscr{B}$ returns 1 if $\mathscr{A}$'s guess is correct i.e., $\beta = \beta'$; otherwise $\mathscr{B}_1$ returns 0. The advantage of $\mathscr{B}_1$ in solving the DDH1 instance is given by

$$
\begin{aligned}
\mathsf{Adv}_{\mathcal{G}}^{\text{DDH1}}(\mathscr{B}_1) &= |\Pr[\mathscr{B}_1 \text{ returns } 1|\mu = 0] - \Pr[\mathscr{B}_1 \text{ returns } 1|\mu \xleftarrow{\text{U}} \mathbb{Z}_p]|\\
&= |\Pr[\beta = \beta'|\mu = 0] - \Pr[\beta = \beta'|\mu \xleftarrow{\text{U}} \mathbb{Z}_p]|\\
&= |\Pr[X_{real}] - \Pr[X_{0,1}]|.
\end{aligned}
$$

Since $\mathsf{Adv}_{\mathcal{G}}^{\text{DDH1}}(\mathscr{B}_1) \le \varepsilon_{\text{DDH1}}$, we have $|\Pr[X_{real}] - \Pr[X_{0,1}]| \le \varepsilon_{\text{DDH1}}$. □

Lemma 6.5.2. $|\Pr[X_{k-1,1}] - \Pr[X_{k,0}]| \le \varepsilon_{\text{DDH2}}$.

Proof. $\mathscr{B}_2$ is given an instance $(\mathcal{G}, P_1, P_2, rP_2, cP_2, (rc+\gamma)P_2)$ of DDH2 and asked to decide whether $\gamma = 0$ or $\gamma \xleftarrow{\text{U}} \mathbb{Z}_p$. It simulates the game as described below.

Setup: Pick scalars $u, \Delta'_1, \Delta'_3, \Delta'_4, d_1, d_2, (e_{j,1}, e_{j,2}, \Delta'_{2,j})_{j=1}^h \xleftarrow{\text{U}} \mathbb{Z}_p$ and $b \xleftarrow{\text{U}} \mathbb{Z}_p^\times$ and (implicitly) set

$$d = d_1 + cd_2, \quad \Delta_1 = \frac{\Delta'_1 + d}{b}, \quad \Delta_3 = \frac{\Delta'_3 + c}{b}, \quad \Delta_4 = \frac{\Delta'_4 + u}{b},$$

$$e_j = e_{j,1} + ce_{j,2}, \quad \Delta_{2,j} = \frac{\Delta'_{2,j} + e_j}{b} \quad \text{for } j = 1, \ldots, h.$$

Parameters are generated as follows.

$U_1 = -\Delta'_1 P_1$, $V_{1,j} = -\Delta'_{2,j} P_1$ for $j = 1, \ldots, h$, $W_1 = -\Delta'_3 P_1$,
$g_T = e(P_1, P_2)^{-\Delta'_4}$
$\mathcal{PP} : (P_1, bP_1, U_1, (V_{1,j})_{j=1}^h, W_1, g_T)$

The elements $\Delta_1, \Delta_{2,j}, \Delta_3, d, e_j$ that are part of the $\mathcal{MSK}$ are not available to $\mathscr{B}_2$. Even without these, $\mathscr{B}_2$ can generate keys as explained in the simulation of the key generation phases.

Key Generation Phases 1 & 2: $\mathscr{A}$ queries on identities $\mathbf{id}_1, \mathbf{id}_2, \ldots, \mathbf{id}_q$. $\mathscr{B}$ responds to the i-th query ($i \in [1, q]$) considering three cases.

Case 1: $i > k$

$\mathscr{B}_2$ returns a normal key, $\mathcal{SK}_{\mathbf{id}_i} = (\mathcal{S}_1, \mathcal{S}_2)$ with $\mathcal{S}_1 = ((K_i)_{i\in[1,5]}, (D_{1,j}, E_{1,j})_{j\in[\ell+1,h]})$ and $\mathcal{S}_2 = ((J_i)_{i\in[1,5]}, (D_{2,j}, E_{2,j})_{j\in[\ell+1,h]})$. The master secret is not completely available to $\mathscr{B}_2$ and hence the $\mathscr{JR}$-$\mathscr{AHIBE}$.KeyGen needs a modification. The $\mathcal{S}_1$-components are computed as shown below.

$$r_1, r_2 \xleftarrow{\text{U}} \mathbb{Z}_p,$$

$$K_1 = r_1 P_2, \ K_2 = r_1(cP_2),$$

$$K_3 = \left(u + r_1\left(d_1 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,1}\right)\right) P_2 + r_1\left(d_2 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,2}\right)(cP_2) = \left(u + r_1\left(d + \sum_{j=1}^{\ell} \mathsf{id}_j e_j\right)\right) P_2,$$

$$K_4 = -b^{-1} r_1(\Delta'_3 P_2 + cP_2) = -r_1\left(\frac{\Delta'_3 + c}{b}\right) P_2 = -r_1 \Delta_3 P_2,$$

$$\begin{aligned} K_5 &= -b^{-1}\left(\Delta'_4 + u + r_1\left(\Delta'_1 + d_1 + \sum_{j=1}^{\ell} \mathsf{id}_j(\Delta'_{2,j} + e_{j,1})\right)\right) P_2 - b^{-1} r_1\left(d_2 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,2}\right)(cP_2) \\ &= b^{-1}\left(-\Delta'_4 - u - r_1\left(\Delta'_1 + d + \sum_{j=1}^{\ell} \mathsf{id}_j(\Delta'_{2,j} + e_j)\right)\right) P_2 \\ &= \left(-\frac{\Delta'_4 + u}{b} - r_1\left(\frac{\Delta'_1 + d}{b} + \sum_{j=1}^{\ell} \mathsf{id}_j\left(\frac{\Delta'_{2,j} + e_j}{b}\right)\right)\right) P_2 \\ &= \left(-\Delta_4 - r_1\left(\Delta_1 + \sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j}\right)\right) P_2, \end{aligned}$$

for $j = \ell + 1, \ldots, h$,

$$D_{1,j} = r_1(e_{j,1}P_2 + e_{j,2}(cP_2)) = r_1 e_j P_2,$$

$$E_{1,j} = -r_1 b^{-1}(\Delta'_{2,j} + e_{j,1})P_2 - r_1 b^{-1} e_{j,2}(cP_2) = -r_1\left(\frac{\Delta'_{2,j} + e_j}{b}\right)P_2 = -r_1\Delta_{2,j}P_2.$$

$\mathcal{S}_2$-components are generated in a similar fashion using a randomiser $r_2 \xleftarrow{\text{U}} \mathbb{Z}_p$ and leaving out the scalars u and Δ'_4. Details are omitted.

Case 2: $i < k$

In this case, $\mathscr{B}_2$ first creates a normal key $\mathcal{SK}_{\mathbf{id}_i}$ and runs $\mathcal{JR}$-$\mathcal{AHIBE}$.SFKeyGen on $\mathcal{SK}_{\mathbf{id}_i}$. This is possible because the only scalar used in $\mathcal{JR}$-$\mathcal{AHIBE}$.SFKeyGen is b which is known to $\mathscr{B}_2$.

Case 3: $i = k$

Let $\mathcal{SK}_{\mathbf{id}_k} = (\mathcal{S}_1, \mathcal{S}_2)$ be the key that $\mathscr{B}_2$ generates for $\mathbf{id}_k$. Elements of $\mathcal{S}_2$ are created normally (as indicated in **Case 1**). In the $\mathcal{S}_1$-portion of $\mathcal{SK}_{\mathbf{id}_k}$, $\mathscr{B}_2$ embeds the DDH2 instance (consisting of $P_2, cP_2, rP_2, (rc+\gamma)P_2$) by generating the components as:

$$K_1 = rP_2, \; K_2 = (rc+\gamma)P_2,$$

$$\begin{aligned}
K_3 &= uP_2 + \left(d_1 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,1}\right)(rP_2) + \left(d_2 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,2}\right)(rc+\gamma)P_2 \\
&= uP_2 + r\left(d_1 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,1} + c\left(d_2 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,2}\right)\right)P_2 + \gamma\left(d_2 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,2}\right)P_2 \\
&= \left(u + r\left(d + \sum_{j=1}^{\ell} \mathsf{id}_j e_j\right)\right)P_2 + \gamma\left(d_2 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,2}\right)P_2,
\end{aligned}$$

$$K_4 = -b^{-1}(\Delta'_3 rP_2 + (rc+\gamma)P_2) = -r\left(\frac{\Delta'_3 + c}{b}\right)P_2 - \left(\frac{\gamma}{b}\right)P_2 = -r\Delta_3 P_2 - \left(\frac{\gamma}{b}\right)P_2,$$

$$\begin{aligned}
K_5 &= -b^{-1}\left(\Delta'_1 + d_1 + \sum_{j=1}^{\ell} \mathsf{id}_j(\Delta'_{2,j} + e_{j,1})\right)(rP_2) - b^{-1}\left(d_2 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,2}\right)(rc+\gamma)P_2 \\
&= -b^{-1}r\left(\Delta'_1 + d + \sum_{j=1}^{\ell} \mathsf{id}_j(\Delta'_{2,j} + e_j)\right)P_2 - b^{-1}\gamma\left(d_2 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,2}\right)P_2 \\
&= -r\left(\frac{\Delta'_1 + d}{b} + \sum_{j=1}^{\ell} \mathsf{id}_j\left(\frac{\Delta'_{2,j} + e_j}{b}\right)\right)P_2 - \left(\frac{\gamma}{b}\right)\left(d_2 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,2}\right)P_2 \\
&= -r\left(\Delta_1 + \sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j}\right)P_2 - \left(\frac{\gamma}{b}\right)\left(d_2 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,2}\right)P_2,
\end{aligned}$$

for $j = \ell + 1, \ldots, h$,

$$D_{1,j} = e_{j,1}(rP_2) + e_{j,2}(rc+\gamma)P_2 = re_jP_2 + \gamma e_{j,2}P_2,$$

$$E_{1,j} = -b^{-1}(\Delta'_{2,j} + e_{j,1})rP_2 - b^{-1}e_{j,2}(rc+\gamma)P_2$$

$$= -r\left(\frac{\Delta'_{2,j} + e_j}{b}\right)P_2 - \left(\frac{\gamma e_{j,2}}{b}\right)P_2$$
$$= -r\Delta_{2,j}P_2 - \left(\frac{\gamma e_{j,2}}{b}\right)P_2.$$

When $\gamma = 0$, $\mathcal{SK}_{\mathbf{id}_k}$ is normal with $r_1 = r$; otherwise, it is partial semi-functional with

$r_1 = r$, $\gamma_1 = \gamma$,
$\pi = d_2 + \sum_{j=1}^{\ell} \mathsf{id}_j e_{j,2}$ and
$\pi_j = e_{j,2}$ for $j = \ell + 1, \ldots, h$

set implicitly.

Challenge: $\mathscr{B}_2$ obtains two message-identity pairs $(m_0, \widehat{\mathbf{id}}_0), (m_1, \widehat{\mathbf{id}}_1)$ from $\mathscr{A}$. It then picks $\beta \xleftarrow{\text{U}} \{0,1\}$, $s, \mu \xleftarrow{\text{U}} \mathbb{Z}_p$ and generates a semi-functional encryption of M_β under $\widehat{\mathbf{id}}_\beta = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$ given by $\widehat{\mathcal{C}} = (\widehat{C}_0, \widehat{C}_1, \widehat{C}_2, \widehat{C}_3, \widehat{\mathsf{tag}})$ where

$$\widehat{\mathsf{tag}} = -d_2 - \sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j e_{j,2},$$
$$\widehat{C}_0 = M_\beta \cdot g_T^s \cdot e(P_1, P_2)^{u\mu},$$
$$\widehat{C}_1 = sP_1 + \mu P_1,$$
$$\widehat{C}_2 = sbP_1,$$
$$\begin{aligned}
\widehat{C}_3 &= s\left(U_1 + \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j V_{1,j} + \widehat{\mathsf{tag}} W_1\right) + \mu\left(d_1 + \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j e_{j,1}\right)P_1 \\
&= s\left(U_1 + \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j V_{1,j} + \widehat{\mathsf{tag}} W_1\right) \\
&\quad + \mu\left((d_1 + cd_2) + \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j (e_{j,1} + ce_{j,2}) + \widehat{\mathsf{tag}} \cdot c\right)P_1 - \mu\left(d_2 c + \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j e_{j,2} c\right)P_1 - \widehat{\mathsf{tag}} \cdot c\mu P_1 \\
&= s\left(U_1 + \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j V_{1,j} + \widehat{\mathsf{tag}} W_1\right) \\
&\quad + \mu\left(d + \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j e_j + \widehat{\mathsf{tag}} \cdot c\right)P_1 + c\mu\left(-d_2 - \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j e_{j,2} - \widehat{\mathsf{tag}}\right)P_1 \\
&= s\left(U_1 + \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j V_{1,j} + \widehat{\mathsf{tag}} W_1\right) + \mu\left(d + \textstyle\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j e_j + \widehat{\mathsf{tag}} \cdot c\right)P_1.
\end{aligned}$$

The last step follows due to the fact that $\widehat{\mathsf{tag}} = -d_2 - \sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j e_{j,2}$. Note that $\widehat{\mathcal{C}}$ is properly formed. Also, this is the only way $\mathscr{B}_2$ can generate a semi-functional ciphertext since no encoding of c is available in the group $\mathbb{G}_1$. An implication is that $\mathscr{B}_2$ can only create a nominally semi-functional ciphertext for $\mathbf{id}_k$ since the relation $\mathsf{tag} = -\pi$ will hold, thus providing no information to $\mathscr{B}_2$ about the semi-functionality of $\mathcal{SK}_{\mathbf{id}_k}$.

Guess: $\mathscr{A}$ returns its guess β' of β.

$\mathscr{B}_2$ outputs 1 if $\mathscr{A}$ wins and 0 otherwise. Also, $\mathscr{B}_2$ simulates $\mathsf{G}_{k-1,1}$ if $\gamma = 0$ and $\mathsf{G}_{k,0}$ if $\gamma \xleftarrow{\text{U}} \mathbb{Z}_p$. Therefore, the advantage of $\mathscr{B}_2$ in solving the DDH2 instance is given by

$$\begin{aligned}
\mathsf{Adv}_{\mathcal{G}}^{\text{DDH2}}(\mathscr{B}_2) &= |\Pr[\mathscr{B}_2 \text{ returns } 1 | \gamma = 0] - \Pr[\mathscr{B}_2 \text{ returns } 1 | \gamma \xleftarrow{\text{U}} \mathbb{Z}_p]| \\
&= |\Pr[\beta = \beta' | \mu = 0] - \Pr[\beta = \beta' | \mu \xleftarrow{\text{U}} \mathbb{Z}_p]| \\
&= |\Pr[X_{k-1,1}] - \Pr[X_{k,0}]|.
\end{aligned}$$

It now follows that $|\Pr[X_{k-1,1}] - \Pr[X_{k,0}]| \leq \varepsilon_{\text{DDH2}}$ from the fact that $\mathsf{Adv}_{\mathcal{G}}^{\text{DDH2}}(\mathscr{B}) \leq \varepsilon_{\text{DDH2}}$ for all t-time adversaries $\mathscr{B}$. What remains is to show that all the information provided to the adversary have the correct distribution. The scalars $b, u, \Delta'_1, \Delta'_3, \Delta'_4, d_1, d_2, (e_{j,1}, e_{j,2}, \Delta'_{2,j})_{j=1}^h$ chosen by $\mathscr{B}_2$ and r, c, γ from the instance are uniformly and independently distributed. As a consequence the following quantities have the correct distribution.

- r_1, γ_1 for the k-th key
- Δ_4, Δ_3
- $d, (e_j)_{j=1}^h$ and hence $\Delta_1, (\Delta_{2,j})_{j=1}^h$

The same scalars also determine $\pi, (\pi_j)_{j=\ell+1}^h$ for k-th identity and $\widehat{\mathsf{tag}}$ for challenge ciphertext which are required to be uniform and independent quantities. We now argue that this is indeed the case. Let $\mathbf{id}_k = (\mathsf{id}_1, \ldots, \mathsf{id}_h)$ and $\widehat{\mathbf{id}}_\beta = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_h)$ where, for convenience we assume that $\mathsf{id}_{\ell+1} = \cdots = \mathsf{id}_h = \widehat{\mathsf{id}}_{\widehat{\ell}+1} = \cdots \widehat{\mathsf{id}}_h = 0$. The quantities $\pi, (\pi_j)_{j=2}^h, \widehat{\mathsf{tag}}$ are given by the following equation.

$$\begin{pmatrix} \pi \\ \pi_2 \\ \vdots \\ \pi_h \\ \widehat{\mathsf{tag}} \end{pmatrix} = \begin{pmatrix} 1 & \mathsf{id}_1 & \mathsf{id}_2 & \mathsf{id}_3 & \mathsf{id}_4 & \cdots & \mathsf{id}_h \\ 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 1 \\ -1 & -\widehat{\mathsf{id}}_1 & -\widehat{\mathsf{id}}_2 & -\widehat{\mathsf{id}}_3 & -\widehat{\mathsf{id}}_4 & \cdots & -\widehat{\mathsf{id}}_h \end{pmatrix} \begin{pmatrix} d_2 \\ e_{1,2} \\ e_{2,2} \\ \vdots \\ e_{h-1,2} \\ e_{h,2} \end{pmatrix} \tag{6.2}$$

Observe that

- the first and last rows in the above matrix are linearly independent since identity components are in $\mathbb{Z}_p^\times$ and $\mathbf{id}_k \neq \widehat{\mathbf{id}}_\beta$. All other rows are linearly independent of these two rows. Further, since $\ell, \widehat{\ell} >= 1$, we have $\mathsf{id}_1 \neq 0$ and $\widehat{\mathsf{id}}_1 \neq 0$. Hence the matrix has rank $h+1$.
- $d_2, e_{1,2}, \ldots, e_{h,2}$ are information theoretically hidden from $\mathscr{A}$ and also chosen from uniform and independent distributions over $\mathbb{Z}_p$.

Conditioned on these observations, we conclude that $\pi, (\pi_j)_{j=2}^h, \widehat{\mathsf{tag}}$ are uniformly and independently distributed in $\mathscr{A}$'s view. □

Lemma 6.5.3. $|\Pr[X_{k,0}] - \Pr[X_{k,1}]| \leq \varepsilon_{\text{DDH2}}$.

The proof is similar to that of Lemma 6.5.2. The difference is that $\mathscr{B}_2$ creates a partial semi-functional key for $\mathbf{id}_k$, the k-the identity queried by $\mathscr{A}$, and then embeds the DDH2 instance in $\mathcal{S}_2$-portion of the key. $\mathscr{B}_2$ advantage in solving DDH2 will now depend on whether the $\mathscr{A}$ can determine whether $\mathcal{SK}_{\mathbf{id}_k}$ is partial or fully semi-functional.

Lemma 6.5.4. $|\Pr[X_{q,1}] - \Pr[X_{final}]| \leq (2q/p)$.

Proof. In $\mathsf{G}_{q,1}$, all the keys returned to $\mathscr{A}$ are semi-functional and so is the challenge ciphertext. To argue that $\Pr[X_{q,1}] = \Pr[X_{final}]|$, we modify the $\mathcal{JR}$-$\mathcal{AHIBE}$.Setup and $\mathcal{JR}$-$\mathcal{AHIBE}$.SFKeyGen algorithms so that the modification results in G_{final} and the distribution of information provided to the adversary before and after the modification are statistically indistinguishable except with probability $2q/p$.

$\mathcal{JR}$-$\mathcal{AHIBE}$.Setup: Pick scalars $\Delta_1', \Delta_3', \Delta_4', u, c, d, (e_j, \Delta_{2,j}')_{j=1}^{h} \xleftarrow{\text{U}} \mathbb{Z}_p$ and $b \xleftarrow{\text{U}} \mathbb{Z}_p^{\times}$ and compute parameters as:

$U_1 = -\Delta_1' P_1$, $V_{1,j} = -\Delta_{2,j}' P_1$ for $j = 1, \ldots, h$, $W_1 = -\Delta_3' P_1$,
$g_T = e(P_1, P_2)^{-\Delta_4'}$
$\mathcal{PP} : (P_1, bP_1, U_1, (V_{1,j})_{j=1}^{h}, W_1, g_T)$

setting

$$\Delta_1 = \frac{\Delta_1' + d}{b}, \quad \Delta_3 = \frac{\Delta_3' + c}{b}, \quad \Delta_4 = \frac{\Delta_4' + u}{b},$$

$$\Delta_{2,j} = \frac{\Delta_{2,j}' + e_j}{b} \qquad \text{for } j = 1, \ldots, h.$$

$\mathcal{JR}$-$\mathcal{AHIBE}$.SFKeyGen: Choose $r_1, r_2, \pi', \sigma', (\pi_j', \sigma_j')_{j=\ell+1}^{h} \xleftarrow{\text{U}} \mathbb{Z}_p$, $\gamma_1, \gamma_2 \xleftarrow{\text{U}} \mathbb{Z}_p$ and compute the individual components as follows.

$$K_1 = r_1 P_2, \; K_2 = r_1 c P_2 + \gamma_1 P_2, \qquad J_1 = r_2 P_2, \; J_2 = r_2(cP_2) + \gamma_2 P_2,$$

$$K_3 = \pi' P_2, \qquad J_3 = \sigma' P_2,$$

$$K_4 = -r_1 \left(\frac{\Delta_3' + c}{b} \right) P_2 - \left(\frac{\gamma_1}{b} \right) P_2, \qquad J_4 = -r_2 \left(\frac{\Delta_3' + c}{b} \right) P_2 - \left(\frac{\gamma_2}{b} \right) P_2,$$

$$K_5 = -\frac{1}{b} \left(\pi' + \Delta_4' + r_1 \left(\Delta_1' + \sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j}' \right) \right) P_2, \qquad J_5 = -\frac{1}{b} \left(\sigma' + r_2 \left(\Delta_1' + \sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j} \right) \right) P_2,$$

for $j = \ell + 1, \ldots, h$,

$$D_{1,j} = \pi_j' P_2, \qquad D_{2,j} = \sigma_j' P_2,$$

$$E_{1,j} = -\left(\frac{r_1 \Delta_{2,j}' + \pi_j'}{b} \right) P_2, \qquad E_{2,j} = -\left(\frac{r_2 \Delta_{2,j}' + \sigma_j'}{b} \right) P_2.$$

The setting of $K_3 = \pi' P_2$ fixes the product $\gamma_1 \pi$ that appears in its semi-functional form i.e., $\left(u + r_1 \left(d + \sum_{j=1}^{\ell} \mathsf{id}_j e_j\right) + \gamma_1 \pi\right) P_2$. The other component where π' is used is K_5 that also fixes $\gamma_1 \pi$ in its semi-functional term. It is necessary to ensure that these two are equal. We show below that

K_5 is indeed well-formed in this sense.

$$\begin{aligned}
K_5 &= -\frac{1}{b}\left(\pi' + \Delta_4' + r_1\left(\Delta_1' + \sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j}'\right)\right)P_2 \\
&= -\frac{1}{b}\left(u + r_1\left(d + \sum_{j=1}^{\ell} \mathsf{id}_j e_j\right) + \gamma_1\pi + \Delta_4' + r_1\left(\Delta_1' + \sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j}'\right)\right)P_2 \\
&= -\frac{1}{b}\left((\Delta_4' + u) + r_1\left((\Delta_1' + d) + \sum_{j=1}^{\ell} \mathsf{id}_j(\Delta_{2,j}' + e_j)\right)\right)P_2 + \gamma_1\pi P_2 \\
&= -\left(\frac{\Delta_4' + u}{b} + r_1\left(\frac{\Delta_1' + d}{b} + \sum_{j=1}^{\ell} \mathsf{id}_j\left(\frac{\Delta_{2,j}' + e_j}{b}\right)\right)\right)P_2 + \gamma_1\pi P_2 \\
&= -\left(\Delta_4 + r_1\left(\Delta_1 + \sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j}\right)\right)P_2 + \gamma_1\pi P_2,
\end{aligned}$$

Similarly, setting $D_{1,j} = \pi_j' P_2$ fixes $\gamma_1\pi_j$ since $D_{1,j}$ has the form $r_1 e_j + \gamma_1\pi_j$. $E_{1,j}$ is computed using π_j' and we justify below that is is properly formed.

$$\begin{aligned}
E_{1,j} &= -\left(\frac{r_1\Delta_{2,j}' + \pi_j'}{b}\right)P_2 \\
&= -\left(\frac{r_1\Delta_{2,j}' + r_1 e_j + \gamma_1\pi_j}{b}\right)P_2 \\
&= -r_1\left(\frac{\Delta_{2,j}' + e_j}{b}\right)P_2 - \frac{\gamma_1\pi_j}{b}P_2 \\
&= -r_1\Delta_{2,j}P_2 - \frac{\gamma_1\pi_j}{b}P_2
\end{aligned}$$

The scalars $\pi', (\pi_j')_{j=1}^{h}$ define the products $\gamma_1\pi, (\gamma_1\pi_j)_{j=1}^{h}$ respectively. Since γ_1 is chosen uniformly from $\mathbb{Z}_p$, $\pi, (\pi_j)_{j=1}^{h}$ are uniformly and independently distributed in $\mathbb{Z}_p$ except when $\gamma_1 = 0$. Similarly, it is possible to show that $J_5, (E_{2,j})_{j=\ell+1}^{h}$ are well-formed and $\sigma, (\sigma_j)_{j=1}^{h}$ have the proper distribution given that $\gamma_2 \neq 0$. Furthermore, all elements of the key are generated independent of $u, d, (e_j)_{j=1}^{h}$ that determines the independence of the ciphertext from the key. Let us now take a look at the challenge ciphertext:

$\widehat{C}_0 = M_\beta \cdot g_T^s \cdot e(P_1, P_2)^{u\mu}$,
$\widehat{C}_1 = sP_1 + \mu P_1$,
$\widehat{C}_2 = sbP_1$,
$\widehat{C}_3 = -s\left(\Delta_1' + \sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j\Delta_{2,j}' + \widehat{\mathsf{tag}}\Delta_3'\right)P_1 + \mu\left(d + \sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j e_j + \widehat{\mathsf{tag}} \cdot c\right)P_1$,

where $\widehat{\mathsf{tag}}, \mu, s \xleftarrow{\text{U}} \mathbb{Z}_p$. Recall that $u, d, (e_j)_{j=1}^{h}$ are chosen independently and uniformly at random from $\mathbb{Z}_p$. Consequently, components $\widehat{C}_0$ and $\widehat{C}_1$ are randomly distributed in $\mathbb{G}_T$ and $\mathbb{G}_1$ respectively. Also these two components are independent of all other information (including keys and public parameters) provided to $\mathscr{A}$. Therefore the bit β is information theoretically hidden from the adversary implying that the resulting game (obtained by modifying $\mathcal{JR}$-$\mathcal{AHIBE}$.SFKeyGen) is G_{final}. The distribution of public parameters remains the unchanged. Let F_i denote the event that $\gamma_1 = 0$

or $\gamma_2 = 0$ for an extract query on $\mathbf{id}_i$ (for $i \in [1, q]$). Clearly $\Pr[\mathsf{F}_i] \le 2/p$. The keys have the correct distribution unless the event $\mathsf{F} = \cup_{i=1}^{q} \mathsf{F}_i$ occurs. Thus we have $|\Pr[X_{q,1}] - \Pr[X_{final}]| \le \Pr[\mathsf{F}] \le \sum_{i=1}^{q} \Pr[\mathsf{F}_i] = 2q/p$. $\square$

6.5.1 Scheme $\mathcal{JR}$-$\mathcal{HIBE}$

This section presents the second (non-anonymous) HIBE construction. As discussed in Section 6.4, two sub-hashes in the key are combined to form the identity-hash required for cancellation with the ciphertext. The sub-hashes are determined by the vectors $\mathbf{v}_1 = (d, e_1, \ldots, e_h)$ and $\mathbf{v}_2 = (\Delta_1, \Delta_{2,1}, \ldots, \Delta_{2,h})$. In order to realise anonymity, these vectors are kept as part of the master secret in $\mathcal{JR}$-$\mathcal{AHIBE}$. Additional elements had to be provided in the key to enable rerandomisation during delegation. It turns out that we can obtain a non-anonymous scheme by making these vectors public. The availability of these vectors facilitates rerandomisation and hence the keys no longer need extra components for this purpose. As a result, keys are shorter and algorithms KeyGen, Delegate are more efficient in comparison to $\mathcal{JR}$-$\mathcal{AHIBE}$.

The method followed here in obtaining a non-anonymous HIBE did not work out for previously known anonymous HIBE schemes [107] and $\mathcal{LW}$-$\mathcal{AHIBE}$. This is due to the following reasons. The element $\mathbb{G}_T$ would be of the form $e(P_1, P_2)^{\alpha}$ where α is part of the master secret. P_1 and P_2 would be required for encryption and delegation respectively as a result of which both P_1 and P_2 would be present in $\mathcal{PP}$. However, this leaks α information theoretically thus revealing the message too! The splitting of α here in terms of Δ_4 and u precisely overcomes this problem. These scalars further provide the randomness required to generate semi-functional components.

Regarding the dual system proof, we mentioned in Section 6.4 that some elements in the master secret provide the randomness required to generate semi-functional components during simulation. In $\mathcal{JR}$-$\mathcal{HIBE}$, the scalars $d, c, (e_j)_{j \in [1,h]}$ are revealed information theoretically in the public parameters. Although d, c, e_j being hidden provides more randomness, they are not essential to generating the required amount of randomness in the proof. The scalar u, hidden by $\mathbb{G}_T$ in the public parameters, is sufficient.

We now provide a definition of $\mathcal{JR}$-$\mathcal{HIBE}$ = ($\mathcal{JR}$-$\mathcal{HIBE}$.Setup, $\mathcal{JR}$-$\mathcal{HIBE}$.Encrypt, $\mathcal{JR}$-$\mathcal{HIBE}$.KeyGen, $\mathcal{JR}$-$\mathcal{HIBE}$.Delegate, $\mathcal{JR}$-$\mathcal{HIBE}$.Decrypt) where the algorithms are as follows.

$\mathcal{JR}$-$\mathcal{HIBE}$.Setup(κ): Generate a Type-3 pairing $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, F_1, F_2)$ based on the security parameter κ. Compute parameters as follows.

$$P_1 \xleftarrow{\text{U}} \mathbb{G}_1^{\times},\ P_2 \xleftarrow{\text{U}} \mathbb{G}_2^{\times}$$
$$\Delta_1, \Delta_3, \Delta_4, c, d, u, (\Delta_{2,j}, e_j)_{j=1}^{h} \xleftarrow{\text{U}} \mathbb{Z}_p,\ b \xleftarrow{\text{U}} \mathbb{Z}_p^{\times},$$

$$U_1 = (-\Delta_1 b + d)P_1,\ V_{1,j} = (-\Delta_{2,j} b + e_j)P_1 \text{ for } j = 1, \ldots, h,\ W_1 = (-\Delta_3 b + c)P_1,$$
$$g_T = e(P_1, P_2)^{-\Delta_4 b + u},$$

$$\mathcal{PP} : (P_1, bP_1, U_1, (V_{1,j})_{j=1}^{h}, W_1, P_2, \Delta_1 P_2, \Delta_3 P_2, dP_2, cP_2, (\Delta_{2,j}P_2, e_j P_2)_{j=1}^{h}, g_T)$$
$$\mathcal{MSK} : (\Delta_4, u)$$

$\mathcal{JR}$-$\mathcal{HIBE}$.Encrypt($\mathcal{PP}, M, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$): Pick $\mathsf{tag}, s \xleftarrow{\text{U}} \mathbb{Z}_p$ and set the ciphertext $\mathcal{C} = (C_0, C_1, C_2, C_3, \mathsf{tag})$ where

$C_0 = M \cdot (g_T)^s$, $C_1 = sP_1$, $C_2 = sbP_1$, $C_3 = s(U_1 + \sum_{j=1}^{\ell} \mathsf{id}_j V_{1,j} + \mathsf{tag} W_1)$.

$\mathcal{JR}$-$\mathcal{HIBE}$.KeyGen($\mathcal{MSK}$, $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$): Pick $r \xleftarrow{\text{U}} \mathbb{Z}_p$ and compute the secret key $\mathcal{SK}_{\mathbf{id}} = ((K_i)_{i\in[1,5]}, (D_j, E_j)_{j\in[\ell+1,h]})$ for $\mathbf{id}$ where,

$K_1 = rP_2$, $K_2 = rcP_2$, $K_3 = \left(u + r(d + \sum_{j=1}^{\ell} \mathsf{id}_j e_j)\right) P_2$,
$K_4 = -r\Delta_3 P_2$, $K_5 = \left(-\Delta_4 - r(\Delta_1 + \sum_{j=1}^{\ell} \mathsf{id}_j \Delta_{2,j})\right) P_2$,
$D_j = re_j P_2$, $E_j = -r\Delta_{2,j} P_2$ for $j = \ell+1, \ldots, h$.

$\mathcal{JR}$-$\mathcal{HIBE}$.Delegate($\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathsf{id}_{\ell+1}$): Let $\mathbf{id} : \mathsf{id}_{\ell+1} = (\mathsf{id}_1, \ldots, \mathsf{id}_{\ell+1})$. $\mathcal{SK}_{\mathbf{id}:\mathsf{id}_{\ell+1}}$ is generated from $\mathcal{SK}_{\mathbf{id}}$ as follows.

$\tilde{r} \xleftarrow{\text{U}} \mathbb{Z}_p^{\times}$,

$K_1 \leftarrow K_1 + \tilde{r}P_2$, $K_2 \leftarrow K_2 + \tilde{r}cP_2$, $K_3 \leftarrow (K_3 + \mathsf{id}_{\ell+1} D_{\ell+1}) + \tilde{r}(d + \sum_{j=1}^{\ell+1} \mathsf{id}_j e_j) P_2$,
$K_4 \leftarrow K_4 - \tilde{r}\Delta_3 P_2$, $K_5 \leftarrow (K_5 + \mathsf{id}_{\ell+1} E_{\ell+1}) - \tilde{r}(\Delta_1 + \sum_{j=1}^{\ell+1} \mathsf{id}_j \Delta_{2,j}) P_2$,
$D_j \leftarrow D_j + \tilde{r}e_j P_2$, $E_j \leftarrow E_j - \tilde{r}\Delta_{2,j} P_2$ for $j = \ell+2, \ldots, h$,

setting $r \leftarrow r + \tilde{r}$. Note that the distribution of $\mathcal{SK}_{\mathbf{id}:\mathsf{id}_{\ell+1}}$ is same as that of a freshly generated key for $\mathbf{id} : \mathsf{id}_{\ell+1}$ via the KeyGen algorithm.

$\mathcal{JR}$-$\mathcal{HIBE}$.Decrypt($\mathcal{C}, \mathcal{SK}_{\mathbf{id}}$): Return M' computed as:

$$M' = \frac{C_0 \cdot e(C_3, K_1)}{e(C_1, \mathsf{tag}K_2 + K_3) e(C_2, \mathsf{tag}K_4 + K_5)}.$$

Note 6.5.1. The encryption and decryption algorithms of $\mathcal{JR}$-$\mathcal{AHIBE}$ and $\mathcal{JR}$-$\mathcal{HIBE}$ are identical and hence the correctness of decryption for $\mathcal{JR}$-$\mathcal{HIBE}$ follows from that of $\mathcal{JR}$-$\mathcal{AHIBE}$.

Note 6.5.2. The KeyGen and Delegate algorithms for $\mathcal{JR}$-$\mathcal{HIBE}$ are identical to the portion of the corresponding algorithms for $\mathcal{JR}$-$\mathcal{AHIBE}$ which modify the $\mathcal{S}_1$-components of the key. The $\mathcal{S}_2$ components of the key in $\mathcal{JR}$-$\mathcal{AHIBE}$ are not required in $\mathcal{JR}$-$\mathcal{HIBE}$.

Discussion. Setting $h = 1$ in $\mathcal{JR}$-$\mathcal{HIBE}$ yields a non-anonymous variant of JR-IBE-D. The resulting IBE has efficiency comparable to JR-IBE-D but has seven extra elements from $\mathbb{G}_2$ in public parameters. It is interesting to note that $\mathcal{JR}$-$\mathcal{HIBE}$ is the only known HIBE within the dual system framework which has rerandomisable keys. The same holds for the corresponding IBE as well.

6.5.2 Security of $\mathcal{JR}$-$\mathcal{HIBE}$ - An Overview

The security of $\mathcal{JR}$-$\mathcal{HIBE}$ is very similar to that of $\mathcal{JR}$-$\mathcal{AHIBE}$. We only highlight the main differences and omit the details of the proof.

The definition of semi-functional ciphertexts remains the same. The semi-functional components in keys are defined as for $\mathcal{S}_1$ in $\mathcal{JR}$-$\mathcal{AHIBE}$. Keys in $\mathcal{JR}$-$\mathcal{HIBE}$ do not contain the second set of components $\mathcal{S}_2$. Hence, the notion of partial semi-functionality is not required.

The game sequence is G_{real}, G_0, $(\mathsf{G}_k)_{k=1}^{q}$, G_{final}, where G_{real} is the actual HIBE CPA-security game ind-cpa (defined in Section 2.2.1.3).In G_0, challenge ciphertext is semi-functional and all keys are normal. G_k, $0 \le k \le q$ is similar to G_0 except that the first k keys are semi-functional and the rest are normal. In G_{final}, challenge ciphertext is a semi-functional encryption of a random message and all keys are semi-functional. The theorem below summarises the exact security guarantee obtained for $\mathcal{JR}$-$\mathcal{HIBE}$.

Theorem 6.5.2. *If* $(\varepsilon_{\text{DDH1}}, t_1)$-DDH1 *and* $(\varepsilon_{\text{DDH2}}, t_2)$-DDH2 *assumptions hold in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$ *respectively, then* $\mathcal{JR}$-$\mathcal{HIBE}$ *is* (ε, t)-IND-ID-CPA*-secure where* $\varepsilon \le \varepsilon_{\text{DDH1}} + q \cdot \varepsilon_{\text{DDH2}} + (q/p)$, $t_1 = t + O(h\rho)$ *and* $t_2 = t + O(h\rho)$. ρ *is the maximum time required for one scalar multiplication in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$.

Since the structure of the ciphertext in $\mathcal{JR}$-$\mathcal{HIBE}$ and $\mathcal{JR}$-$\mathcal{AHIBE}$ are identical, so is the first reduction (based on DDH1). The second reduction is also similar; it is only needed to show that the elements in $\mathbb{G}_2$ that are made public can indeed be generated. The third reduction has one difference. We no longer need to argue about the independence of all information provided to the attacker with respect to the elements $d, (e_j)_{j\in[1,h]}$. In $\mathcal{JR}$-$\mathcal{AHIBE}$, this was required to show anonymity i.e, the hash of the identity is masked by a random quantity. We only need to show that the message to be masked by a random quantity in the last game and this is done by arguing that the adversary's view (excluding the challenge ciphertext) is independent of the scalar u.

6.6 Anonymity and Constant-Size Ciphertexts

As mentioned in the beginning of this chapter, many HIBE schemes (such as [86]) have the prefix decryption property. That is, an entity with identity $\mathbf{id}'$ and a corresponding secret key $\mathcal{SK}_{\mathbf{id}'}$ can decrypt any ciphertext corresponding to $\mathbf{id}$ where $\mathbf{id}'$ is a prefix of $\mathbf{id}$. As an example, consider the Gentry-Silverberg HIBE [86] (GS-HIBE) based on a symmetric pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The ciphertext for an identity $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ consists of points $rP_0, rP_2, \ldots, rP_\ell$ from $\mathbb{G}$ and the n-bit string $M \oplus H_2(e(P_1, Q_0)^r)$ where $H_1 : \{0,1\}^* \to \mathbb{G}$, $H_2 : \mathbb{G}_T \to \{0,1\}^n$ are cryptographic hash functions, $P_i = H_1(\mathsf{id}_1, \ldots, \mathsf{id}_i)$ for $i \in [1, \ell]$, elements P_0, Q_0 come from $\mathcal{PP}$ and M is the message. Note that when we remove the points $rP_{\ell'+1}, \ldots, rP_\ell$ ($\ell' < \ell$), the remaining components form a valid ciphertext for the identity $\mathbf{id}' = (\mathsf{id}_1, \ldots, \mathsf{id}_{\ell'})$ and hence can be decrypted using a secret key for $\mathbf{id}'$. The prefix decryption property holds in this case.

In the non-anonymous setting, $\mathbf{id}$ is known. Hence, decryption can be done via a secret key for $\mathbf{id}$ that is derived from $\mathcal{SK}_{\mathbf{id}'}$ by a sequence of calls to the Delegate algorithm. In case of anonymous HIBE, $\mathbf{id}$ is not known and as a result, delegation is not possible. If the ciphertext contains separate components corresponding to each level of $\mathbf{id}$, prefix decryption would still hold. As shown above (for the case of GS-HIBE), one can truncate the ciphertext retaining only the components corresponding to $\mathbf{id}'$ and then perform decryption using $\mathcal{SK}_{\mathbf{id}'}$. The anonymous HIBE of Okamoto-Takashima[122] also has this property. But in case of an anonymous HIBE where the ciphertext size is constant, as in $\mathcal{JR}$-$\mathcal{AHIBE}$, it is not possible to decrypt the ciphertext with $\mathcal{SK}_{\mathbf{id}'}$. The reason is that is no way to remove (or truncate) the randomised components corresponding $\mathbf{id} \setminus \mathbf{id}'$ from the ciphertext (here, $\mathbf{id} \setminus \mathbf{id}'$ denotes the suffix of $\mathbf{id}'$ in $\mathbf{id}$ i.e., $\mathsf{id}_{\ell'}, \ldots, \mathsf{id}_\ell$). More precisely, given the hash $s(U_1 + \sum_{j=1}^{\ell} \mathsf{id}_j V_{1,j} + \mathsf{tag} W_1)$ for $\mathbf{id}$ it is impossible to extract a hash for $\mathbf{id}'$ since we have no knowledge of $sV_{1,j}$'s. This limitation is not particular to our work and neither is something that arises due to the techniques that we use. It is an inherent limitation and is present in all previously known HIBE constructions which simultaneously achieve constant-size ciphertexts and anonymity.

On the contrary, as pointed out earlier, the absence of prefix decryption property is acceptable and possibly useful in certain applications. For related discussions on this issue, the reader is referred to [86] and [58].

6.7 Detailed Comparison to Existing HIBE Schemes.

Table 6.1 provides a comparison of $\mathcal{JR}$-$\mathcal{HIBE}$ with all previously proposed non-anonymous CC-HIBE schemes. In terms of security, $\mathcal{JR}$-$\mathcal{HIBE}$ is comparable to [122] and [54]. The security of the construction in [114] is based on sub-group decision assumptions that cannot be considered to be standard assumptions. $\mathcal{JR}$-$\mathcal{HIBE}$ achieves the best efficiency compared to all other schemes. Table 6.2 compares $\mathcal{LW}$-$\mathcal{AHIBE}$ and $\mathcal{JR}$-$\mathcal{AHIBE}$ with all previously proposed anonymous HIBE schemes. In terms of security and efficiency, there is no construction that is comparable to $\mathcal{JR}$-$\mathcal{AHIBE}$.

We fix some notation required to compare different parameters of HIBE constructions. h: maximum depth of the HIBE; ℓ: length of the identity tuple; q: number of key extraction queries. In [49], N is the number of bits in an identity and k represents number of blocks of N/k bits. #pp, #msk, #cpr and #key denote number of group elements in the public parameters, master secret, ciphertext and key respectively. Enc, Dec, KGen and Deleg indicate the efficiency of encryption, decryption, key generation and delegation algorithms. For Type-3 pairing based schemes, $\mathcal{PP}$ and ciphertexts consist elements of $\mathbb{G}_1$; $\mathcal{MSK}$ and keys consist elements of $\mathbb{G}_2$. #pp $= (a, b)$ means that there are a elements from $\mathbb{G}_1$, $\mathbb{G}_2$ and b elements of $\mathbb{G}_T$. #cpr $= (a, b)$ denotes a elements from $\mathbb{G}_1$ and b elements from $\mathbb{Z}_p$ where $p = |\mathbb{G}_1|$. We do not consider the $\mathbb{G}_T$ element that masks the message in our comparison as it is present in all constructions. Enc $= (a, b)$ implies that a scalar multiplications are required in $\mathbb{G}_1$ and b exponentiations in $\mathbb{G}_T$; 'Dec' is measured in terms of number of pairings; 'KGen' is determined by number of scalar multiplications in $\mathbb{G}_2$; 'Deleg', by number of scalar multiplications in $\mathbb{G}_2$. 'Assump' denotes the set of underlying complexity assumptions; Deg is a shorthand for security degradation. 'Prefix Dec' indicates whether or not the HIBE supports prefix decryption. 'Const #cpr' denotes constant number of elements in the ciphertext (or constant-size ciphertext).

Scheme	[24]	[49]	[50]	[114]	[122]	[54]	$\mathcal{JR}$-$\mathcal{HIBE}$
Pairing	Type-1	Type-1	Type-1	Composite	Type-1	Type-3	Type-3
Security	selective-id	adaptive-id	selective$^+$-id	adaptive-id	adaptive-id	adaptive-id	adaptive-id
Assump.	Decisional h-wBDHI	h-wDBDHI*	h-wDBDHI*	Subgroup Decision	DLin	d-Lin	SXDH
Deg.	1	$O((kq2^{N/k})^h)$	1	$O(q)$	$O(q)$	$O(q)$	$O(q)$
#pp	$(h+4,0)$	$(h+3+hk,0)$	$(2h+3,1)$	$(h+3,1)$	$(32h^2+16h+25,1)$	$(2d(d+1)(h+2),d)$	$(3h+9,1)$
#msk	1	1	1	1	5	$d+1$	2
#cpr	(2,0)	(2,0)	(3,0)	(2,0)	(13,0)	(2(d+1),0)	(3,1)
#key	$h-\ell+2$	$(k+1)(h-\ell)+2$	$2(h-\ell+1)$	$h-\ell+2$	$8h+5$	$(d+1)(h-\ell+2)$	$2(h-\ell)+5$
Enc	$(\ell+2,1)$	(2,1)	$(\ell+2,1)$	$(\ell+2,1)$	$32h+23$	$(d(d+1)(\ell+2),d)$	$(\ell+4,1)$
Dec	2	2	2	2	13	$2(d+1)$	3
KGen	$h+2$	$2(h-\ell+1)$	$2h-\ell+2$	$2h-\ell+4$	$16h(h+\ell)+10$	$d(d+1)(h+2)$	$2h+7$
Deleg.	$\ell+2$	$2(h-\ell)$	$2h-\ell+1$	$2h-\ell+6$	$16h(h+\ell+1)+10$	$d(d+1)(h+2)+d+1$	$2h+9$

Table 6.1: Comparison of non-anonymous CC-HIBE schemes based on pairings without random oracles.

Efficiency comparison of $\mathcal{JR}$-$\mathcal{HIBE}$ with [114]. In absolute terms, the number of group elements required for composite-order based schemes is less than that required in the new HIBE schemes. However, only counting group elements is not a proper comparison. One has to consider the actual size for representing a single group element at a desired security level.

For concreteness, let us consider a security level of 128 bits. For Type-3 pairings, using Table-2 of [43], elements of $\mathbb{G}_1$ and $\mathbb{G}_2$ can be represented using 257 and 513 bits respectively. In contrast, the order of $\mathbb{G}_1 = \mathbb{G}_2$ for composite-order pairings is a product of at least three primes. The basic security requirement is that this group order should be hard to factor. To attain 128-bit security level, the length of the bit representation of the group order should be about 3000 bits (or more). So, for schemes based on composite-order groups, the length of representations of elements of $\mathbb{G}_1$ (and $\mathbb{G}_2$) will be about 3000 bits. This is about 12 times (resp. 6 times) more than the length of bit representation of elements of $\mathbb{G}_1$ (resp. $\mathbb{G}_2$) using Type-3 pairings. The wide difference in the length of representations of group elements more than adequately compensates for the absolute number of group elements in composite-order HIBE schemes being lesser than that in the newly proposed HIBE scheme.

For example, ciphertexts in $\mathcal{JR}$-$\mathcal{AHIBE}$ (or $\mathcal{JR}$-$\mathcal{HIBE}$) consist of 3 elements of $\mathbb{G}_1$ which is about 770 bits whereas ciphertexts in the HIBE of [114] will be about 9000 bits (3 elements each having length about 3000 bits). Similar considerations apply to public parameters ($\mathcal{PP}$), master secret key ($\mathcal{MSK}$) and decryption keys. The larger length of the parameters also lead to a significant slow down in the basic operations of scalar multiplication and pairing computation leading to much slower algorithms for encryption, decryption, key generation and key delegation.

Comparing $\mathcal{JR}$-$\mathcal{HIBE}$ with [122] and [54]. The schemes in [122, 54] both achieve similar security guarantees as $\mathcal{JR}$-$\mathcal{HIBE}$. The construction of [122] the number of elements in the ciphertext and the number of pairings required for decryption is 13 as opposed to just 3 in $\mathcal{JR}$-$\mathcal{HIBE}$. The scheme in [54] achieves similar parameters when $d = 1$ (d-Lin is XDH when $d = 1$) but is still less efficient compared to $\mathcal{JR}$-$\mathcal{HIBE}$ in terms of ciphertext size and decryption time. Ciphertext in [54] will consist of 4 $\mathbb{G}_1$-elements whereas $\mathcal{JR}$-$\mathcal{HIBE}$ contains 3 $\mathbb{G}_1$-elements along with an element of $\mathbb{Z}_p$. If an element of $\mathbb{G}_1$ is represented using two elements of $\mathbb{Z}_p$, then $\mathcal{JR}$-$\mathcal{HIBE}$ ciphertexts consist of 7 $\mathbb{Z}_p$ elements as opposed to 8 in [54]. Certainly, $\mathcal{JR}$-$\mathcal{HIBE}$ has shorter ciphertexts.

From Table 6.1 and the previous discussion, the only non-anonymous HIBE scheme which is comparable in efficiency and security to $\mathcal{JR}$-$\mathcal{HIBE}$ is the Chen-Wee scheme described in [54] for $d = 1$ whence d-Lin becomes DDH. $\mathcal{JR}$-$\mathcal{HIBE}$ has shorter ciphertexts and faster encryption and decryption algorithms, while the Chen-Wee scheme has shorter decryption keys and faster key generation and delegation algorithms. For an encryption scheme, encryption and decryption will be used more often than key generation and delegation, so, the advantage of $\mathcal{JR}$-$\mathcal{HIBE}$ over the Chen-Wee scheme outweighs the disadvantages. When $d > 1$, the Chen-Wee scheme is based on progressively weaker assumptions than the SXDH assumption and the resulting schemes also become progressively more inefficient.

$\mathcal{LW}$-$\mathcal{AHIBE}$ in comparison with previous HIBE schemes. Observe that $\mathcal{LW}$-$\mathcal{AHIBE}$ was the one among the first two CC-HIBE schemes based on prime order pairings achieving adaptive security from static assumptions. The only comparable scheme is that of De Caro et al. [63] which

Scheme	[36]	[144]	[63]	[129]	[108],*LW-AHIBE*	[124]	*JR-AHIBE*
Pairing	Type-3	Composite	Composite	Type-1	Type-3	Type-1	Type-3
Security	selective-id	selective-id	adaptive-id	selective-id	adaptive-id	adaptive-id	adaptive-id
Assump.	DLin,DBDH	ℓ-wBDH*, ℓ-cDH	Subgroup Decision	h-BDHE Aug. h-DLin	LW1,LW2,DBDH [108]:3-DH,XDH [137]:A1	DLin	SXDH
Deg.	$O(1)$	$O(1)$	$O(q)$	$O(1)$	$O(q)$	$O(hq)$	$O(q)$
Prefix Dec.	No	No	No	No	No	Yes	No
Const #cpr	No	Yes	Yes	Yes	Yes	No	Yes
#pp	$(2(h^2+3h+2),1)$	$(h+6,1)$	$(h+4,1)$	$(h+6,1)$	$(3h+6,1)$	$(4(9h+4),1)$	$(h+4,1)$
#msk	h^2+5h+7	$h+4$	2	4	$h+6$	$18h+10$	$2h+6$
#cpr	$(2h+5,0)$	(3,0)	(2,0)	(4,0)	(6,0)	$(9\ell+5,0)$	(3,1)
#key	$(h+3)(3h-\ell+5)$	$3(h-\ell+3)$	$2(h-\ell+2)$	$3(h-\ell+4)$	$6(h-\ell+2)$	$(4h-2\ell+1)(9\ell+5)+36(h-\ell)$	$4(h-\ell)+10$
Enc	$(2(\ell+3)(h+2)+1,1)$	$(\ell+6,1)$	$(\ell+4,1)$	$(\ell+5,1)$	$(3(\ell+2),1)$	$27\ell+15$	$(\ell+4,1)$
Dec	$2h+3$	4	2	4	6	$9\ell+5$	3
KGen	$h^3+h^2(5-\ell)+$ $h(7-3\ell)-2\ell+2$	$3h-2\ell+2$	$4(h+2-3\ell)$	$(h+2(h-\ell+8))$	$6h-5\ell+12$	$(2h+3)(27\ell+10)$	$2(2h-2\ell+5)$
Deleg.	$5(h+2)(h+3)+1$	$6(h-\ell)+21$	$4(h-\ell)+11$	$(4(h-\ell)+25)$	$2(h-\ell+3)$	$(9\ell+5)(6h\ell+14h-2\ell^2-8\ell+5)$	$4(h-\ell+5)$

Table 6.2: Comparison of anonymous HIBE schemes based on pairings without random oracles.

is based on composite order pairings. The comments regarding the efficiency benefits of using prime order groups over composite order groups mentioned above also apply here. An independent work by Park and Lee [129] also proposed a HIBE construction identical to *LW-AHIBE*. In contrast to our proof, they prove security based on SXDH and asymmetric 3-party Diffie-Hellman assumptions in addition to LW1, LW2 and DBDH.

***JR-AHIBE* and other anonymous HIBE schemes.** It is clear from Table 6.2 that all anonymous HIBE schemes possess either constant-size ciphertexts or the prefix decryption property and not both. The Boyen-Waters HIBE [36] has none of the two properties. The Okamoto-Takashima scheme [124] supports prefix decryption and at the same time achieves anonymity but at the cost of non-constant size of the ciphertext (the size is linear in the depth of the identity). In addition, ciphertexts in their scheme reveal the length of the recipient identity unlike the Boyen-Waters HIBE. *JR-AHIBE*, on the other hand, is anonymous and has short ciphertexts but lacks prefix decryption. All other efficiency parameters are better in case of *JR-AHIBE*.

We conclude that among anonymous HIBE schemes, *JR-AHIBE* is the most efficient scheme with all the standard provable properties. We emphasise that the efficiency and provable security properties achieved for *JR-AHIBE* have not been simultaneously achieved earlier, either for composite-order pairings, or, for prime-order pairings. For use in practice, one may choose the Okamoto-Takashima scheme or *JR-AHIBE* according to whether the application requires prefix decryption or not.

Subsequent work by Blazy et al. A recent work [21] presents HIBE schemes (both with and without anonymity) generically constructed via a transformation from message authentication codes (MAC). Security is based on the d-Lin assumptions. Consider the case $d = 1$. For the schemes with public parameters comparable to our schemes ciphertexts are larger than that of *JR-AHIBE* and *JR-HIBE*. They also present a non-anonymous scheme with a tighter reduction and ciphertexts shorter than *JR-HIBE*-ciphertexts. But the public parameters are $O(hn)$ where n denotes the length (in bits) of each identity.

Chapter 7

Identity-Based Broadcast Encryption

In this chapter, we present the first efficient IBBE constructions that achieve security against adaptive-identity attacks under standard hardness assumptions. All previously known schemes either achieved security against selective-identity attacks, or used non-standard assumptions, or could be obtained by specialising inner product encryption making them quite inefficient. Further, our constructions are based on Type-3 pairings, whereas previous works on IBBE used Type-1 pairings.

Our Contributions. A simple way to encrypt a single message to a set of identities is to use an IBE scheme to encrypt it separately to each of the identities. Such a strategy, however, does not allow any savings in the header size. An IBE encryption results in a ciphertext which consists of several elements of $\mathbb{G}_1$. To obtain a non-trivial IBBE scheme, it is of interest to try and share some of the group elements in the ciphertext across all the encryptions. This will lead to a reduction in the size of the ciphertext over the trivial scheme of separate encryption to each identity.

Currently, the most efficient IBE scheme that is known is due to Jutla and Roy [103]. Actually a simple variant of their original proposal, described as JR-IBE-D in Chapter 6, is the IBE of choice. In this work, we investigate the possibility of converting this IBE scheme into an IBBE scheme. The intuitive idea is to share the randomiser across all the identities. Doing this directly, however, does not admit a security proof. To get around the problem, we need to put a bound on the size of the set of identities to which a single message can be simultaneously encrypted and then let the size of the public parameters be determined by this bound. The group elements in the public parameters allow the computation of polynomial hash of each of the identities. These hashes vary with the identities whereas the group elements which do not depend on the identity remain the same for all the identities. It is due to this feature that we are able to get a substantial practical reduction in the size of the ciphertext. The resulting scheme, denoted $\mathcal{IBBE}_1$, can be proved to be secure against adaptive-identity attacks using the dual system proof technique introduced by Waters [158]. The underlying hardness assumptions consist of the standard DDH assumptions in the groups $\mathbb{G}_1$ and $\mathbb{G}_2$ (DDH1 and DDH2 respectively).

A ciphertext in $\mathcal{IBBE}_1$ contains ℓ $\mathbb{Z}_p$-elements (called tags) where ℓ is the number of identities to which encryption is to be done. Our second scheme, $\mathcal{IBBE}_2$, is a modification of $\mathcal{IBBE}_1$ which provides a method whereby the number of tags in the ciphertext goes down and hence results in shorter ciphertexts. The security of this scheme can be reduced from the security of $\mathcal{IBBE}_1$ using a hybrid argument. We describe a method whereby the tags can be generated using a hash function resulting in an even further reduction in the size of the ciphertext. The reduction is more significant in the case of $\mathcal{IBBE}_1$ than in the case of $\mathcal{IBBE}_2$. The trade-off for doing this is that the hash function needs to be modelled as a random oracle for the security proof. User storage in both $\mathcal{IBBE}_1$ and $\mathcal{IBBE}_2$ consists of a constant number of group elements of $\mathbb{G}_2$.

Naor, Naor and Lotspiech [117] had provided a combinatorial framework called the complete subtree (CS) scheme for symmetric key BE. Dodis and Fazio [69] had shown how to combine an IBE scheme with the CS scheme to obtain a PKBE scheme. (Refer to Chapter 4 for a detailed discussion). We build on this framework and show that combining an IBBE scheme with the CS scheme leads to a PKBE scheme with even better parameters. Concretely, we discuss the issue of combining the CS scheme with $\mathcal{IBBE}_1$.

Note: IBBE can be viewed as a special case of inner product encryption [112, 122, 125]. Most adaptively secure inner product encryption schemes are constructed using dual pairing vector spaces and hence lead to very inefficient schemes. This is because ciphertexts and keys usually contain vectors of dimension at least 4 (over $\mathbb{G}_1$ or $\mathbb{G}_2$) resulting in too much ciphertext overhead in the broadcast setting. It is due to this reason, we do not discuss IBBE schemes that can be obtained by specialising inner product encryption.

7.1 IBBE – A First Construction

All our constructions are described in the KEM-DEM framework (refer to Definition 2.1.5 for a definition of IBBE-KEM). For the sake of simplicity, we use the term IBBE in place of IBBE-KEM. We start by providing a brief overview of our first IBBE construction – $\mathcal{IBBE}_1$. The starting point is JR-IBE-D (described in Section 6.3.1 of Chapter 6) that achieves adaptive-identity security from the DDH assumptions in $\mathbb{G}_1$ and $\mathbb{G}_2$. Let N_1, N_2, N_T and N_p denote the sizes of representation of elements in $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ and $\mathbb{Z}_p$ respectively. A ciphertext in JR-IBE-D consists of the three elements C_1, C_2 and C_3 from $\mathbb{G}_1$; the element C_0 from $\mathbb{G}_T$; and the element tag from $\mathbb{Z}_p$. The size of one ciphertext is $N_T + 3N_1 + N_p$.

Now consider the setting of identity-based broadcast encryption. Suppose that $S = \{\mathsf{id}_1, \ldots, \mathsf{id}_\ell\} \subseteq \mathscr{I}$ is a set of identities corresponding to the intended recipients of a message. A natural way to extend the IBE scheme to the broadcast setting is as follows. The user keys will be the usual IBE decryption keys and the public parameters will also remain the same. Components C_1, C_2 would still remain the same since they are independent of the identity. The mask $(g_T)^s$ used to encrypt the message in C_0 will now play the role of the session key i.e., $K = (g_T)^s$. Introduce separate identity-hashes for each identity but randomised with the same scalar. In particular, C_3 is replaced by $C_{3,i} = s(U_1 + \mathsf{id}_i V_1 + \mathsf{tag}_i W_1)$, $i \in [1, \ell]$.

We would like to emphasise that having separate hashes for each identity requires the use of separate tags for the different hashes. Otherwise, one can get hold of sV_1 by just taking the difference between $C_{3,i}$ and $C_{3,j}$ for some $i \neq j$. With sV_1, an attacker can construct a header for $S' = S \cup \{\mathsf{id}\}$ for any id of its choice. This header when decapsulated using a secret key for id, results in the same session key that the header for S encapsulates. So, not having separate tags makes the scheme insecure.

For the scheme with separate tags as described above, the header size will be $(2+\ell)N_1 + \ell N_p$. This is better than performing separate IBE encryptions for each identity resulting in header size of $\ell(N_T + 3N_1 + N_p)$. However, the scheme as described does not seem to admit a security proof. Defining $C_{3,i}$ as above leads to problems during simulation within the dual system framework. To see why the above method fails, we take a look at the dual system proof of JR-IBE-D.

The structure of dual-system proof: An important step in a dual system proof is to show that a normal key is computationally indistinguishable from a semi-functional key. When the attacker requests a key for an identity id, a DDH2 instance is embedded in the key $\mathcal{SK}_{\mathsf{id}}$ in such a way that the power of the attacker in determining whether $\mathcal{SK}_{\mathsf{id}}$ is normal or semi-functional can be used to solve the particular instance. At the same time the simulator needs to create a valid semi-functional ciphertext for the challenge identity $\widehat{\mathsf{id}}$. One must also ensure any semi-functional ciphertext that the simulator creates for id cannot provide any extra advantage in solving the problem instance. All this is achieved (in JR-IBE-D) by embedding a degree-one polynomial $f(x) = Ax + B$ in both the $\widehat{\mathsf{tag}}$ in the ciphertext for $\widehat{\mathsf{id}}$ and the scalar π in the semi-functional components of $\mathcal{SK}_{\mathsf{id}}$. Moreover, A and B are programmed into the public parameters in such a way that they are information theoretically hidden from an attacker's viewpoint. Specifically, they are embedded in parameters V_1 and U_1 in the $\mathcal{PP}$.

First of all, a degree one polynomial in random variables A, B provides pairwise independence when evaluated at two different points (A, B are uniformly and independently distributed). This ensures correct distribution of $\pi = f(\mathsf{id})$ and $\mathsf{tag} = f(\widehat{\mathsf{id}})$. Secondly, the only way of creating a semi-functional ciphertext for an identity id' is by setting $\mathsf{tag}' = f(\mathsf{id}')$ implying that any attempt by the simulator to create a semi-functional ciphertext for id will set $\mathsf{tag} = \pi$. As a result, decryption is successful and the simulator gains no information about the semi-functionality of $\mathcal{SK}_{\mathsf{id}}$.

Independence issue for IBBE scheme: In the extension to the broadcast setting discussed above, we need to argue about the independence of $\widehat{\ell}$ tags $\mathsf{tag}_1, \ldots, \mathsf{tag}_{\widehat{\ell}}$ in the challenge header for $\widehat{S} = \{\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}}\}$, plus the scalar π in the secret key for some $\mathsf{id} \notin \widehat{S}$. Also we need to argue about the joint distribution of all the tags in a single step since they all share the same randomiser. A degree one polynomial does not provide sufficient amount of randomness to do so. This is exactly where the dual system argument fails.

To overcome this problem, we introduce the restriction that the maximum size of a privileged users' set should be at most m. Then we replace the JR-IBE-D identity hash by a degree-m polynomial hash in the identity. Such a polynomial provides $(m+1)$-wise independence. Since one needs to argue about the independence of at most m tags and one π, this hash will suffice for a dual system proof.

The coefficients of the polynomial are determined by the public parameters. So instead of U_1, V_1, $\mathcal{PP}$ will now contain elements $U_{1,j}$ for $j = 0, \ldots, m$. Define component $C_{3,i}$ as $C_{3,i} = s(\sum_{j=0}^{m}(\mathsf{id}_i)^j U_{1,j} + \mathsf{tag}_i W_1)$ for $\mathsf{id}_i \in S$. Also, as in JR-IBE-D, $U_{1,j}$'s and W_1 are created using linear combinations of certain scalars in the master secret i.e., $U_{1,j} = (e_j + b\Delta_j)P_1$ for $j = 0, \ldots, m$ and $W_1 = (c + b\Delta)P_1$. So the secret key for an identity id will now consist of the two sub-hashes $\sum_{j=0}^{m}(\mathsf{id})^j e_j$ and $\sum_{j=0}^{m}(\mathsf{id})^j \Delta_j$. These sub-hashes are combined using b in C_2 during decryption to cancel out the hash in $C_{3,i}$ if $\mathsf{id} = \mathsf{id}_i$.

The technique of using polynomials to hash identities has been used earlier by Chatterjee and Sarkar in [46] in the context of IBBE. However, they only obtain weaker security against selective-identity attacks.

7.1.1 Construction of $\mathcal{IBBE}_1$

We define our first IBBE construction

$$\mathcal{IBBE}_1 = (\mathcal{IBBE}_1.\mathsf{Setup}, \mathcal{IBBE}_1.\mathsf{Encap}, \mathcal{IBBE}_1.\mathsf{KeyGen}, \mathcal{IBBE}_1.\mathsf{Decap})$$

as follows.

$\mathcal{IBBE}_1.\mathsf{Setup}(\kappa, m)$: Generate a Type-3 pairing $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, F_1, F_2)$ based on the security parameter κ. Here, $\mathscr{I} = \mathbb{Z}_p$ and $\mathscr{K} = \mathbb{G}_T$. In practice, a hash function can be used to map $\mathbb{G}_T$ to the actual key space of the symmetric encryption scheme. Compute parameters as follows.

$P_1 \xleftarrow{\text{U}} \mathbb{G}_1^\times$, $P_2 \xleftarrow{\text{U}} \mathbb{G}_2^\times$
$\alpha_1, \alpha_2, c, \Delta, (e_j, \Delta_j)_{j=0}^m \xleftarrow{\text{U}} \mathbb{Z}_p$, $b \xleftarrow{\text{U}} \mathbb{Z}_p^\times$,

$U_{1,j} = (\Delta_j b + e_j)P_1$ for $j = 0, \ldots, m$, $W_1 = (\Delta b + c)P_1$,
$g_T = e(P_1, P_2)^{\alpha_1 b + \alpha_2}$,

$\mathcal{PP} : (P_1, bP_1, (U_{1,j})_{j=0}^m, W_1, g_T)$
$\mathcal{MSK} : (P_2, cP_2, \alpha_1, \alpha_2, \Delta, (e_j, \Delta_j)_{j=0}^m)$

$\mathcal{IBBE}_1.\mathsf{KeyGen}(\mathcal{MSK}, \mathsf{id})$: Choose $r \xleftarrow{\text{U}} \mathbb{Z}_p$ and compute the secret key $\mathcal{SK}_{\mathsf{id}} = (D_1, D_2, D_3, D_4, D_5)$ as follows.

$D_1 = rP_2$, $D_2 = rcP_2$, $D_3 = \left(\alpha_1 + r(\sum_{j=0}^m (\mathsf{id})^j e_j)\right) P_2$,
$D_4 = r\Delta P_2$, $D_5 = \left(\alpha_2 + r(\sum_{j=0}^m (\mathsf{id})^j \Delta_j)\right) P_2$,

$\mathcal{IBBE}_1.\mathsf{Encap}(\mathcal{PP}, S = \{\mathsf{id}_1, \ldots, \mathsf{id}_\ell\})$: If $\ell \leq m$, pick $s, (\mathsf{tag}_i)_{i=1}^\ell \xleftarrow{\text{U}} \mathbb{Z}_p$. Compute the session key as $K = g_T^s$. The header is given by $\mathsf{Hdr} = (C_1, C_2, (C_{3,i}, \mathsf{tag}_i)_{i=1}^\ell)$ where

$C_1 = sP_1$, $C_2 = sbP_1$,
$C_{3,i} = s(\sum_{j=0}^m (\mathsf{id}_i)^j U_{1,j} + \mathsf{tag}_i W_1)$ for $i = 1, \ldots, \ell$.

$\mathcal{IBBE}_1.\mathsf{Decap}(\mathcal{PP}, S, \mathsf{id}, \mathcal{SK}_{\mathsf{id}}, \mathsf{Hdr})$: Suppose that $S = \{\mathsf{id}_1, \ldots, \mathsf{id}_\ell\}$. If $\mathsf{id} \in S$, there is some index $i \in [1, \ell]$ such that $\mathsf{id} = \mathsf{id}_i$. The session key is derived as follows.

$$K = \frac{e(C_1, \mathsf{tag}_i D_2 + D_3) e(C_2, \mathsf{tag}_i D_4 + D_5)}{e(C_{3,i}, D_1)}.$$

Correctness: Let $S = \{\mathsf{id}_1, \ldots, \mathsf{id}_\ell\} \subseteq \mathscr{I}$ with $\ell \leq m$. Let $(\mathsf{Hdr}, K) \longleftarrow \mathcal{IBBE}_1.\mathsf{Encap}(\mathcal{PP}, S; s)$ where $\mathsf{Hdr} = (C_1, C_2, (C_{3,i}, \mathsf{tag}_i)_{i=1}^{\ell})$ and let $\mathcal{SK}_{\mathsf{id}_i} \xleftarrow{\mathrm{R}} \mathcal{IBBE}_1.\mathsf{KeyGen}(\mathcal{MSK}, \mathsf{id}_i; r)$ for some $\mathsf{id}_i \in S$.

$$
\begin{aligned}
&\frac{e(C_1, \mathsf{tag}_i D_2 + D_3) e(C_2, \mathsf{tag}_i D_4 + D_5)}{e(C_{3,i}, D_1)} \\
&= \frac{e(sP_1, \mathsf{tag}_i \cdot rcP_2 + (\alpha_1 + r(\sum_{j=0}^{m} (\mathsf{id}_i)^j e_j))P_2) \cdot e(sbP_1, \mathsf{tag}_i r\Delta P_2 + (\alpha_2 + r(\sum_{j=0}^{m} (\mathsf{id}_i)^j \Delta_j))P_2)}{e(s(\sum_{j=0}^{m} (\mathsf{id}_i)^j U_{1,j} + \mathsf{tag}_i W_1), rP_2)} \\
&= \frac{e(sP_1, \alpha_1 P_2) \cdot e(sP_1, P_2)^{\mathsf{tag}_i rc + r(\sum_{j=0}^{m} (\mathsf{id}_i)^j e_j)} \cdot e(sP_1, b\alpha_2 P_2) e(sP_1, P_2)^{\mathsf{tag}_i \cdot r\Delta b + r(\sum_{j=0}^{m} (\mathsf{id}_i)^j \Delta_j b)}}{e((\sum_{j=0}^{m} (\mathsf{id}_i)^j \Delta_j b + \mathsf{tag}_i \Delta b + \sum_{j=0}^{m} (\mathsf{id}_i)^j e_j + \mathsf{tag}_i c) P_1, P_2)^{rs}} \\
&= \frac{e(P_1, (\alpha_1 + b\alpha_2) P_2)^s \cdot e((\sum_{j=0}^{m} (\mathsf{id}_i)^j \Delta_j b + \mathsf{tag}_i \Delta b + \sum_{j=0}^{m} (\mathsf{id}_i)^j e_j + \mathsf{tag}_i c) P_1, P_2)^{rs}}{e((\sum_{j=0}^{m} (\mathsf{id}_i)^j \Delta_j b + \mathsf{tag}_i \Delta b + \sum_{j=0}^{m} (\mathsf{id}_i)^j e_j + \mathsf{tag}_i c) P_1, P_2)^{rs}} \\
&= g_T^s.
\end{aligned}
$$

Header size and user storage: The header consists of $(2 + \ell)$ elements of $\mathbb{G}_1$, ℓ elements of $\mathbb{Z}_p$ and one element of $\mathbb{G}_T$. Using the previous notation, the size of the header is $(2 + \ell)N_1 + \ell N_p + N_T$. The number of keys to be stored by each user consists of 5 elements of $\mathbb{G}_2$.

Use of random oracles. Let $H : \{0,1\}^\kappa \times [1, m] \to \mathbb{Z}_p$ be a hash function that takes a seed (say z) of length κ, an index $i \in [1, m]$ as input and produces a value in $\mathbb{Z}_p$ as output. If H is modeled as a random oracle, then for distinct inputs, the outputs will be independent and uniformly distributed in $\mathbb{Z}_p$. Such an H can be used to reduce the header size in the following manner. In the $\mathcal{IBBE}_1$ header, the tags are replaced by a uniform random κ-bit quantity z. The actual tags are generated by evaluating H on inputs (z, i) for each $i \in [1, \ell]$ where $|S| = \ell$. The size of the resulting header will be $N_T + (2 + \ell)N_1 + \kappa$. In practical terms, the efficiency gain over $\mathcal{IBBE}_1$ is quite significant. The modified scheme, which we call $\mathcal{IBBE}_1^{\mathrm{RO}}$ (RO denotes random oracle), can be shown to be secure via a reduction from an adversary breaking its security to an adversary against scheme $\mathcal{IBBE}_1$. Essentially, the tags that the adversary against $\mathcal{IBBE}_1$ obtains as part of the challenge header are returned as answers to the random oracle queries that the adversary against $\mathcal{IBBE}_1^{\mathrm{RO}}$ makes. Note that the use of random oracles is "minimal". It may be possible to use ROs more effectively to further reduce the header size.

Getting rid of tags? It would be nice to be able to completely get rid of the tags. These tags play a crucial role in the dual system proof. Lewko and Waters [114] proposed a different type of dual system encryption where the role of the tags is shifted to some scalars in the semi-functional components (similar to the scalar π in an $\mathcal{IBBE}_1$ secret key). However, one must also ensure that a semi-functional component can be decrypted by a normal key which in turn requires that these scalars in the semi-functional components cancel out during decryption. This can be done with multiple copies of the identity hash (as in [114]) in the ciphertext. In the context of broadcast encryption, having multiple copies of the identity hash in the ciphertext increases the header size. So, it does not seem likely that the technique of [114] will help reduce the header size any further.

Restriction on the size of the identity set: In the encapsulation algorithm we have assumed that the number of identities ℓ to which the message is to be encrypted is at most m, the parameter of the IBBE scheme. If it turns out that $\ell > m$, then the set of identities will be divided into $\lceil \ell/m \rceil$ groups and the encapsulation algorithm will be applied separately to each group. The resulting header size will be $\lceil \ell/m \rceil ((m+2)N_1 + mN_p + N_T)$. Since this is quite routine, we will simply analyse the scheme under the assumption that $\ell \le m$.

7.1.2 Security of $\mathcal{IBBE}_1$

The scheme $\mathcal{IBBE}_1$ is proved secure in the sense of $\mathsf{IND\text{-}BID\text{-}CPA}$ (Section 2.2.3). via the dual system technique. The following theorem formally states the security guarantee we prove for the scheme $\mathcal{IBBE}_1$.

Theorem 7.1.1. *If* $(\varepsilon_{\text{DDH1}}, t_1)$-DDH1 *and* $(\varepsilon_{\text{DDH2}}, t_2)$-DDH2 *assumptions hold in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$ *respectively, then* $\mathcal{IBBE}_1$ *is* (ε, t, q)-$\mathsf{IND\text{-}BID\text{-}CPA}$*-secure where* $\varepsilon \le \varepsilon_{\text{DDH1}} + 2q \cdot \varepsilon_{\text{DDH2}} + (q/p)$, $t_1 = t + O(m^2\rho)$ *and* $t_2 = t + O(m^2\rho)$. ρ *is the maximum time required for one scalar multiplication in* $\mathbb{G}_1$ *or* $\mathbb{G}_2$.

Proof. We start by appropriately defining semi-functional headers and user keys for $\mathcal{IBBE}_1$. Let $\mathcal{IBBE}_1.\mathsf{SFEncap}$ and $\mathcal{IBBE}_1.\mathsf{SFKeyGen}$ be algorithms that generate semi-functional headers and user keys (respectively) described as follows.

$\mathcal{IBBE}_1.\mathsf{SFEncap}(\mathcal{PP}, \mathcal{MSK}, S, (\mathsf{Hdr}, K))$: Takes as input a header-key pair created by $\mathcal{IBBE}_1.\mathsf{Encap}$ algorithm on a set S and modifies it to obtain semi-functional header and session key. Let $S = \{\mathsf{id}_1, \ldots, \mathsf{id}_\ell\}$ and $\mathsf{Hdr} = (C_1, C_2, (C_{3,i}, \mathsf{tag}_i)_{i=1}^{\ell})$. Pick $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$ and modify K and the components of Hdr as follows.

$$K \leftarrow K \cdot e(P_1, P_2)^{\alpha_1 \mu}, \quad C_1 \leftarrow C_1 + \mu P_1, \quad C_2 \leftarrow C_2,$$

$$C_{3,i} \leftarrow C_{3,i} + \mu(\sum_{j=0}^{m} (\mathsf{id}_i)^j e_j + \mathsf{tag}_i \cdot c)P_1 \text{ for } i = 1, \ldots, \ell.$$

Return the modified session key K along with the header $\mathsf{Hdr} = (C_1, C_2, (C_{3,i}, \mathsf{tag}_i)_{i=1}^{\ell})$.

$\mathcal{IBBE}_1.\mathsf{SFKeyGen}(\mathcal{MSK}, \mathcal{SK}_{\mathsf{id}})$: This algorithm takes in a normal secret key $\mathcal{SK}_{\mathsf{id}} = (D_1, \ldots, D_5)$ for identity id and generates a semi-functional key as follows.

$$\gamma, \pi \xleftarrow{\text{U}} \mathbb{Z}_p,$$

$$D_1 \leftarrow D_1, \quad D_2 \leftarrow D_2 + \gamma P_2, \quad D_3 \leftarrow D_3 + \gamma\pi P_2,$$
$$D_4 \leftarrow D_4 - \left(\frac{\gamma}{b}\right) P_2, \quad D_5 \leftarrow D_5 - \left(\frac{\gamma\pi}{b}\right) P_2.$$

The resulting key $\mathcal{SK}_{\mathsf{id}} = (D_1, \ldots, D_5)$ is returned.

We need to show that all the semi-functionality properties are satisfied. Let $(\mathsf{Hdr} = (C_1, C_2, (C_{3,i}, \mathsf{tag}_i)_{i=1}^{\ell}), K)$ be a header-key pair for the set $S = \{\mathsf{id}_1, \ldots, \mathsf{id}_\ell\}$ and let $\mathcal{SK}_{\mathsf{id}_i}$ be a user key for an identity $\mathsf{id}_i \in S$. Consider the following cases.

$\mathcal{SK}_{id_i}$ is semi-functional and (Hdr, K) is normal: Let $\mathcal{SK}_{\mathsf{id}_i} \longleftarrow \mathcal{IBBE}_1.\mathsf{SFKeyGen}(\mathcal{MSK}, \mathcal{SK}'_{\mathsf{id}_i}; \gamma, \pi)$ where $\mathcal{SK}'_{id_i}$ is a normally generated key for id_i.

The requirement is that when Hdr is decapsulated with $\mathcal{SK}_{\mathsf{id}}$, the result is K. The following calculation shows that this requirement is satisfied.

$$\begin{aligned}
&\frac{e(C_1, \mathsf{tag}_i D_2 + D_3) e(C_2, \mathsf{tag}_i D_4 + D_5)}{e(C_{3,i}, D_1)} \\
&= K \cdot e(sP_1, \mathsf{tag}_i \gamma P_2 + \gamma\pi P_2) e(sbP_1, -\mathsf{tag}_i(\gamma/b)P_2 - (\gamma\pi/b)P_2) \\
&= K \cdot e(sP_1, \mathsf{tag}_i \gamma P_2 + \gamma\pi P_2) e(sP_1, -\mathsf{tag}_i \gamma P_2 - \gamma\pi P_2) \\
&= K.
\end{aligned}$$

The second step follows from the correctness condition i.e., a normal header when decapsulated with a normal user key gives the corresponding normal session key.

$\mathcal{SK}_{\mathsf{id}_i}$ is normal and (Hdr, K) is semi-functional: Let (Hdr', K') be a normally generated header-key pair and let $(\mathsf{Hdr}, K) \longleftarrow \mathcal{IBBE}_1.\mathsf{SFEncap}(\mathcal{PP}, \mathcal{MSK}, S, (\mathsf{Hdr}', K'); \mu)$. We have

$$\begin{aligned}
&\frac{e(C_1, \mathsf{tag}_i D_2 + D_3) e(C_2, \mathsf{tag}_i D_4 + D_5)}{e(C_{3,i}, D_1)} \\
&= K \cdot \frac{e(\mu P_1, \mathsf{tag}_i D_2 + D_3)}{e(\mu(\sum_{j=0}^{m} (\mathsf{id}_i)^j e_j + \mathsf{tag}_i \cdot c)P_1, D_1)} \\
&= K \cdot \frac{e(\mu P_1, r(\sum_{j=0}^{m} (\mathsf{id}_i)^j e_j + \mathsf{tag}_i \cdot c)P_2)}{e(\mu(\sum_{j=0}^{m} (\mathsf{id}_i)^j e_j + \mathsf{tag}_i \cdot c)P_1, rP_2)} \\
&= K,
\end{aligned}$$

as required.

Both $\mathcal{SK}_{\mathsf{id}_i}$ and (Hdr, K) are semi-functional: Let (Hdr', K') be a normally generated header-key pair and $\mathcal{SK}'_{\mathsf{id}_i}$ a normal key for id_i. Also let $(\mathsf{Hdr}, K) \longleftarrow \mathcal{IBBE}_1.\mathsf{SFEncap}(\mathcal{PP}, \mathcal{MSK}, S, (\mathsf{Hdr}', K'); \mu)$ and $\mathcal{SK}_{\mathsf{id}_i} \longleftarrow \mathcal{IBBE}_1.\mathsf{SFKeyGen}(\mathcal{MSK}, \mathcal{SK}'_{\mathsf{id}_i}; \gamma, \pi)$. In this case, the key obtained by running the $\mathcal{IBBE}_1.\mathsf{Decap}$ algorithm is masked by a factor of $e(P_1, P_2)^{\mu\gamma(\mathsf{tag}_i + \pi)}$ as shown below.

$$\begin{aligned}
&\frac{e(C_1, \mathsf{tag}_i D_2 + D_3) e(C_2, \mathsf{tag}_i D_4 + D_5)}{e(C_{3,i}, D_1)} \\
&= K \cdot e(\mu P_1, \mathsf{tag}_i \gamma P_2 + \gamma\pi P_2) \\
&= K \cdot e(P_1, P_2)^{\mu\gamma(\mathsf{tag}_i + \pi)}.
\end{aligned}$$

In the second step we retain only pairings between semi-functional components since all other pairings involving semi-functional components get cancelled.

Note that the masking factor vanishes when $\mathsf{tag}_i = -\pi$. Then $\mathcal{SK}_{id_i}$ and Hdr are called *nominally semi-functional* for id_i.

Now, given that semi-functional algorithms are defined, consider a sequence of games G_{real}, G_0, $(\mathsf{G}_k)_{k=1}^{q}$, G_{final} between an adversary $\mathscr{A}$ and a challenger with the games defined as follows.

- G_{real}: the actual IBBE security game $\mathsf{ind\text{-}ibbe\text{-}cpa}$ (described in Section 2.2.3).

- G_k, $0 \le k \le q$: challenge header is semi-functional; K_0 is semi-functional; first k user keys are semi-functional.
- G_{final}: challenge header is semi-functional; K_0 is random.

Let $X_\Box$ denote the event that $\mathscr{A}$ wins in $\mathsf{G}_\Box$. In Lemmas 7.1.1, 7.1.2 and 7.1.3, we show that

- $|\Pr[X_{real}] - \Pr[X_0]| \le \varepsilon_{\text{DDH1}}$,
- $|\Pr[X_{k-1}] - \Pr[X_k]| \le \varepsilon_{\text{DDH2}}$ for $k = 1, \ldots, q$,
- $|\Pr[X_q] - \Pr[X_{final}]| \le q/p$.

Clearly, the bit β is statistically hidden from the attacker in G_{final}, which means that $\Pr[X_{final}] = 1/2$. Hence, the advantage of $\mathscr{A}$ in breaking the security of $\mathcal{IBBE}_1$ is given by

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ind\text{-}ibbe\text{-}cpa}}_{\mathcal{IBBE}_1}(\mathscr{A}) &= |\Pr[X_{real}] - \frac{1}{2}| \\
&= |\Pr[X_{real}] - \Pr[X_{final}]| \\
&\le |\Pr[X_{real}] - \Pr[X_0]| + \sum_{k=1}^{q}(|\Pr[X_{k-1}] - \Pr[X_k]|) \\
&\qquad + |\Pr[X_q] - \Pr[X_{final}]| \\
&\le \varepsilon_{\text{DDH1}} + 2q\varepsilon_{\text{DDH2}} + \frac{q}{p}.
\end{aligned}$$

□

In the sequel, $\mathscr{B}_1$ (resp. $\mathscr{B}_2$) is a DDH1-solver (resp. DDH2-solver). We argue that $\mathscr{B}_1$, using the adversary's ability to distinguish between G_{real} and G_0, can solve DDH1. Similarly, $\mathscr{A}$'s power to distinguish between G_{k-1} and G_k for $k \in [1, q]$, can be leveraged to build a DDH2-solver $\mathscr{B}_2$.

Lemma 7.1.1. $|\Pr[X_{real}] - \Pr[X_0]| \le \varepsilon_{\text{DDH1}}$.

Proof. Let $(\mathcal{G}, P_1, bP_1, sbP_1, P_2, (s+\mu)P_1)$ be the instance of DDH1 that $\mathscr{B}_1$ has to solve i.e., decide whether $\mu = 0$ or $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$. The phases of the game are simulated by $\mathscr{B}_1$ as described below.

Setup: Choose $\alpha_1, \alpha_2, c, \Delta, (e_j, \Delta_j)_{j=0}^m \xleftarrow{\text{U}} \mathbb{Z}_p$ and set parameters as:

$U_{1,j} = \Delta_j(bP_1) + e_jP_1$ for $j = 0, \ldots, m$, $W_1 = \Delta(bP_1) + cP_1$,
$g_T = e(P_1, P_2)^{\alpha_1} e(bP_1, P_2)^{\alpha_2}$
$\mathcal{PP} : (P_1, bP_1, (U_{1,j})_{j=0}^m, W_1, g_T)$

All the secret scalars present in the $\mathcal{MSK}$ are known. $\mathscr{B}_1$ can thus create normal keys. However, $\mathscr{B}_1$'s lack of knowledge of the scalar b or its encoding in $\mathbb{G}_2$ prevents it from creating semi-functional keys.

Key Extraction Phases 1 & 2: $\mathscr{B}_1$ answers all of $\mathscr{A}$'s queries with normal keys generated by the $\mathcal{IBBE}_1$.KeyGen algorithm.

Challenge: $\mathscr{A}$ sends a challenge set $\widehat{S} = \{\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}}\}$. $\mathscr{B}$ sets $(\widehat{\mathsf{Hdr}}, K_0)$ as follows.

For $i = 1, \ldots, \widehat{\ell}$, choose $\widehat{\mathsf{tag}}_i \xleftarrow{\text{U}} \mathbb{Z}_p$,
$K_0 = e(sbP_1, P_2)^{\alpha_2} e((s+\mu)P_1, P_2)^{\alpha_1} = g_T^s e(P_1, P_2)^{\alpha_1 \mu}$,
$\widehat{C}_1 = (s+\mu)P_1 = sP_1 + \mu P_1$,
$\widehat{C}_2 = sbP_1$,

For $i = 1, \ldots, \widehat{\ell}$,
$\widehat{C}_{3,i} = (\sum_{j=0}^m \Delta_j (\widehat{\mathsf{id}}_i)^j + \widehat{\mathsf{tag}}_i \cdot \Delta)(sbP_1) + (\sum_{j=0}^m e_j (\widehat{\mathsf{id}}_i)^j + \widehat{\mathsf{tag}}_i \cdot c)(s+\mu)P_1$
$= (\sum_{j=0}^m (\widehat{\mathsf{id}}_i)^j (\Delta_j b + e_j) + \widehat{\mathsf{tag}}_i (\Delta b + c))(sP_1) + (\sum_{j=0}^m e_j (\widehat{\mathsf{id}}_i)^j + \widehat{\mathsf{tag}}_i \cdot c)(\mu P_1)$
$= s(\sum_{j=0}^m (\widehat{\mathsf{id}}_i)^j U_{1,j} + \widehat{\mathsf{tag}}_i W_1) + \mu(\sum_{j=0}^m e_j (\widehat{\mathsf{id}}_i)^j + \widehat{\mathsf{tag}}_i \cdot c)P_1$.

$\mathscr{B}_1$ sets $\widehat{\mathsf{Hdr}} = (\widehat{C}_1, \widehat{C}_2, (\widehat{C}_{3,i}, \widehat{\mathsf{tag}}_i)_{i=1}^{\widehat{\ell}})$. It then samples $K_1 \xleftarrow{\text{U}} \mathbb{G}_T$, $\beta \xleftarrow{\text{U}} \{0,1\}$ and returns the pair $(\widehat{\mathsf{Hdr}}, K_\beta)$ to $\mathscr{A}$. Observe that $(\widehat{\mathsf{Hdr}}, K_0)$ is normal if $\mu = 0$ and semi-functional when $\mu \xleftarrow{\text{U}} \mathbb{Z}_p$.

Guess: $\mathscr{A}$ outputs its guess β' and halts.

$\mathscr{B}$ returns 1 if $\mathscr{A}$'s guess is correct i.e., $\beta = \beta'$; otherwise $\mathscr{B}_1$ returns 0. The advantage of $\mathscr{B}_1$ in solving the DDH1 instance is given by

$$\begin{aligned}\mathsf{Adv}_{\mathcal{G}}^{\text{DDH1}}(\mathscr{B}_1) &= |\Pr[\mathscr{B}_1 \text{ returns } 1 | \mu = 0] - \Pr[\mathscr{B}_1 \text{ returns } 1 | \mu \xleftarrow{\text{U}} \mathbb{Z}_p]| \\ &= |\Pr[\beta = \beta' | \mu = 0] - \Pr[\beta = \beta' | \mu \xleftarrow{\text{U}} \mathbb{Z}_p]| \\ &= |\Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{real}] - \Pr[\mathscr{A} \text{ wins in } \mathsf{G}_0]| \\ &= |\Pr[X_{real}] - \Pr[X_0]|.\end{aligned}$$

Since $\mathsf{Adv}_{\mathcal{G}}^{\text{DDH1}}(\mathscr{B}_1) \le \varepsilon_{\text{DDH1}}$, we have $|\Pr[X_{real}] - \Pr[X_0]| \le \varepsilon_{\text{DDH1}}$. □

Lemma 7.1.2. *$|\Pr[X_{k-1}] - \Pr[X_k]| \le \varepsilon_{\text{DDH2}}$ for $k \in [1, q]$.*

Proof. $\mathscr{B}_2$ is given an instance $(\mathcal{G}, P_1, P_2, rP_2, cP_2, (rc+\gamma)P_2)$ of DDH2 and has to decide whether $\gamma = 0$ or $\gamma \xleftarrow{\text{U}} \mathbb{Z}_p$. It simulates the game as described below.

Setup: Pick scalars $\alpha_1, \alpha_2', c, \Delta', (e_{j,1}, e_{j,2}, \Delta_j')_{j=0}^m \xleftarrow{\text{U}} \mathbb{Z}_p$ and $b \xleftarrow{\text{U}} \mathbb{Z}_p^\times$ and (implicitly) set

$$\alpha_2 = \frac{\alpha_2' - \alpha_1}{b}, \quad \Delta = \frac{\Delta' - c}{b},$$

$$e_j = e_{j,1} + c e_{j,2}, \quad \Delta_j = \frac{\Delta_j' - e_j}{b} \quad \text{for } j = 0, \ldots, m.$$

Parameters are generated as follows.
$U_{1,j} = \Delta_j' P_1$ for $j = 0, \ldots, m$, $W_1 = -\Delta' P_1$,
$g_T = e(P_1, P_2)^{\alpha_2'}$
$\mathcal{PP} : (P_1, bP_1, (U_{1,j})_{j=0}^m, W_1, g_T)$

The elements Δ, Δ_j, e_j that are part of the $\mathcal{MSK}$ are not available to $\mathscr{B}_2$. Even without these, $\mathscr{B}_2$ can generate keys as explained in the simulation of the key generation phases.

Key Extraction Phases: $\mathscr{A}$ queries on identities $\mathsf{id}_1, \mathsf{id}_2, \ldots, \mathsf{id}_q$. $\mathscr{B}$ responds to the ν-th query ($\nu \in [1, q]$) considering three cases.

Case 1: $\nu > k$

$\mathscr{B}_2$ returns a normal key, $\mathcal{SK}_{\mathsf{id}_\nu} = (D_1, \ldots, D_5)$. The master secret is not completely available to $\mathscr{B}_2$ and hence the $\mathcal{IBBE}_1$.KeyGen needs a modification. The components of the key are computed as shown below.

$$r_\nu \xleftarrow{\text{U}} \mathbb{Z}_p,$$

$$D_1 = r_\nu P_2,\ D_2 = r_\nu (cP_2),$$

$$D_3 = \left(\alpha_1 + r_\nu \left(\sum_{j=0}^{m} (\mathsf{id}_\nu)^j e_{j,1}\right)\right) P_2 + r_\nu \left(\sum_{j=0}^{m} (\mathsf{id}_\nu)^j e_{j,2}\right)(cP_2) = \left(\alpha_1 + r_\nu \left(\sum_{j=0}^{m} (\mathsf{id}_\nu)^j e_j\right)\right) P_2,$$

$$D_4 = b^{-1} r_\nu (\Delta' P_2 - cP_2) = r_\nu \left(\frac{\Delta' - c}{b}\right) P_2 = r_\nu \Delta P_2,$$

$$\begin{aligned}
D_5 &= b^{-1}\left(\alpha_2' - \alpha_1 + r_\nu \left(\sum_{j=0}^{m} (\mathsf{id}_\nu)^j (\Delta_j' - e_{j,1})\right)\right) P_2 - b^{-1} r_\nu \left(\sum_{j=0}^{m} (\mathsf{id}_\nu)^j e_{j,2}\right)(cP_2) \\
&= b^{-1}\left(\alpha_2' - \alpha_1 + r_\nu \left(\sum_{j=0}^{m} (\mathsf{id}_\nu)^j (\Delta_j' - e_{j,1} - ce_{j,2})\right)\right) P_2 \\
&= \left(\frac{\alpha_2' - \alpha_1}{b} + r_\nu \left(\sum_{j=0}^{m} (\mathsf{id}_\nu)^j \left(\frac{\Delta_j' - e_j}{b}\right)\right)\right) P_2 \\
&= \left(\alpha_2 + r_\nu \left(\sum_{j=0}^{m} (\mathsf{id}_\nu)^j \Delta_j\right)\right) P_2.
\end{aligned}$$

Case 2: $\nu < k$

In this case, $\mathscr{B}_2$ first creates a normal key $\mathcal{SK}_{\mathsf{id}_\nu}$ and runs $\mathcal{IBBE}_1$.SFKeyGen on $\mathcal{SK}_{\mathsf{id}_\nu}$. This is possible because the only scalar used in $\mathcal{IBBE}_1$.SFKeyGen is b which is known to $\mathscr{B}_2$.

Case 3: $\nu = k$

$\mathscr{B}_2$ embeds the DDH2 instance (consisting of $P_2, cP_2, rP_2, (rc+\gamma)P_2$) in the key $\mathcal{SK}_{\mathsf{id}_k} = (D_1, \ldots, D_5)$ for id_k by generating the components as shown below.

$$D_1 = rP_2,\ D_2 = (rc + \gamma)P_2,$$

$$\begin{aligned}
D_3 &= \alpha_1 P_2 + \left(\sum_{j=0}^{m} (\mathsf{id}_k)^j e_{j,1}\right)(rP_2) + \left(\sum_{j=0}^{m} (\mathsf{id}_k)^j e_{j,2}\right)(rc+\gamma)P_2 \\
&= \alpha_1 P_2 + r\left(\sum_{j=0}^{m} (\mathsf{id}_k)^j (e_{j,1} + ce_{j,2})\right) P_2 + \gamma \left(\sum_{j=0}^{m} (\mathsf{id}_k)^j e_{j,2}\right) P_2 \\
&= \left(\alpha_1 + r\left(\sum_{j=0}^{m} (\mathsf{id}_k)^j e_j\right)\right) P_2 + \gamma \left(\sum_{j=0}^{m} (\mathsf{id}_k)^j e_{j,2}\right) P_2,
\end{aligned}$$

$$D_4 = b^{-1}(\Delta' rP_2 - (rc+\gamma)P_2) = r\left(\frac{\Delta' - c}{b}\right) P_2 - \left(\frac{\gamma}{b}\right) P_2 = r\Delta P_2 - \left(\frac{\gamma}{b}\right) P_2,$$

$$\begin{aligned}
D_5 &= b^{-1}\left(\sum_{j=0}^{m}(\mathsf{id}_k)^j(\Delta'_j - e_{j,1})\right)(rP_2) - b^{-1}\left(\sum_{j=0}^{m}(\mathsf{id}_k)^j e_{j,2}\right)(rc+\gamma)P_2 \\
&= b^{-1}r\left(\sum_{j=0}^{m}(\mathsf{id}_k)^j(\Delta'_j - e_j)\right)P_2 - b^{-1}\gamma\left(\sum_{j=0}^{m}(\mathsf{id}_k)^j e_{j,2}\right)P_2 \\
&= r\left(\sum_{j=0}^{m}(\mathsf{id}_k)^j\left(\frac{\Delta'_j - e_j}{b}\right)\right)P_2 - \left(\frac{\gamma}{b}\right)\left(\sum_{j=0}^{m}(\mathsf{id}_k)^j e_{j,2}\right)P_2 \\
&= r\left(\sum_{j=0}^{m}(\mathsf{id}_k)^j\Delta_j\right)P_2 - \left(\frac{\gamma}{b}\right)\left(\sum_{j=0}^{m}(\mathsf{id}_k)^j e_{j,2}\right)P_2,
\end{aligned}$$

implicitly setting $r_k = r$ and $\gamma_k = \gamma$. When $\gamma = 0$, $\mathcal{SK}_{\mathsf{id}_k}$ is normal; otherwise, it is semi-functional with $\pi_k = \sum_{j=0}^{m}(\mathsf{id}_k)^j e_{j,2}$ set implicitly.

Challenge: $\mathscr{B}_2$ obtains the challenge set $\widehat{S} = \{\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}}\}$ from $\mathscr{A}$. It then picks $s, \mu \xleftarrow{\text{U}} \mathbb{Z}_p$ and generates semi-functional key K_0 and header $\widehat{\mathsf{Hdr}} = (\widehat{C}_1, \widehat{C}_2, (\widehat{C}_{3,i}, \widehat{\mathsf{tag}}_i)_{i=1}^{\widehat{\ell}})$ as follows.

$$\begin{aligned}
&\widehat{\mathsf{tag}}_i = -\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j e_{j,2}, \\
&K_0 = g_T^s \cdot e(P_1, P_2)^{\alpha_1\mu}, \\
&\widehat{C}_1 = sP_1 + \mu P_1, \\
&\widehat{C}_2 = sbP_1, \\
&\text{For } i = 1, \ldots, \widehat{\ell}, \\
&\widehat{C}_{3,i} = s\left(\textstyle\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j U_{1,j} + \widehat{\mathsf{tag}}_i W_1\right) + \mu\left(\textstyle\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j e_{j,1}\right)P_1 \\
&\quad = s\left(\textstyle\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j U_{1,j} + \widehat{\mathsf{tag}}_i W_1\right) \\
&\qquad +\mu\left(\textstyle\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j(e_{j,1} + ce_{j,2}) + \widehat{\mathsf{tag}}_i \cdot c\right)P_1 - \mu\left(\textstyle\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j ce_{j,2}\right)P_1 - \widehat{\mathsf{tag}}_i \cdot c\mu P_1 \\
&\quad = s\left(\textstyle\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j U_{1,j} + \widehat{\mathsf{tag}}_i W_1\right) \\
&\qquad +\mu\left(\textstyle\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j e_j + \widehat{\mathsf{tag}}_i \cdot c\right)P_1 - c\mu\left(\textstyle\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j e_{j,2} + \widehat{\mathsf{tag}}_i\right)P_1 \\
&\quad = s\left(\textstyle\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j U_{1,j} + \widehat{\mathsf{tag}}_i W_1\right) + \mu\left(\textstyle\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j e_j + \widehat{\mathsf{tag}}_i \cdot c\right)P_1.
\end{aligned}$$

The last step follows due to the fact that $\widehat{\mathsf{tag}} = -\sum_{j=0}^{m}(\widehat{\mathsf{id}}_i)^j e_{j,2}$. $\mathscr{B}_2$ chooses $K_1 \xleftarrow{\text{U}} \mathbb{G}_T$, $\beta \xleftarrow{\text{U}} \{0,1\}$ and returns $(\widehat{\mathsf{Hdr}}, K_\beta)$ to $\mathscr{A}$. Note that $\widehat{\mathsf{Hdr}}$ and K_0 are properly formed. Also, this is the only way $\mathscr{B}_2$ can generate a semi-functional header-key pair since no encoding of c is available in the group $\mathbb{G}_1$. An implication is that $\mathscr{B}_2$ can only create a nominally semi-functional header component with index i for a set of intended recipients containing id_k as the i-th identity. This is because the relation $\mathsf{tag}_i = -\pi_k$ will hold. This provides no information to $\mathscr{B}_2$ about the semi-functionality of $\mathcal{SK}_{\mathsf{id}_k}$.

Guess: $\mathscr{A}$ returns its guess β' of β.

$\mathscr{B}_2$ outputs 1 if $\mathscr{A}$ wins and 0 otherwise. Also, $\mathscr{B}_2$ simulates G_{k-1} if $\gamma = 0$ and G_k if $\gamma \xleftarrow{\text{U}} \mathbb{Z}_p$.

Therefore, the advantage of $\mathscr{B}_2$ in solving the DDH2 instance is given by

$$\begin{aligned}
\mathsf{Adv}_{\mathcal{G}}^{\mathrm{DDH2}}(\mathscr{B}_2) &= |\Pr[\mathscr{B}_2 \text{ returns } 1|\gamma = 0] - \Pr[\mathscr{B}_2 \text{ returns } 1|\gamma \xleftarrow{\mathrm{U}} \mathbb{Z}_p]| \\
&= |\Pr[\beta = \beta'|\mu = 0] - \Pr[\beta = \beta'|\mu \xleftarrow{\mathrm{U}} \mathbb{Z}_p]| \\
&= |\Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{k-1}] - \Pr[\mathscr{A} \text{ wins in } \mathsf{G}_k]| \\
&= |\Pr[X_{k-1}] - \Pr[X_k]|.
\end{aligned}$$

It now follows that $|\Pr[X_{k-1}] - \Pr[X_k]| \le \varepsilon_{\mathrm{DDH2}}$ from the fact that $\mathsf{Adv}_{\mathcal{G}}^{\mathrm{DDH2}}(\mathscr{B}) \le \varepsilon_{\mathrm{DDH2}}$ for all t-time adversaries $\mathscr{B}$. What remains is to show that all the information provided to the adversary have the correct distribution. The scalars $b, \alpha_1, \alpha_2', \Delta', (e_{j,1}, e_{j,2}, \Delta_j')_{j=0}^{m}$ chosen by $\mathscr{B}_2$ and r, c, γ from the instance are uniformly and independently distributed in their respective domains. These scalars determine the distribution of the following quantities.

- α_2, Δ
- $(e_j)_{j=0}^{m}$ and hence $(\Delta_j)_{j=0}^{m}$
- r_k, γ_k
- π_k
- $\widehat{\mathsf{tag}}_1, \ldots, \widehat{\mathsf{tag}}_{\widehat{\ell}}$

(α_2, Δ) are uniquely determined by (α_2', Δ'). Scalars r_k, γ_k have the correct distribution since they are set to r, γ respectively. Also, all other information is independent of r, γ. We will now argue that π_k and $\widehat{\mathsf{tag}}_1, \ldots, \widehat{\mathsf{tag}}_{\widehat{\ell}}$ are properly distributed. They are given by the following equation.

$$\begin{pmatrix} \pi_k \\ \widehat{\mathsf{tag}}_1 \\ \widehat{\mathsf{tag}}_2 \\ \vdots \\ \widehat{\mathsf{tag}}_{\widehat{\ell}} \end{pmatrix} = \begin{pmatrix} 1 & \mathsf{id}_k & (\mathsf{id}_k)^2 & \cdots & (\mathsf{id}_k)^m \\ 1 & \widehat{\mathsf{id}}_1 & (\widehat{\mathsf{id}}_1)^2 & \cdots & (\widehat{\mathsf{id}}_1)^m \\ 1 & \widehat{\mathsf{id}}_2 & (\widehat{\mathsf{id}}_2)^2 & \cdots & (\widehat{\mathsf{id}}_2)^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \widehat{\mathsf{id}}_{\widehat{\ell}} & (\widehat{\mathsf{id}}_{\widehat{\ell}})^2 & \cdots & (\widehat{\mathsf{id}}_{\widehat{\ell}})^m \end{pmatrix} \begin{pmatrix} e_{0,2} \\ e_{1,2} \\ \vdots \\ e_{m,2} \end{pmatrix} \tag{7.1}$$

One can make the following observations.

- $\mathsf{id}_k, \widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}}$ are all distinct since $\mathsf{id}_k \notin \widehat{S}$. Also $\widehat{\ell} \le m$. Hence the above matrix of order $(\widehat{\ell} + 1) \times (m + 1)$ over $\mathbb{Z}_p$ is a Vandermonde matrix and has rank $\widehat{\ell} + 1$.
- $e_{0,2}, e_{1,2}, \ldots, e_{m,2}$ are information theoretically hidden from $\mathscr{A}$ and also chosen uniformly and independently over $\mathbb{Z}_p$.

From these observations, it follows that $\pi_k, \widehat{\mathsf{tag}}_1, \ldots, \widehat{\mathsf{tag}}_{\widehat{\ell}}$ are uniformly and independently distributed in $\mathscr{A}$'s view.

The scalars $(\Delta_j)_{j=0}^{m}$ are uniquely determined by $(\Delta_j')_{j=0}^{m}$ and $(e_j)_{j=0}^{m}$. So all that we need to show is that the quantities $e_j = e_{j,1} + c e_{j,2}$ for $j \in [0, m]$ have the right distribution conditioned on π_k and tags being determined by $(e_{j,2})_{j=0}^{m}$. This follows from the fact that $e_{j,1}$'s are uniformly and independently distributed in $\mathbb{Z}_p$ thus making the e_j's uniform random quantities in $\mathbb{Z}_p$. $\square$

Lemma 7.1.3. $|\Pr[X_q] - \Pr[X_{final}]| \le q/p$.

Proof. In G_q, all the user keys returned to $\mathscr{A}$ are semi-functional and so is the challenge header and key. We now modify the setup and key extraction phases so that the modification results in G_{final} and then argue that the resulting game is indistinguishable from G_q except for probability q/p.

Setup: Pick scalars $\alpha_1, \alpha_2', \Delta', c, (\Delta_j', e_j)_{j=0}^m \xleftarrow{\text{U}} \mathbb{Z}_p$ and $b \xleftarrow{\text{U}} \mathbb{Z}_p^\times$ and compute parameters as:

$U_{1,j} = \Delta_j' P_1$ for $j = 0, \ldots, m$, $W_1 = \Delta' P_1$,
$g_T = e(P_1, P_2)^{\alpha_2'}$
$\mathcal{PP} : (P_1, bP_1, (U_{1,j})_{j=0}^m, W_1, g_T)$

setting

$$\alpha_2 = \frac{\alpha_2' - \alpha_1}{b}, \quad \Delta = \frac{\Delta' - c}{b},$$
$$\Delta_j = \frac{\Delta_j' - e_j}{b} \quad \text{for } j = 0, \ldots, m.$$

Although α_1 is sampled during setup, it has no effect on the distribution g_T and hence that of $\mathcal{PP}$. This is because g_T is created using α_2' which is chosen independent of α_1.

Key Extraction: On a key extract query for id, choose $r, \pi', \gamma \xleftarrow{\text{U}} \mathbb{Z}_p$, and compute the individual components as follows.

$$D_1 = rP_2,\ D_2 = rcP_2 + \gamma P_2,\ D_3 = \pi' P_2 + r\left(\sum_{j=0}^m (\mathsf{id})^j e_j\right) P_2,$$

$$D_4 = r\left(\frac{\Delta' - c}{b}\right) P_2 - \left(\frac{\gamma}{b}\right) P_2,$$

$$D_5 = \left(\frac{\alpha_2' - \pi'}{b}\right) + r\left(\sum_{j=0}^m (\mathsf{id})^j \Delta_j\right) P_2.$$

Computing D_3 as $D_3 = \pi' P_2 + r\left(\sum_{j=0}^m (\mathsf{id})^j e_j\right) P_2$ sets $\pi' = \alpha_1 + \gamma\pi$, where $\gamma\pi$ defines the semi-functional component. The other component where π' is used is D_5 that also fixes $\gamma\pi$ in its semi-functional term. It is necessary to ensure that these two are equal. We show below that D_5 is indeed well-formed in this sense.

$$\begin{aligned}
D_5 &= \left(\frac{\alpha_2' - \pi'}{b}\right) P_2 + r\left(\sum_{j=0}^m (\mathsf{id})^j \Delta_j\right) P_2 \\
&= \left(\frac{\alpha_2' - \alpha_1 - \gamma\pi}{b}\right) P_2 + r\left(\sum_{j=0}^m (\mathsf{id})^j \Delta_j\right) P_2 \\
&= \left(\frac{\alpha_2' - \alpha_1}{b}\right) P_2 + r\left(\sum_{j=0}^m (\mathsf{id})^j \Delta_j\right) P_2 - \left(\frac{\gamma\pi}{b}\right) P_2 \\
&= \left(\alpha_2 + r\left(\sum_{j=0}^m (\mathsf{id})^j \Delta_j\right)\right) P_2 - \left(\frac{\gamma\pi}{b}\right) P_2.
\end{aligned}$$

The scalar π is determined by α_1, π' and γ. It will be uniformly distributed in $\mathbb{Z}_p$ unless $\gamma = 0$. Furthermore, D_3 and D_5 are generated using π' which is chosen independent of α_1, thus making the key independent of α_1.

Challenge: The header and K_0 for the challenge set of privileged users $\widehat{S} = \{\mathsf{id}_1, \dots, \mathsf{id}_{\widehat{\ell}}\}$ are computed as:

$s, \mu \xleftarrow{\text{U}} \mathbb{Z}_p$, $(\mathsf{tag}_i)_{i=1}^{\widehat{\ell}} \xleftarrow{\text{U}} \mathbb{Z}_p$,
$K_0 = g_T^s \cdot e(P_1, P_2)^{\alpha_1 \mu}$,
$\widehat{C}_1 = sP_1 + \mu P_1$,
$\widehat{C}_2 = sbP_1$,
For $i = 1, \dots, \widehat{\ell}$,
$\widehat{C}_{3,i} = s\left(\sum_{j=0}^{m} (\widehat{\mathsf{id}}_i)^j \Delta'_j + \widehat{\mathsf{tag}}_i \Delta'\right) P_1 + \mu\left(d + \sum_{j=0}^{m} (\widehat{\mathsf{id}}_i)^j e_j + \widehat{\mathsf{tag}}_i \cdot c\right) P_1$.

The computation above shows that the challenge header consisting of $\widehat{C}_1, \widehat{C}_2, (\widehat{C}_{3,i}, \widehat{\mathsf{tag}}_i)_{i=1}^{\widehat{\ell}}$ is generated independent of α_1. Recall that α_1 is chosen independently and uniformly at random from $\mathbb{Z}_p$. Also, public parameters and all the keys are generated independent of α_1. Hence the conditional distribution of α_1 given the public parameters, keys and the challenge header is the same as its unconditional distribution. As a result, K_0 would be uniformly distributed in $\mathbb{G}_T$ and independent of all other information provided to $\mathscr{A}$. Therefore the bit β is information theoretically hidden from the adversary implying that the resulting game (obtained by modifying $\mathcal{IBBE}_1$.SFKeyGen) is G_{final}.

Suppose the adversary makes queries on $\mathsf{id}_1, \dots, \mathsf{id}_q$. Let γ_i denote the scalar used in generating the semi-functional components in $\mathcal{SK}_{\mathsf{id}_i}$ and let F_i ($i \in [1, q]$) denote the event that $\gamma_i = 0$. Clearly, $\Pr[F_i] = 1/p$ for a fixed i. Observe that G_q and G_{final} proceed identically unless the failure event $\mathsf{F} = \cup_{i=1}^{q} \mathsf{F}_i$ occurs. By the difference lemma (Shoup [149]), we have $|\Pr[\mathsf{G}_q] - \Pr[\mathsf{G}_{final}]| \le \Pr[F] \le \sum_{i=1}^{q} \Pr[\mathsf{F}_i] = q/p$. □

7.2 Towards Shorter Headers Without Random Oracles

The header size in $\mathcal{IBBE}_1$ is $(\ell + 2)N_1 + \ell N_p$ for a recipient set of size ℓ ($\le m$). As discussed earlier, we cannot do much with the identity hashes and neither can the tags be completely eliminated. One way of tackling the tags is to use a random oracle as has also been mentioned earlier. The question that we address here is whether the issue of increase in the ciphertext size due to the use of tags can be alleviated without resorting to random oracles.

In this section, we provide an answer to this question which results in a trade-off between the number of tags and the number of session key encapsulations. The resulting scheme, which we call $\mathcal{IBBE}_2$, operates as follows. Partition the privileged users' set and encapsulate the session key separately to each subset in the partition by applying the encapsulation algorithm of $\mathcal{IBBE}_1$. These separate encapsulations are not completely independent. The tags are reused across encapsulations. Below, we provide an overview of the scheme followed by the formal details.

Let the maximum size of the privileged users' set be $m = m_1 m_2$. Initialise an $\mathcal{IBBE}_1$ system with m_2 as the input to the Setup algorithm. Suppose we want to encrypt to a set S of size $\ell \le m$.

1. Express ℓ as $\ell = (\ell_1 - 1)m_2 + \ell_2$ where $1 \le \ell_1 \le m_1$ and $1 \le \ell_2 \le m_2$.

2. Partition S into ℓ_1 disjoint subsets $S_1, \ldots, S_{\ell_1}$ so that $|S_j| = m_2$ for $j = 1, \ldots \ell_1 - 1$ and $|S_{\ell_1}| = \ell_2$.
3. Choose random tags $\mathsf{tag}_1, \ldots, \mathsf{tag}_{m_2}$ from $\mathbb{Z}_p$. (We need m_2 tags since each subset S_j is of size at most m_2.)
4. Run $\mathcal{IBBE}_1$.Encap on each S_j (for $j \in [1, \ell_1]$) separately with the tags set to $\mathsf{tag}_1, \ldots, \mathsf{tag}_{m_2}$.

This results in ℓ_1 $\mathcal{IBBE}_1$ headers (referred to as *sub-headers*) with each sub-header consisting of at most m_2 elements of $\mathbb{G}_1$. The $\mathcal{IBBE}_2$ header consists of these sub-headers and the m_2 tags used to construct all the ℓ_1 sub-headers in addition to ℓ_1 elements of $\mathbb{G}_T$ each masking the session key.

The above idea is made concrete below as the scheme

$$\mathcal{IBBE}_2 = (\mathcal{IBBE}_2.\mathsf{Setup}, \mathcal{IBBE}_2.\mathsf{Encap}, \mathcal{IBBE}_2.\mathsf{KeyGen}, \mathcal{IBBE}_2.\mathsf{Decap})$$

whose individual algorithms are as follows.

$\mathcal{IBBE}_2.\mathsf{Setup}(\kappa, m = m_1 m_2 - 1)$: Let $(\mathcal{PP}', \mathcal{MSK}') \xleftarrow{\mathrm{R}} \mathcal{IBBE}_1.\mathsf{Setup}(\kappa, m_2)$. Define $\mathcal{PP} = (\mathcal{PP}', m_2)$ and $\mathcal{MSK} = \mathcal{MSK}'$.

$\mathcal{IBBE}_2.\mathsf{KeyGen}(\mathcal{MSK}, \mathsf{id})$: Return $(\mathcal{SK}_{\mathsf{id}} = (D_1, D_2, D_3, D_4, D_5)) \xleftarrow{\mathrm{R}} \mathcal{IBBE}_1.\mathsf{KeyGen}(\mathcal{MSK}, \mathsf{id})$.

$\mathcal{IBBE}_2.\mathsf{Encap}(\mathcal{PP}, S = \{\mathsf{id}_1, \ldots, \mathsf{id}_\ell\})$: Suppose $\ell = (\ell_1 - 1)m_2 + \ell_2$ with $1 \le \ell_1 \le m_1$ and $1 \le \ell_2 \le m_2$. Partition the set S into ℓ_1 disjoint subsets $S_1, S_2, \ldots, S_{\ell_1}$ where $|S_j| = m_2$ for all $j \in [1, \ell_1 - 1]$ and $|S_{\ell_1}| = \ell_2$. Choose $(s_j)_{j=1}^{\ell_1}, (\mathsf{tag}_i)_{i=1}^{m_2} \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and set

$$(\mathsf{Hdr}_j, K_j) \longleftarrow \mathcal{IBBE}_1.\mathsf{Encap}(\mathcal{PP}', S_j; s_j, \mathsf{tag}_1, \ldots, \mathsf{tag}_{m_2}) \text{ for } j = 1, \ldots, \ell_1.$$

Recall that the notation $\mathcal{A}(\cdot; R)$ denotes running the probabilistic algorithm $\mathcal{A}(\cdot)$ with its random bits set to R.

Choose a session key $K' \xleftarrow{\mathrm{U}} \mathbb{G}_T$ and mask it separately using $K_1, \ldots, K_{\ell_1}$ as follows.

$$C_{0,j} = K' \cdot K_j \text{ for } j \in [1, \ell_1]. \tag{7.2}$$

The header is

$$\vec{\mathsf{Hdr}} = (\mathsf{Hdr}_1, \ldots, \mathsf{Hdr}_{\ell_1}, C_{0,1}, \ldots, C_{0,\ell_1}, \mathsf{tag}_1, \ldots, \mathsf{tag}_{m_2}). \tag{7.3}$$

The actual message is encrypted using the session key K'.

$\mathcal{IBBE}_2.\mathsf{Decap}(\mathcal{PP}, S, \mathsf{id}, \mathcal{SK}_{\mathsf{id}}, \vec{\mathsf{Hdr}})$: Parse S as $(S_1, \ldots, S_{\ell_1})$ and suppose that $\mathsf{id} \in S_j$ for some $j \in [1, \ell_1]$. The session key K is derived as: $K' = C_{0,t} \cdot K_j^{-1}$ where $K_j = \mathcal{IBBE}_1.\mathsf{Decap}(\mathcal{PP}', S_j, \mathsf{id}, \mathcal{SK}_{\mathsf{id}}, \mathsf{Hdr}_j, (\mathsf{tag}_i)_{i=1}^{m_2})$.

Correctness. It is straightforward to verify that the correctness of decapsulation follows from that of $\mathcal{IBBE}_1$.

Masked copies of the session key: The message is encrypted using the session key K' and $C_{0,j}$, $1 \le j \le \ell_1$, are the masked copies of K'. In the above description, K' is from $\mathbb{G}_T$ since this is convenient for the security analysis. In practice, however, K' will be the key for a symmetric encryption scheme and hence will be a κ-bit string, where κ is the security parameter. In this case, the quantities $C_{0,j}$ will be generated as $\mathsf{KDF}(K_j) \oplus K'$, where KDF is a key derivation function which maps an element of $\mathbb{G}_T$ to a κ-bit string. As a result, $C_{0,1}, \ldots, C_{0,\ell_1}$ consists of ℓ_1 κ-bit strings. While considering the efficiency of $\mathcal{IBBE}_2$, we will consider the $C_{0,j}$'s to be κ-bit strings. For the security analysis, on the other hand, we will proceed with considering the $C_{0,j}$'s to be elements of $\mathbb{G}_T$. Modifying this security analysis to consider $C_{0,j}$'s to be κ-bit strings will require considering the security of KDF. This is quite routine and hence we skip it.

Header size for $\mathcal{IBBE}_2$: The total size of the $\mathcal{IBBE}_2$ header is $(\ell+2\ell_1)N_1+m_2N_p+\ell_1\kappa$ (assuming $C_{0,j}$'s to be κ-bit strings). In comparison, the header size for $\mathcal{IBBE}_1$ is $(\ell+2)N_1+\ell N_p$. A reasonable estimate of the group sizes is $N_1 = 2N_p$ and $N_p = 2\kappa$. Also, assume that m_1 and m_2 are around $\sqrt{m}$. For small ℓ, the header sizes of the two IBBE schemes are comparable. For ℓ around m, the header size of $\mathcal{IBBE}_2$ is smaller for $m \geq 25$.

Generating tags using a random oracle. As in the case of $\mathcal{IBBE}_1$, it is possible to construct a variant $\mathcal{IBBE}_2^{\mathrm{RO}}$ of $\mathcal{IBBE}_2$ that is adaptively secure with random oracles. The tags used in encryption are generated using a random oracle as in $\mathcal{IBBE}_1^{\mathrm{RO}}$. The construction $\mathcal{IBBE}_2^{\mathrm{RO}}$ can be obtained by just replacing $\mathcal{IBBE}_1$ by $\mathcal{IBBE}_1^{\mathrm{RO}}$ in the description of $\mathcal{IBBE}_2$ above. Moreover, $\mathcal{IBBE}_2^{\mathrm{RO}}$ can be shown to be secure based on the assumption that $\mathcal{IBBE}_1^{\mathrm{RO}}$ is secure. The header for $\mathcal{IBBE}_2^{\mathrm{RO}}$ consists of $(\ell+2\ell_1)$ elements of $\mathbb{G}_1$ and ℓ_1 κ-bit masked versions of the session key and a single κ-bit quantity from which the m_2 tags are generated using the random oracle. In contrast, the header for $\mathcal{IBBE}_1^{\mathrm{RO}}$ consists of $(\ell+2)$ elements of $\mathbb{G}_1$ and a single κ-bit quantity from which the m_2 tags are generated. As a result, the header size for $\mathcal{IBBE}_2^{\mathrm{RO}}$ is greater than the header size for $\mathcal{IBBE}_1^{\mathrm{RO}}$. So, if the tags are to be generated using a hash function, which is modelled as a random oracle, then it is more advantageous to use $\mathcal{IBBE}_1^{\mathrm{RO}}$ than $\mathcal{IBBE}_2^{\mathrm{RO}}$. We note that the PP size of $\mathcal{IBBE}_2^{\mathrm{RO}}$ is lower than that of $\mathcal{IBBE}_1^{\mathrm{RO}}$, but, this is of lesser significance.

Restriction on the size of the identity set: As in the case of $\mathcal{IBBE}_1$, in the encapsulation algorithm we have assumed that the number of identities ℓ to which the message is to be encrypted is at most m. In case $\ell > m$, then the comment made in the context of $\mathcal{IBBE}_1$ also applies for $\mathcal{IBBE}_2$.

7.2.1 Security of $\mathcal{IBBE}_2$

We show that $\mathcal{IBBE}_2$ is secure if $\mathcal{IBBE}_1$ is secure. More precisely, we prove the following.

Theorem 7.2.1. *If $\mathcal{IBBE}_1$ is (ε, t, q)-*IND-BID-CPA*-secure then $\mathcal{IBBE}_2$ is (ε', t', q)-*IND-BID-CPA*-secure where $\varepsilon' \leq 2m_1\varepsilon$ and $t' = O(m_1 t)$.*

Proof. The proof is via a simple hybrid argument over the session key encryptions. Let $\mathscr{A}$ be a t-time IND-BID-CPA adversary against $\mathcal{IBBE}_2$. We show how to build IND-BID-CPA adversaries $\mathscr{B}_1, \ldots, \mathscr{B}_{\widehat{\ell_1}}$ (where $\widehat{\ell_1} \leq m_1$ is the size of the partition of the challenge set) all running in time t against $\mathcal{IBBE}_1$ such that $\mathsf{Adv}^{\text{ind-ibbe-cpa}}_{\mathcal{IBBE}_2}(\mathscr{A}) \leq \sum_{\nu=1}^{\widehat{\ell_1}} \mathsf{Adv}^{\text{ind-ibbe-cpa}}_{\mathcal{IBBE}_1}(\mathscr{B}_t)$. Since $\widehat{\ell_1} \leq m_1$, the statement of the theorem follows.

Define the following game sequence: $\mathsf{G}_0, \mathsf{G}_1, \ldots, \mathsf{G}_{\widehat{\ell_1}}$ where G_0 is the real ind-ibbe-cpa game; in G_ν ($\nu \in [1, \widehat{\ell_1}]$), the first ν encryptions of the session key are random and the rest are normally formed. Let $Y_\square$ denote the probability that $\mathscr{A}$ wins in $\mathsf{G}_\square$.

Transition from $\mathsf{G}_{\nu-1}$ to G_ν for $\nu \in [1, \widehat{\ell_1}]$: $\mathscr{B}_\nu$ receives the public parameters $\mathcal{PP}'$ of IBBE_1 from its challenger and returns $\mathcal{PP} = (\mathcal{PP}, m_2)$ to $\mathscr{A}$. A key extraction query on an identity id that $\mathscr{A}$ makes is answered with the secret key that $\mathscr{B}_\nu$ receives from its challenger on the same identity. In the challenge phase, $\mathscr{B}_\nu$ receives a set $\widehat{S}$ from $\mathscr{A}$ and partitions it as $(\widehat{S}_1, \ldots, \widehat{S}_{\widehat{\ell_1}})$ with each $|\widehat{S}_j| = m_2$ for $j \in [1, \widehat{\ell_1} - 1]$ and $|S_{\widehat{\ell_1}}| = \widehat{\ell_2}$. $\mathscr{B}_\nu$ provides $\widehat{S}_\nu$ to its challenger and obtains a pair $(\widehat{\mathsf{Hdr}}, K_\beta)$. It then extracts the tags in $\widehat{\mathsf{Hdr}}$, denoted $(\widehat{\mathsf{tag}}_i)_{i=1}^{\widehat{m_2}}$, picks a random bit $\delta \xleftarrow{\text{U}} \{0,1\}$ and sets

$$(\mathsf{Hdr}_j, K_j) \xleftarrow{\text{R}} \mathit{IBBE}_1.\mathsf{Encap}(\mathcal{PP}', S_j; (\mathsf{tag}_i)_{i=1}^{\widehat{m_2}}), \text{ for } j \in [1, \widehat{\ell_1}] \setminus \{\nu\},$$

$$K_0', K_1' \xleftarrow{\text{U}} \mathbb{G}_T,$$

$$C_{0,j} \xleftarrow{\text{U}} \mathbb{G}_T \text{ for } j \in [1, \nu - 1], \quad C_{0,j} \leftarrow K_\delta' \cdot K_j \text{ for } j = [\nu + 1, \widehat{\ell_1}],$$

$$\mathsf{Hdr}_\nu = \widehat{\mathsf{Hdr}}, \quad C_{0,\nu} \leftarrow K_\delta' \cdot K_\beta,$$

$$\widehat{\overrightarrow{\mathsf{Hdr}}} = \left((\mathsf{Hdr}_j, C_{0,j})_{j=1}^{\widehat{\ell_1}}, (\mathsf{tag}_i)_{i=1}^{\widehat{m_2}} \right).$$

$\mathscr{B}_\nu$ returns $\widehat{\overrightarrow{\mathsf{Hdr}}}, K_\delta'$ to $\mathscr{A}$. The adversary $\mathscr{A}$ returns its guess δ' of δ. $\mathscr{B}_\nu$ sets $\beta' = 1$ if $\delta = \delta'$; else it sets $\beta' = 0$ and returns β' to its challenger.

We have

$$\begin{aligned}
\mathsf{Adv}_{\mathit{IBBE}_1}^{\text{ind-ibbe-cpa}}(\mathscr{B}_\nu) &= \left| \Pr[\beta = \beta'] - \frac{1}{2} \right| \\
&= \left| \Pr[\beta' = 1 | \beta = 1] \Pr[\beta = 1] + \Pr[\beta' = 0 | \beta = 0] \Pr[\beta = 0] - \frac{1}{2} \right| \\
&= \frac{1}{2} \left| \Pr[\beta' = 1 | \beta = 1] - \Pr[\beta' = 1 | \beta = 0] \right| \\
&= \frac{1}{2} \left| \Pr[\delta = \delta' | \beta = 1] - \Pr[\delta = \delta' | \beta = 0] \right| \\
&= \frac{1}{2} \left| \Pr[\delta = \delta' \text{ in } \mathsf{G}_\nu] - \Pr[\delta = \delta' \text{ in } \mathsf{G}_{\nu-1}] \right| \\
&= \frac{1}{2} \left| \Pr[Y_\nu] - \Pr[Y_{\nu-1}] \right|.
\end{aligned}$$

Since $\Pr[Y_{\widehat{\ell_1}}] = 1/2$, we have $\mathsf{Adv}_{\mathit{IBBE}_2}^{\text{ind-ibbe-cpa}}(\mathscr{A}) = |\Pr[Y_0] - \Pr[Y_{\widehat{\ell_1}}]| \le \sum_{\nu=1}^{\widehat{\ell_1}} |\Pr[Y_{\nu-1}] - \Pr[Y_\nu]| = 2 \sum_{\nu=1}^{\widehat{\ell_1}} \mathsf{Adv}_{\mathit{IBBE}_1}^{\text{ind-ibbe-cpa}}(\mathscr{B}_\nu)$, as required. □

7.3 Comparison to Existing Schemes

Tables 7.1 and 7.2 provide comparison of IBBE_1, IBBE_2, $\mathit{IBBE}_1^{\text{RO}}$ and $\mathit{IBBE}_2^{\text{RO}}$ with previously known IBBE systems secure with and without random oracles respectively. The ones derived as special cases of inner-product encryption have been omitted due to reasons explained earlier. Apart from these, we have tried to include all previously known IBBE schemes appearing in the literature.

We consider the following schemes for comparison: the early selectively secure constructions [10, 11] based on random oracles (ROs); constructions in [46, 48] with selective security and without ROs; constant-size ciphertext IBBE schemes selectively secure (with and without ROs) proposed by Delerablee [64]; generic constructions of IBBE schemes from "wicked" IBE schemes by Abdalla-Kiltz-Neven [2] instantiated with BBG-HIBE (without ROs) and GS-HIBE (with ROs); two adaptively secure IBBE constructions proposed by Gentry and Waters [87] – one with linear size (in number of privileged users) ciphertexts and the other with sub-linear size ciphertexts referred to as (a) and (b) respectively and a variant of scheme (a) based on ROs.

The basis for comparison are the following parameters – type of pairing, number of group elements in $\mathcal{PP}$ (denoted #pp) from $\mathbb{G}_1$ and $\mathbb{G}_T$, number elements in Hdr (#hdr) from $\mathbb{G}_1$ and $\{0,1\}^n$ (in case a KDF is used), number of elements in a user key (#ukey) from $\mathbb{G}_2$ and $\mathbb{Z}_p$, number of pairings required for decryption, security model and computational assumptions. m denotes the maximum size of the privileged users' set. ℓ ($\leq m$) is the size of the intended recipient set chosen during encryption. In construction [87]-(b) as well as scheme $\mathcal{IBBE}_2$, the maximum number of privileged users is given by $m = m_1 m_2$. The size of the set of users chosen during encryption is given by $\ell = \ell_1 \ell_2$ where $\ell_1 \leq m_1$ and $\ell_2 \leq m_2$. In the comparison, we ignore descriptions of hash functions, pseudorandom functions (PRFs) and other parameters that do not have any significant effect on the space-efficiency.

In the paper by Gentry and Waters [87], construction (b) consists of ℓ_1 separate symmetric encryptions of the message under the ℓ_1 keys generated by calls to the encapsulation algorithm of construction (a). In practice, the ℓ_1 keys would be used to mask a single session key via a KDF and there would be single encryption of the message under the session key. We take this into account in the comparison tables.

The short-hand 'sID' is used to indicate selective identity security and 'aID' is used to indicate security against adaptive-identity attacks. 'CCA' stands for chosen ciphertext attack whereas 'CPA' stands for chosen plaintext attacks. Apart from [11] all other schemes, including ours, have been proved secure against CPA. While CCA-security is the final desired goal, the first challenge in the design of IBBE schemes is to be able to handle adaptive-identity attacks. Most of the research on this topic have focused on this goal. Given that our constructions provide satisfactory solutions to the first problem, adapting known techniques to efficiently achieve CCA-security should form the focus of future work.

The assumptions mentioned in the tables are as follows: decisional bilinear Diffie-Hellman (DBDH), Gap bilinear DH (Gap-BDH), decisional bilinear DH exponent (DBDHE), generalised decisional DH exponent (GDDHE), DBDHE sum (DBDHES), security of a pseudorandom function (PRF) and external DH (XDH). The XDH assumption is a single name for the two decisional Diffie-Hellman (DDH) assumptions in the groups $\mathbb{G}_1$ and $\mathbb{G}_2$.

Based on Tables 7.1 and 7.2, we have the following observations.

1. Apart from $\mathcal{IBBE}_1$, $\mathcal{IBBE}_1^{\mathrm{RO}}$, $\mathcal{IBBE}_2$, $\mathcal{IBBE}_2^{\mathrm{RO}}$ and the constructions of Gentry and Waters [87] (denoted (a), (b), (a)-ROM), all other schemes listed in the tables are secure only in the weaker selective identity model.

Scheme	Pairing	#pp		#hdr		#ukey		#dec	Security	Assumptions
		$\mathbb{G}_1$	$\mathbb{G}_T$	$\mathbb{G}_1$	$\{0,1\}^\kappa$	$\mathbb{G}_2$	$\mathbb{Z}_p$			
[10]	Type-1	3	–	$\ell+1$	–	1	–	2	sID-CPA	DBDH
[11]	Type-1	3	–	3ℓ	–	1	–	2	sID-CCA	Gap-BDH
[2] (from GS-HIBE)	Type-1	$m+2$	1	$\ell+1$	–	$O(m)$	–	$\ell+1$	sID-CPA	DBDH
[64]-ROM	Type-1	$m+2$	1	2	–	1	1	2	sID-CPA	GDDHE
[87]-(a)-ROM	Type-1	$4m+2$	–	4	–	1	1	2	aID-CPA	m-DBDHES
$\mathcal{IBBE}_1^{\mathrm{RO}}$	Type-3	$m+4$	1	$\ell+2$	1	5	–	3	aID-CPA	XDH
$\mathcal{IBBE}_2^{\mathrm{RO}}$	Type-3	m_2+4	1	$\ell+2\ell_1$	ℓ_1+1	5	–	3	aID-CPA	XDH

Table 7.1: Comparison of $\mathcal{IBBE}_1^{\mathrm{RO}}$ and $\mathcal{IBBE}_2^{\mathrm{RO}}$ with previously known IBBE systems in the random oracle model. In the case of Type-1 pairings, $\mathbb{G}_2$ is the same as $\mathbb{G}_1$.

Scheme	Pairing	#pp		#hdr			#ukey		#dec	Security	Assumptions
		$\mathbb{G}_1$	$\mathbb{G}_T$	$\mathbb{G}_1$	$\{0,1\}^\kappa$	$\mathbb{Z}_p$	$\mathbb{G}_2$	$\mathbb{Z}_p$			
[46]	Type-1	$m+4$	–	$\ell+1$	–	–	2	–	2	sID-CPA	DBDH
[48]	Type-1	$m+4$	–	2ℓ	–	–	$m+2$	–	2	sID-CPA	$(m+1)$-DBDHE
[2] (from BBG-HIBE)	Type-1	$m+4$	–	2	–	–	$\ell+1$	–	2	sID-CPA	$(\ell-1)$-DBDHE
[64]	Type-1	$m+2$	1	2	–	–	1	–	2	sID-CPA	GDDHE
[87]-(a)	Type-1	$4m+2$	–	4	–	ℓ	1	1	2	aID-CPA	m-DBDHES, PRF
[87]-(b)	Type-1	$4m_2+2$	–	$4\ell_1$	ℓ_1	ℓ_2	1	1	2	aID-CPA	m-DBDHES, PRF
$\mathcal{IBBE}_1$	Type-3	$m+4$	1	$\ell+2$	–	ℓ	5	–	3	aID-CPA	XDH
$\mathcal{IBBE}_2$	Type-3	m_2+4	1	$\ell+2\ell_1$	ℓ_1	m_2	5	–	3	aID-CPA	XDH

Table 7.2: Comparison of $\mathcal{IBBE}_1$ and $\mathcal{IBBE}_2$ with existing IBBE systems without random oracles. In the case of Type-1 pairings, $\mathbb{G}_2$ is the same as $\mathbb{G}_1$.

2. $\mathcal{IBBE}_1$, $\mathcal{IBBE}_2$ are the only known constructions to achieve adaptive security from the standard and static DDH assumptions. The GW constructions are based on non-standard assumptions.

3. The GW constructions have better ciphertext sizes whereas our constructions have better public parameter sizes. The trade-off is the use of a non-static assumption, i.e., the hardness assumption is parameterised by m.

4. Even though the new constructions achieve stronger security from the standard XDH assumption, this is not done at a loss in efficiency. The ciphertext size, user storage and also the encryption and decryption times are comparable to previous constructions. Apart from the comparison of the number of group elements, it is also to be noted that the new constructions use Type-3 pairings whereas the previous constructions used Type-1 pairings. This leads to significantly smaller sizes for $\mathbb{G}_1$ which leads to smaller ciphertexts and faster encryption and decryption algorithms.

7.4 From IB(B)E to PKBE: Dodis-Fazio Revisited

Dodis and Fazio [68] described a method to build a public-key broadcast encryption scheme from an identity-based encryption scheme. The core idea behind this conversion is a combinatorial structure called complete subtree (CS) symmetric key revocation scheme introduced by Naor, Naor and Lotspeich [117].

In the CS scheme, the number of users n is assumed to be a power of 2 and the users are organized as the leaves of a complete binary tree $\mathcal{T}$ of height $\log n$. If v is a node of $\mathcal{T}$, define $\mathcal{S}_v$ to be the set of all leaf nodes in the subtree rooted at v. Further, let $\mathscr{C}$ be the collection of $\mathcal{S}_v$ for all v in $\mathcal{T}$. A centre assigns keys to subsets in $\mathscr{C}$. During a pre-distribution phase, a user corresponding to a leaf node u receives keys for all subsets in $\mathscr{C}$ which contains u. During an actual broadcast, the centre identifies a set of r revoked users. A partition of the other $n-r$ users is created using subsets from $\mathscr{C}$. Suppose the partition consists of h subsets $\mathcal{S}_1, \ldots, \mathcal{S}_h$. The actual message is encrypted using a session key and the session key is then encrypted using the keys corresponding to the h subsets $\mathcal{S}_1, \ldots, \mathcal{S}_h$. The encryptions of the session key constitute the header. It has been shown in [117] that each user has to store $\log n$ keys and the size of the header is at most $r \log(n/r)$.

Dodis and Fazio [68] presented a method to combine the CS scheme with an IBE scheme to obtain a PKBE scheme. The idea is as follows. The role of the centre in the CS scheme is played by the PKG of the IBE scheme. Set-up of the PKBE scheme consists of the following steps:

- the PKG runs the Setup algorithm of an IBE scheme;
- assigns an identity $\mathsf{id}_{\mathcal{S}}$ to each subset $\mathcal{S}$ in the collection $\mathscr{C}$;
- generates corresponding keys $\mathcal{SK}_{\mathsf{id}_{\mathcal{S}}}$ using the KeyGen algorithm of the IBE scheme;
- provides each user u with $\mathcal{SK}_{\mathsf{id}_{\mathcal{S}}}$ for each $\mathcal{S}$ to which it belongs;
- publishes $\mathcal{PP}$ and the structure $\mathcal{T}$ as the public key of the PKBE scheme.

Here $\mathcal{PP}$ consists of the public parameters of the IBE scheme.

For an actual broadcast, an entity forms a partition of the set of privileged users as in the CS scheme. As before, suppose that the partition consists of h sets $\mathcal{S}_1, \ldots, \mathcal{S}_h$ from $\mathscr{C}$. Let the corresponding identities be $\mathsf{id}_{\mathcal{S}_1}, \ldots, \mathsf{id}_{\mathcal{S}_h}$. As in the CS scheme, the actual message is encrypted using a session key. Using $\mathcal{PP}$, the session key is encrypted h times to the identities $\mathsf{id}_{\mathcal{S}_1}, \ldots, \mathsf{id}_{\mathcal{S}_h}$. These encryptions of the session key form the header. A user in any of the $\mathcal{S}$'s has a secret key $\mathcal{SK}_{\mathsf{id}_{\mathcal{S}}}$ corresponding to id_i. This allows the user to decrypt the corresponding encryption of the session key. The security of the scheme follows from the security of the IBE scheme. A user needs to store $\log n$ IBE keys and a header consists of at most $r \log(n/r)$ IBE encryptions of the session key.

Developing upon the Dodis-Fazio agenda described above, we suggest that the CS scheme be combined with an identity-based *broadcast* encryption scheme to obtain a PKBE scheme. Most of the details will remain unchanged. The only difference will be in the encryption. Suppose as above that $\mathcal{S}_1, \ldots, \mathcal{S}_h$ is the partition of the set of all privileged users and let $\{\mathsf{id}_{\mathcal{S}_1}, \ldots, \mathsf{id}_{\mathcal{S}_h}\}$ be the set of identities corresponding these sets. The Dodis-Fazio transformation mentions that encryptions are to be made individually to these identities. Using an IBBE scheme, on the other hand, one can make a single encryption to the set of identities $\{\mathsf{id}_{\mathcal{S}_1}, \ldots, \mathsf{id}_{\mathcal{S}_h}\}$. Decryption will be as before. The advantage is that the header size will go down. It is routine to argue that the security of the scheme will follow from the security of the IBBE scheme.

To illustrate the trade-offs, suppose that the Dodis-Fazio transformation is instantiated with the JR-IBE-D. The resulting PKBE will have headers consisting of at most $3r \log(n/r), r \log(n/r), r \log(n/r)$ elements from $\mathbb{G}_1, \mathbb{G}_T, \mathbb{Z}_p$ respectively. If on the other hand, we use $\mathcal{IBBE}_1$ as the IBBE scheme to obtain a PKBE scheme from the CS scheme, the maximum header size will be $2 + r\log(n/r), 1, r\log(n/r)$ elements from $\mathbb{G}_1, \mathbb{G}_T, \mathbb{Z}_p$ respectively. The trade-off is that the size of the public parameters will go up. Since public parameters is a static quantity and needs to be downloaded once, the savings in the size of the ciphertext will far outweigh the increase in the size of the public parameters. In arriving at the figures $2 + r\log(n/r), 1, r\log(n/r)$, we have assumed that the number of elements h in the header is at most m, the parameter in the $\mathcal{IBBE}_1$ scheme. If, on the other hand, h is more than m, then this would lead to a header consisting of encryptions to $\lceil h/m \rceil$ sets of identities as mentioned earlier.

Naor, Naor and Lotspeich [117] described another symmetric key BE scheme called the subset difference (SD) scheme. Dodis and Fazio [68] showed how to use a HIBE to convert the SD scheme to a PKBE scheme. This is not relevant in the current context and hence, we do not discuss this any further.

Chapter 8

Bounded DFA-Based ABE with Adaptive Security

We construct DFA-based key-policy ABE schemes with bounded functionality in the public index model that achieve adaptive security without random oracles. The schemes are built upon composite order pairings that have natural structure (orthogonality and parameter hiding) suitable for dual system proofs. Using the dual system technique, the constructions are proved secure under static subgroup decision assumptions over composite-order pairings.

Waters' ABE construction. DFA-based ABE was formalised by Waters in [160]. In this system, a secret key is associated with a deterministic finite automaton (DFA) $\mathcal{M}$ and the ciphertext is defined for a message m and an index is a string w over the input alphabet of the DFA (Σ). Decryption succeeds if $\mathcal{M}$ accepts w. As a result, the system supports the class of regular languages. Waters also described a construction (denoted $\mathcal{W}$-$\mathcal{ABE}$ in this work) based on symmetric prime-order bilinear groups in the public index model. This construction was shown to be selectively secure without random oracles based on the eXpanded Decisional Bilinear Diffie-Hellman Exponent (XDBDHE) assumption parametrised by $\widehat{w}$, the length of the challenge string $\widehat{w}$. The proof uses a partitioning strategy in which the challenge string $\widehat{w}$ is embedded in the public parameters in such a way that in the security reduction, the simulator can create secret keys for any DFA $\mathcal{M}$ that does not accept $\widehat{w}$. Also, the secret key for $\mathcal{M}$ is created based on the computation of $\mathcal{M}$ on input $\widehat{w}$. The complex assumption is required to account for multiple occurrences of symbols from Σ in $\widehat{w}$ (which is of arbitrary length) while the public parameters are fixed based only on $|\Sigma|$.

Our Contribution. The main goal of this work is to build an adaptive secure DFA-based ABE system. One could simply lift the scheme $\mathcal{W}$-$\mathcal{ABE}$ to the composite order setting and try using the dual system technique to prove security. Let us see why such a direct adaptation of dual system method fails. Consider a system with Σ as the alphabet and a composite order pairing $\mathcal{G} = (p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, e, G)$ with $\mathcal{G}_{\text{pub}} = (N, \mathbb{G}, \mathbb{G}_T, e, G)$ as its public description. Since most DFAs used in practice have small alphabets, we can pick a group element $H_\sigma \xleftarrow{\text{U}} \mathbb{G}_{p_1}$ corresponding to each symbol $\sigma \in \Sigma$ and include these elements in the public parameters. Let $w = w_1 \cdots w_\ell$ be a string over Σ to which a ciphertext $\mathcal{C}$ is encrypted and $\mathcal{SK}_{\mathcal{M}}$, a secret key for an automaton

$\mathcal{M} = (Q, \Sigma, q_0, q_f, \delta)$. String w is encoded in $\mathcal{C}$ in such a way that the order of symbols is also maintained (as in [160]). Suppose that we attempt defining semi-functional components in the usual way. In the dual system method, semi-functional components for ciphertexts and keys usually mimic the structure of the normal ciphertexts and keys respectively. In the composite order setting, they are defined by adding components from group $\mathbb{G}_{p_2}$. Distribution of the additional components is required to be statistically hidden from the adversary's point of view for an effective security reduction. Since there is a single group element (H_σ) for each symbol σ, the entropy it provides is sufficient only for simulating one semi-functional component. If more than one components in the ciphertext or key correspond to the same symbol σ then their corresponding semi-functional components are simulated using the randomness provided by H_σ and hence revealed information theoretically. As a result, the dual system proof is affected.

The solution to this problem is to restrict the number of occurrences of symbols in transitions and strings during system setup. We adapt a technique previously used by Lewko et al. [112] in the context of attribute-based encryption over monotone access structures. A string w can contain at most one occurrence of each $\sigma \in \Sigma$. Similarly, at most one transition can contain a symbol σ. We call the resulting construction the *basic construction*, denoted $\mathscr{B}$-$\mathscr{ABE}$. This scheme supports only an extremely small class of languages. For instance, consider the alphabet $\{0, 1\}$. With the single-use restriction, then the scheme works for only 4 strings - $0, 1, 01, 10$! Nevertheless, this restriction can be relaxed and we show this via our next (full) construction, $\mathscr{F}$-$\mathscr{ABE}$. This scheme is obtained by putting a bound on the number of occurrences of each symbol in strings as well as transitions at setup. Suppose a symbol can appear at most $\mathsf{s_{max}}$ times in a string and at most $\mathsf{t_{max}}$ times in the set of transitions. Then our public parameters will contain $\mathsf{s_{max}} \times \mathsf{t_{max}}$ group elements corresponding to each symbol. Essentially H_σ is replaced by a matrix $\mathbf{H}_\sigma$ of order $\mathsf{s_{max}} \times \mathsf{t_{max}}$. Ciphertext and key are defined for w and $\mathcal{M}$ (respectively) in such a way that only one acceptance path and hence decryption sequence exists if $\mathcal{M}$ accepts w. Also, if $\mathcal{M}$ rejects w, then there is no way to decrypt. Since each entry in $\mathbf{H}_\sigma$ is distinct, simulating semi-functional components will no longer be a problem. If we assume $\mathsf{s_{max}}$ and $\mathsf{t_{max}}$ to be linear in κ, the security parameter, then this scheme supports a significantly large class of functionalities. Although the selectively secure scheme of [160] supports unbounded functionality, security is only limited to bounded functionality for otherwise the ℓ-XDBDHE assumption becomes meaningless[1]. On the other hand, our system is limited to bounded functionality in the construction itself and in addition is adaptively secure.

Pair Encoding and Predicate Encryption. In a recent work, Attrapadung [8] proposed the notion of *pair encoding schemes* and uses it to generically construct *predicate encryption* (PE) schemes. This work provides new insights into the dual system methodology and how to employ these in proving adaptive security of the generic PE constructions. As a result, PE for a large class of predicates are shown to have full security. This includes the DFA-based predicate i.e., the predicate encompassing the class of all regular languages. The constructions are based on composite-order pairings and have been translated to the prime-order setting in a follow-up work [9]. While adaptive security is obtained without imposing any restrictions as in our constructions, the proofs (in both settings) rely on parametrised assumptions such as the one used in [160]. Our proof, on the other hand, is based on static assumptions. Also, the constructions in [8, 9] are based on *large*

[1]As ℓ increases the assumption becomes stronger. In addition, the number of powers of a group element given out in the problem instance also increases. It has been reported in [57] that such instances are prone to attacks.

universe alphabets i.e., the alphabet size for the DFAs are of size super-polynomial or exponential in the security parameter. Some languages may have more efficient DFAs over small alphabets in comparison to large alphabets. Therefore, it is important to obtain adaptive security for regular languages over small alphabets.

Independent Work by Pandit and Barua [127]. Pandit and Barua [127] have independently obtained constructions of adaptively secure DFA-based ABE over finite regular languages achieving similar functionality as ours. While our constructions are based on composite-order bilinear pairings, they take the path of dual pairing vector spaces [119, 120] and obtain security from decisional linear (DLin) assumption.

8.1 Waters' DFA-Based ABE

We first review Waters' DFA-based ABE [160] based on symmetric pairings (following the Definition 2.1.7) in order to explain the intuition underlying our constructions. Let $\mathcal{W}\text{-}\mathcal{ABE} = (\mathcal{W}\text{-}\mathcal{ABE}.\mathsf{Setup}, \mathcal{W}\text{-}\mathcal{ABE}.\mathsf{KeyGen}, \mathcal{W}\text{-}\mathcal{ABE}.\mathsf{Encrypt}, \mathcal{W}\text{-}\mathcal{ABE}.\mathsf{Decrypt})$ denote the ABE construction of Waters [160] with the algorithms defined as follows.

$\mathcal{W}\text{-}\mathcal{ABE}.\mathsf{Setup}(\Sigma, \kappa)$: Generate a symmetric pairing $\mathcal{G} = (p, \mathbb{G}, \mathbb{G}_T, e, F)$ according to the security parameter κ. Choose a generator $P \xleftarrow{\text{U}} \mathbb{G}^\times$, elements $H_{\text{start}}, H_{\text{end}}, (H_\sigma)_{\sigma\in\Sigma}, U \xleftarrow{\text{U}} \mathbb{G}$ and $\alpha \xleftarrow{\text{U}} \mathbb{Z}_p$. The public parameters and master secret are given by

$\mathcal{PP}$: $(\mathcal{G}, \Sigma, P, H_{\text{start}}, H_{\text{end}}, (H_\sigma)_{\sigma\in\Sigma}, U, e(P,P)^\alpha)$,
$\mathcal{MSK}$: $(-\alpha P)$.

$\mathcal{W}\text{-}\mathcal{ABE}.\mathsf{Encrypt}(\mathcal{PP}, w = w_1 \cdots w_\ell, m)$: Choose randomisers $s_0, s_1, \ldots, s_\ell \xleftarrow{\text{U}} \mathbb{Z}_p$. Compute the ciphertext elements as follows.

$$C_m = m \cdot e(P,P)^{\alpha s_\ell},$$

$$C_{0,1} = C_{\text{start},1} = s_0 P, \quad C_{\text{start},2} = s_0 H_{\text{start}},$$

For $i = 1, \ldots, \ell$,
$$C_{i,1} = s_i P, \quad C_{i,2} = s_i H_{w_i} + s_{i-1} U,$$

$$C_{\text{end},1} = C_{\ell,1} = s_\ell P, \quad C_{\text{end},2} = s_\ell H_{\text{end}}.$$

The ciphertext is given by $\mathcal{C} = (C_m, C_{\text{start},1}, C_{\text{start},2}, (C_{i,1}, C_{i,2})_{i\in[1,\ell]}, C_{\text{end},1}, C_{\text{end},2}, w)$.

$\mathcal{W}\text{-}\mathcal{ABE}.\mathsf{KeyGen}(\mathcal{MSK}, \mathcal{M} = (Q, \Sigma, q_0, q_f, \delta))$: For each $x \in \mathbb{Z}_{|Q|}$, pick $D_x \xleftarrow{\text{U}} \mathbb{G}$. Choose elements r_{start}, for all $t \in \mathcal{T}$, r_t and for all $q_x \in Q$, r_{end_x} uniformly and independently at random from $\mathbb{Z}_p$. Compute the elements of the key as follows.

$$K_{\text{start},1} = D_0 + r_{\text{start}} H_{\text{start}}, \quad K_{\text{start},2} = r_{\text{start}} P,$$

For all $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,
$$K_{t,1} = -D_x + r_t U, \quad K_{t,2} = r_t P, \quad K_{t,3} = D_y + r_t H_\sigma,$$

For all $q_x \in Q$,
$K_{\mathrm{end}_x,1} = -\alpha P + D_f + r_{\mathrm{end}_x} H_{\mathrm{end}}, \quad K_{\mathrm{end}_x,2} = r_{\mathrm{end}_x} P.$

The secret key for automaton $\mathcal{M}$ is given by $\mathcal{SK}_{\mathcal{M}} = (K_{\mathrm{start},1}, K_{\mathrm{start},2}, (K_{t,1}, K_{t,2}, K_{t,3})_{t \in \mathcal{T}}, (K_{\mathrm{end}_x,1}, K_{\mathrm{end}_x,2})_{q_x \in Q})$.

$\mathcal{W}$-$\mathcal{ABE}$.$\mathsf{Decrypt}(\mathcal{C}, \mathcal{SK}_{\mathcal{M}})$: Suppose that $\mathsf{Accept}(\mathcal{M}, w) = 1$ and $w = w_1 \cdots w_\ell$. Then there exists a sequence of transitions $t_1, t_2, \ldots, t_\ell$ with $t_i = (q_{x_{i-1}}, q_{x_i}, w_i)$ where $x_0 = 0$ and $q_{x_\ell} \in F$. Decryption consists of several stages of computation. First compute

$$\begin{aligned} A_0 &= e(C_{\mathrm{start},1}, K_{\mathrm{start},1}) e(C_{\mathrm{start},2}, K_{\mathrm{start},2})^{-1} \\ &= e(P, D_0)^{s_0} \end{aligned}$$

Then compute intermediate values A_i (for $i = 1, \ldots, \ell$) as follows.

$$\begin{aligned} A_i &= A_{i-1} \cdot e(C_{i-1,1}, K_{t_i,1}) e(C_{i,2}, K_{t_i,2})^{-1} e(C_{i,1}, K_{t_i,3}) \\ &= e(P, D_{x_i})^{s_i} \end{aligned}$$

The last intermediate $A_{\ell+1}$ is computed as

$$A_{\ell+1} = e(C_{\mathrm{end},1}, K_{\mathrm{end}_{x_\ell},1}) \cdot e(C_{\mathrm{end},2}, K_{\mathrm{end}_{x_\ell},2})^{-1} = e(P,P)^{-\alpha s_\ell} e(D_{x_\ell}, P)^{s_\ell}.$$

Using A_ℓ and $A_{\ell+1}$ the message is unmasked as shown below.

$$m = C_m \cdot A_{\ell+1} \cdot A_\ell^{-1}.$$

8.2 Basic Construction

Described here is our first construction of DFA-based attribute-based encryption scheme $\mathcal{B}$-$\mathcal{ABE}$ = ($\mathcal{B}$-$\mathcal{ABE}$.Setup, $\mathcal{B}$-$\mathcal{ABE}$.KeyGen, $\mathcal{B}$-$\mathcal{ABE}$.$\mathsf{Encrypt}$, $\mathcal{B}$-$\mathcal{ABE}$.$\mathsf{Decrypt}$) in the composite order pairing setting. We impose the following restrictions on automata and strings over which the scheme is built.

Restriction 1: Keys are created only for automata with a unique final state and a single transition corresponding to each symbol

Restriction 2: Input string (part of the ciphertext) can contain only a single occurrence of each symbol

These restrictions are required for the proof to go through. In Section 8.4, we describe how to extend the basic scheme $\mathcal{B}$-$\mathcal{ABE}$ to a full scheme $\mathcal{F}$-$\mathcal{ABE}$ with relaxed restrictions and similar security guarantee.

The construction is similar to $\mathcal{W}$-$\mathcal{ABE}$. Encryption is done in the group $\mathbb{G}_{p_1}$ but the structure is slightly different from that $\mathcal{W}$-$\mathcal{ABE}$. Components of the key are elements of $\mathbb{G}_{p_1 p_3}$ and have the same structure as the $\mathcal{W}$-$\mathcal{ABE}$ keys except that they are additionally randomised by elements of $\mathbb{G}_{p_3}$. The group $\mathbb{G}_{p_2}$ forms the semi-functional space.

$\mathcal{B}$-$\mathcal{ABE}$.$\mathsf{Setup}(\Sigma, \kappa)$: Generate a composite order pairing $\mathcal{G} = (p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, e, G)$ according to the security parameter κ. Choose elements $P, H_{\mathrm{start}}, H_{\mathrm{end}}, (H_\sigma, U_\sigma)_{\sigma \in \Sigma} \xleftarrow{\mathrm{U}} \mathbb{G}_{p_1}$, $P_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}$ and $\alpha \xleftarrow{\mathrm{U}} \mathbb{Z}_N$. The public parameters and master secret are given by

$\mathcal{PP}$: $(\mathcal{G}_{\text{pub}}, \Sigma, P, H_{\text{start}}, H_{\text{end}}, H_{\lambda}, (H_{\sigma}, U_{\sigma})_{\sigma\in\Sigma}, e(P,P)^{\alpha})$,
$\mathcal{MSK}$: $(-\alpha P, P_3)$.

In $\mathcal{W}$-$\mathcal{ABE}$, only a single element U was uses to maintain the link between consecutive symbols but here we require a separate group element U_{σ} corresponding to each symbol σ. This is helpful in the dual system proof.

$\mathcal{B}$-$\mathcal{ABE}$.KeyGen($\mathcal{MSK}, \mathcal{M} = (Q, \Sigma, q_0, q_f, \delta)$): For each $x \in \mathbb{Z}_{|Q|}$, pick $D_x \xleftarrow{\text{U}} \mathbb{G}_{p_1}$. Choose elements r_{start}, for all $t \in \mathcal{T}$, r_t and r_{end} uniformly and independently at random from $\mathbb{Z}_N$. Let $R_{\text{start},1}, R_{\text{start},2}$, $(R_{t,1}, R_{t,2}, R_{t,3})_{t\in\mathcal{T}}$ and $R_{\text{end},1}, R_{\text{end},2}$ be randomly chosen elements of $\mathbb{G}_{p_3}$. Compute the elements of the key as follows.

$K_{\text{start},1} = D_0 + r_{\text{start}}H_{\text{start}} + R_{\text{start},1}, \quad K_{\text{start},2} = r_{\text{start}}P + R_{\text{start},2},$

For all $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,
$K_{t,1} = -D_x + r_tU_{\sigma} + R_{t,1}, \quad K_{t,2} = r_tP + R_{t,2}, \quad K_{t,3} = D_y + r_tH_{\sigma} + R_{t,3},$

$K_{\text{end},1} = -\alpha P + D_f + r_{\text{end}}H_{\text{end}} + R_{\text{end},1}, \quad K_{\text{end},2} = r_{\text{end}}P + R_{\text{end},2}.$

Here D_f corresponds to the final state q_f. The secret key for automaton $\mathcal{M}$ is given by $\mathcal{SK}_{\mathcal{M}} = (K_{\text{start},1}, K_{\text{start},2}, (K_{t,1}, K_{t,2}, K_{t,3})_{t\in\mathcal{T}}, K_{\text{end},1}, K_{\text{end},2})$.

$\mathcal{B}$-$\mathcal{ABE}$.Encrypt($\mathcal{PP}, w = w_1 \cdots w_{\ell}, m$): Choose randomisers $s_0, s_1, \ldots, s_{\ell} \xleftarrow{\text{U}} \mathbb{Z}_N$. Compute the ciphertext elements as follows.

$C_m = m \cdot e(P,P)^{\alpha s_{\ell}},$

$C_{0,1} = C_{\text{start},1} = s_0P, \quad C_{\text{start},2} = s_0H_{\text{start}},$

For $i = 1, \ldots, \ell$,
$C_{i,1} = s_iP, \quad C_{i,2} = s_iH_{w_i} + s_{i-1}U_{w_i},$

$C_{\text{end},1} = C_{\ell,1} = s_{\ell}P, \quad C_{\text{end},2} = s_{\ell}H_{\text{end}}.$

The ciphertext is given by $\mathcal{C} = (C_m, C_{\text{start},1}, C_{\text{start},2}, (C_{i,1}, C_{i,2})_{i\in[1,\ell]}, C_{\text{end},1}, C_{\text{end},2}, w)$.

$\mathcal{B}$-$\mathcal{ABE}$.Decrypt($\mathcal{C}, \mathcal{SK}_{\mathcal{M}}$): Suppose that Accept$(\mathcal{M}, w) = 1$ and $w = w_1 \cdots w_{\ell}$. Then there exists a sequence of transitions $t_1, t_2, \ldots, t_{\ell}$ with $t_i = (q_{x_{i-1}}, q_{x_i}, w_i)$ where $x_0 = 0$ and $x_{\ell} = f$. Decryption consists of several stages of computation. First compute

$$\begin{aligned} A_0 &= e(C_{\text{start},1}, K_{\text{start},1})e(C_{\text{start},2}, K_{\text{start},2})^{-1} \\ &= e(P, D_0)^{s_0} \end{aligned}$$

Then compute intermediate values A_i (for $i = 1, \ldots, \ell$) as follows.

$$\begin{aligned} A_i &= A_{i-1} \cdot e(C_{i-1,1}, K_{t_i,1})e(C_{i,2}, K_{t_i,2})^{-1}e(C_{i,1}, K_{t_i,3}) \\ &= e(P, D_{x_i})^{s_i} \end{aligned}$$

The last intermediate $A_{\ell+1}$ is computed as

$$A_{\ell+1} = e(C_{\text{end},1}, K_{\text{end},1}) \cdot e(C_{\text{end},2}, K_{\text{end},2})^{-1} = e(P,P)^{-\alpha s_{\ell}}e(D_f, P)^{s_{\ell}}.$$

Using A_ℓ and $A_{\ell+1}$ the message is unmasked as shown below.

$$m = C_m \cdot A_{\ell+1} \cdot A_\ell^{-1}.$$

Correctness. To show that decryption is correct, we need to show that the intermediate values $A_0, A_{\ell+1}$ and A_i for $i \in [1, \ell]$ have the claimed structure. It is enough to show that if A_{i-1} has the right structure, then so does A_i. By induction on i, it follows that $A_\ell = e(P, D_{x_\ell})^{s_\ell}$ for $i \in [1, \ell]$.

$$\begin{aligned}
A_0 &= e(C_{\text{start},1}, K_{\text{start},1}) e(C_{\text{start},2}, K_{\text{start},2})^{-1} \\
&= e(s_0 P, D_0 + r_{\text{start}} H_{\text{start}} + R_{\text{start},1}) e(s_o H_{\text{start}}, r_{\text{start}} P + R_{\text{start},2})^{-1} \\
&= e(P, D_0)^{s_0} e(P, H_{\text{start}})^{s_0 r_{\text{start}}} e(H_{\text{start}}, P)^{-s_0 r_{\text{start}}} \\
&= e(P, D_0)^{s_0} \\
A_i &= A_{i-1} \cdot e(C_{i-1,1}, K_{t_i,1}) e(C_{i,2}, K_{t_i,2})^{-1} e(C_{i,1}, K_{t_i,3}) \\
&= e(P, D_{x_{i-1}})^{s_{i-1}} e(s_{i-1}P, -D_{x_{i-1}} + r_{t_i} U_{w_i} + R_{t_i,1}) e(s_i H_{w_i} + s_{i-1} U_{w_i}, r_{t_i} P + R_{t_i,2})^{-1} \\
&\qquad e(s_i P, D_{x_i} + r_{t_i} H_{w_i} + R_{t_i,3}) \\
&= e(P, D_{x_{i-1}})^{s_{i-1}} e(P, D_{x_{i-1}})^{-s_{i-1}} e(P, U_{w_i})^{s_{i-1} r_{t_i}} e(H_{w_i}, P)^{-s_i r_{t_i}} e(U_{w_i}, P)^{-s_{i-1} r_{t_i}} e(P, D_{x_i})^{s_i} e(P, H_{w_i})^{s_i r_{t_i}} \\
&= e(P, D_{x_i})^{s_i} \\
A_{\ell+1} &= e(C_{\text{end},1}, K_{\text{end},1}) \cdot e(C_{\text{end},2}, K_{\text{end},2})^{-1} \\
&= e(s_\ell P, -\alpha P + D_f + r_{\text{end}} H_{\text{end}} + R_{\text{end},1}) e(s_\ell H_{\text{end}}, r_{\text{end}} P + R_{\text{end},2})^{-1} \\
&= e(P, P)^{-\alpha s_\ell} e(P, D_f)^{s_\ell} e(P, H_{\text{end}})^{s_\ell r_{\text{end}}} e(H_{\text{end}}, P)^{-s_\ell r_{\text{end}}} \\
&= e(P, P)^{-\alpha s_\ell} e(D_f, P)^{s_\ell}
\end{aligned}$$

Note that $\mathbb{G}_{p_3}$ components get cancelled due to the orthogonality property of composite order groups.

Ciphertext-Policy FE. It is possible to obtain a ciphertext-policy FE scheme by constructing a dual of the above scheme. The structure of the ciphertext and key get interchanged. A key will encode a string w and a ciphertext will encode an automaton $\mathcal{M}$. Also, randomisation in $\mathbb{G}_{p_3}$ is done only for the key (i.e., components corresponding to the input string w). The same assumptions can also be used for the proof of security.

8.3 Security Proof for $\mathcal{B}$-$\mathcal{ABE}$

We require algorithms ReRandCT and ReRandK for randomising ciphertexts and keys respectively in the proof to ensure correct distribution of components. Essentially, these algorithms additively rerandomise ciphertexts and keys.

8.3.1 Algorithms for Rerandomisation

We describe the rerandomisation algorithms here. Except for the $\mathbb{G}_{p_3}$ components of the keys the algorithms are identical to those in [160].

ReRandCT($\mathcal{C}$): This algorithm picks $s'_0, s'_1, \ldots, s'_\ell \xleftarrow{\text{U}} \mathbb{Z}_N$ and modifies the ciphertext elements as shown below.

$$C_m \leftarrow C_m \cdot e(P,P)^{\alpha s'_\ell},$$

$$C_{\text{start},1} \leftarrow C_{\text{start},1} + s'_0 P, \quad C_{\text{start},2} \leftarrow C_{\text{start},2} + s'_0 H_{\text{start}},$$

For $i = 1, \dots, \ell$,
$$C_{i,1} \leftarrow C_{i,2} + s'_i P, \quad C_{i,2} \leftarrow C_{i,2} + s'_i H_{w_i} + s'_{i-1} P_1,$$

$$C_{\text{end},1} \leftarrow C_{\text{end},1} + s'_\ell P, \quad C_{\text{end},2} \leftarrow C_{\text{end},2} + s'_\ell H_{\text{end}}.$$

The new randomisers for the ciphertext will be $s_i + s'_i$ $(i = 0, \dots, \ell)$. The string w remains the same.

$\mathsf{ReRandK}(\mathcal{SK}_{\mathcal{M}})$: Choose uniform and independent random scalars r'_{start}, for all $t \in \mathcal{T}$, r'_t and r'_{end} from $\mathbb{Z}_N$. Also choose $D'_x \xleftarrow{\text{U}} \mathbb{G}_{p_1}$ for every $q_x \in Q$ and $R'_{\text{start},1}, R'_{\text{start},2}, \{R'_{t,1}, R'_{t,2}, R'_{t,3}\}_{t \in \mathcal{T}}, R'_{\text{end},1}, R'_{\text{end},2} \xleftarrow{\text{U}} \mathbb{G}_{p_3}$. Reconstruct components of the key as follows.

$$K_{\text{start},1} \leftarrow K_{\text{start},1} + D'_0 + r'_{\text{start}} H_{\text{start}} + R'_{\text{start},1}, \quad K_{\text{start},2} \leftarrow K_{\text{start},2} + r'_{\text{start}} P + R'_{\text{start},2}$$

For $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,
$$K_{t,1} \leftarrow K_{t,1} - D'_x + r'_t P_1 + R'_{t,1}, \quad K_{t,2} \leftarrow K_{t,2} + r'_t P + R'_{t,2}, \quad K_{t,3} \leftarrow K_{t,3} + D'_y + r'_t H_\sigma + R'_{t,3},$$

$$K_{\text{end},1} \leftarrow K_{\text{end},1} + D'_f + r'_{\text{end}} H_{\text{end}} + R'_{\text{end},1} \ ,$$
$$K_{\text{end},2} \leftarrow K_{\text{end},2} + r'_{\text{end}} P + R'_{\text{end},2}.$$

8.3.2 Defining Semi-Functionality

As usual, a dual system proof requires defining semi-functional ciphertexts and keys. Two types of semi-functional keys need to be defined for our proof of security – Type-1 and Type-2. Let P_2 be a random generator of the group $\mathbb{G}_{p_2}$ and

$$\pi_{\text{start}}, (\pi_{h,\sigma}, \pi_{u,\sigma})_{\sigma \in \Sigma} \xleftarrow{\text{U}} \mathbb{Z}_N.$$

These scalars are common to both semi-functional keys and ciphertexts.

Semi-functional Ciphertext
Pick $\gamma_0, \dots, \gamma_\ell, \pi_{\text{end}} \xleftarrow{\text{U}} \mathbb{Z}_N$. Semi-functional ciphertext is obtained by modifying normally generated ciphertext $\mathcal{C} = (C_m, C_{\text{start},1}, C_{\text{start},2}, (C_{i,1}, C_{i,2})_{i \in [1,\ell]}, C_{\text{end},1}, C_{\text{end},2}, w)$ as:

$$C_{\text{start},1} \leftarrow C_{\text{start},1} + \gamma_0 P_2, \quad C_{\text{start},2} \leftarrow C_{\text{start},2} + \gamma_0 \pi_{\text{start}} P_2,$$

For $i = 1, \dots, \ell$,
$$C_{i,1} \leftarrow C_{i,1} + \gamma_i P_2, \quad C_{i,2} \leftarrow C_{i,2} + (\gamma_i \pi_{h,w_i} + \gamma_{i-1} \pi_{u,w_i}) P_2,$$

$$C_{\text{end},1} \leftarrow C_{\text{end},1} + \gamma_\ell P_2, \quad C_{\text{end},2} \leftarrow C_{\text{end},1} + \pi_{\text{end}} P_2.$$

C_m remains unchanged. Restriction 2 mentioned in Section 8.2 is required here to ensure that only one value of $\pi_{h,\sigma}$ or $\pi_{u,\sigma}$ is revealed for any $\sigma \in \Sigma$ in the challenge ciphertext. Keeping value of $\pi_{\cdot,\sigma}$

statistically hidden is very essential for the security argument. On the other hand, providing too many copies of $\pi_{\cdot,\sigma}$ would information theoretically reveal its value to the adversary.

Type-1 Semi-functional Key

Let $\mu_{\text{start}}, \mu_{\text{end}}, (\mu_t)_{t\in\mathcal{T}}, \tau_{\text{end}} \overset{\text{U}}{\longleftarrow} \mathbb{Z}_N$, $(z_x)_{q_x\in Q} \overset{\text{U}}{\longleftarrow} \mathbb{Z}_N$ and $\mathcal{SK}_{\mathcal{M}} = (K_{\text{start},1}, K_{\text{start},2}, (K_{t,1}, K_{t,2}, K_{t,3})_{t\in\mathcal{T}}, K_{\text{end},1}, K_{\text{end},2})$ be a normal key generated by the $\mathcal{B}\text{-}\mathcal{ABE}$.KeyGen algorithm. Its components are modified as:

$$K_{\text{start},1} \leftarrow K_{\text{start},1} + (z_0 + \mu_{\text{start}}\pi_{\text{start}})P_2, \quad K_{\text{start},2} \leftarrow K_{\text{start},2} + \mu_{\text{start}}P_2,$$

For all $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,

$$K_{t,1} \leftarrow K_{t,1} + (z_x + \mu_t\pi_{u,\sigma})P_2, \quad K_{t,2} \leftarrow K_{t,2} + \mu_t P_2, \quad K_{t,3} \leftarrow K_{t,3} + (z_y + \mu_t\pi_{h,\sigma})P_2\ ,$$

$$K_{\text{end},1} \leftarrow K_{\text{end},1} + (z_f + \tau_{\text{end}})P_2, \quad K_{\text{end},2} \leftarrow K_{\text{end},2} + \mu_{\text{end}}P_2.$$

The first restriction plays a crucial role here. It ensures that the π-values are statistically hidden from the adversary.

Type-2 Semi-functional Key

Type 2 semi-functional keys are similar to Type-1 except that the components $K_{\text{start},1}, K_{\text{start},2}, (K_{t,1}, K_{t,2}, K_{t,3})_{t\in\mathcal{T}}$ will no longer have any semi-functional terms. Also, $K_{\text{end},1}$ does not contain the scalar z_f.

In the proof, it is ensured that at most one key can be Type-1 semi-functional at any point in the hybrid sequence of games. The rest of the semi-functional keys are Type-2. Otherwise, multiple copies of the π-values would have to be provided to the adversary and the whole purpose of imposing the two restrictions would be defeated.

Consider decryption of a ciphertext $\mathcal{C}$ for message m and string $w = w_1 \cdots w_\ell$ by a key $\mathcal{SK}_{\mathcal{M}}$ where $\mathsf{Accept}(\mathcal{M}, w) = 1$. Decryption succeeds unless both $\mathcal{C}$ and $\mathcal{SK}_{\mathcal{M}}$ semi-functional. This is because $\mathbb{G}_{p_2}$ (semi-functional) components get cancelled when paired with elements of $\mathbb{G}_{p_1}$ (by orthogonal property of composite order pairing groups). When both $\mathcal{C}$ and $\mathcal{SK}_{\mathcal{M}}$ are semi-functional, the message is masked by an extra factor - $e(P_2, P_2)^{(\mu_{\text{end}}\pi_{\text{end}} - \gamma_\ell\tau_{\text{end}})}$. To see this, note that all other semi-functional components get cancelled since they only mimic the structure of the ciphertext and key, in addition to having π-values common. Decryption will succeed only if $\mu_{\text{end}}\pi_{\text{end}} = \gamma_\ell\tau_{\text{end}}$. We will call such a pair of ciphertext and key as *nominally semi-functional.*

8.3.3 Reductions

We prove $\mathsf{IND\text{-}STR\text{-}CPA}$-security of $\mathcal{B}\text{-}\mathcal{ABE}$ under the three assumptions DSG1, DSG2 and SGDH.

Theorem 8.3.1. *If the* (ε_1, t')-DSG1*,* (ε_2, t')-DSG2*,* (ε_3, t')-SGDH *assumptions hold, then* $\mathcal{B}\text{-}\mathcal{ABE}$ *is* (ε, t, ν)-$\mathsf{IND\text{-}STR\text{-}CPA}$ *secure where*

$$\varepsilon \le \varepsilon_1 + 2\nu\varepsilon_2 + \varepsilon_3$$

and $t = t' - O(\nu|\Sigma|\rho)$*, where* ρ *is an upper bound on the time required for one scalar multiplication in* $\mathbb{G}$*.*

Proof. The proof is organised as a hybrid argument over a sequence of $2\nu + 3$ games – $\mathsf{G}_{real}, \mathsf{G}_{0,1}, (\mathsf{G}_{k,0}, \mathsf{G}_{k,1})_{k=1}^{\nu}, \mathsf{G}_{final}$. G_{real} denotes the actual CPA-security game for DFA-based ABE ind-abe-cpa (defined in Section 2.2.4). $\mathsf{G}_{0,1}$ is just like G_{real} except that the challenge ciphertext is semi-functional. In $\mathsf{G}_{k,0}$ (for $1 \le k \le \nu$), challenge ciphertext is semi-functional, the first $k-1$ keys returned to the adversary are Type-2 semi-functional, k-th key Type-1 semi-functional and the rest are normal. $\mathsf{G}_{k,1}$ $(1 \le k \le \nu)$ is such that first k keys are Type-2 semi-functional and rest are normal. G_{final} is similar to $\mathsf{G}_{\nu,1}$ except that now the challenge ciphertext is a semi-functional encryption of a random message. Let $\mathcal{E}_{\Box}$ denote the events that the adversary wins in $\mathsf{G}_{\Box}$. Note that, in G_{final}, the challenge ciphertext is an encryption of a random message and hence bit β is statistically hidden from the adversary's view implying that $\Pr[\mathcal{E}_{final}] = 1/2$.

The advantage of an t-time adversary $\mathscr{A}$ in winning the ind-abe-cpa against the ABE scheme in the ind-abe-cpa, is given by

$$\mathsf{Adv}_{\text{ABE}}^{\text{ind-abe-cpa}}(\mathscr{A}) = \left|\Pr[\mathcal{E}_{actual}] - \frac{1}{2}\right|.$$

We have

$$\begin{aligned}
\mathsf{Adv}_{\mathscr{B}\text{-}\mathscr{ABE}}^{\text{ind-abe-cpa}}(\mathscr{A}) &= \left|\Pr[\mathcal{E}_{actual}] - \Pr[\mathcal{E}_{final}]\right| \\
&\le |\Pr[\mathcal{E}_{actual}] - \Pr[\mathcal{E}_{0,1}]| + \sum_{k=1}^{\nu}\left(|\Pr[\mathcal{E}_{k-1,1}] - \Pr[\mathcal{E}_{k,0}]| + |\Pr[\mathcal{E}_{k,0}] - \Pr[\mathcal{E}_{k,1}]|\right) \\
&\qquad + |\Pr[\mathcal{E}_{\nu}] - \Pr[\mathcal{E}_{final}]| \\
&\le \varepsilon_{\text{DSG1}} + 2\nu\varepsilon_{\text{DSG2}} + \varepsilon_{\text{SGDH}}
\end{aligned}$$

The last inequality follows from the lemmas 8.3.1, 8.3.2, 8.3.3 and 8.3.4. □

In all the lemmas, $\mathscr{A}$ is a t-time adversary against the ABE scheme and $\mathscr{B}$ is an algorithm running in time t' that interacts with $\mathscr{A}$ and solves one of the three problems DSG1, DSG2 or SGDH.

Lemma 8.3.1. $|\Pr[\mathcal{E}_{actual}] - \Pr[\mathcal{E}_{0,1}]| \le \varepsilon_1$.

Proof. $\mathscr{B}$ receives an instance of problem DSG1, $(\mathcal{G}_{\text{pub}}, P, P_3, T)$, where $T = \theta P + \theta_2 P_2$ and its task is to decide whether $\theta_2 = 0$ or $\theta_2 \xleftarrow{\text{U}} \mathbb{Z}_{p_2}$. The different phases of the game are simulated as described below.

Setup: $\mathscr{B}$ picks $\alpha, v_{\text{start}}, v_{\text{end}}, \{v_{h,\sigma}, v_{u,\sigma}\}_{\sigma\in\Sigma} \xleftarrow{\text{U}} \mathbb{Z}_N$, sets $H_{\text{start}} = v_{\text{start}}P$, $H_{\text{end}} = v_{\text{end}}P$, $H_\sigma = v_{h,\sigma}P$ and $U_\sigma = v_{u,\sigma}P$. It provides $\mathcal{PP}$ to $\mathscr{A}$ and computes $\mathcal{MSK}$.

Key extraction queries: For a query on automaton $\mathcal{M}$, $\mathscr{B}$ runs the $\mathscr{B}\text{-}\mathscr{ABE}$.KeyGen algorithm with input $\mathcal{M}$ and returns the output to $\mathscr{A}$. No generator of $\mathbb{G}_{p_2}$ is provided to $\mathscr{B}$ and hence semi-functional keys cannot be generated.

Challenge: $\mathscr{A}$ provides two messages m_0, m_1, challenge string $\widehat{w} = \widehat{w}_1 \cdots \widehat{w}_{\widehat{\ell}}$. $\mathscr{B}$ chooses $\beta \xleftarrow{\text{U}} \{0,1\}$, $s'_0, \ldots, s'_{\widehat{\ell}} \xleftarrow{\text{U}} \mathbb{Z}_N$ and encrypts m_β to $\widehat{w}$ as follows.

$C_m = m_\beta \cdot e(P, T)^{\alpha s'_{\widehat{\ell}}}$,

$C_{0,1} = s'_0 T, \quad C_{\text{start},2} = s'_0 v_{\text{start}} T,$

For $i = 1, \ldots, \widehat{\ell},$
$\quad C_{i,1} = s'_i T, \quad C_{i,2} = (s'_i v_{h,w_i} + s'_{i-1} v_{u,w_i})T,$

$C_{\text{end},1} = C_{\ell,1}, \quad C_{\text{end},2} = s'_\ell v_{\text{end}} T.$

Randomiser s_i is inherently set to $s'_i\theta$ for $i = 0, \ldots, \widehat{\ell}$. Let $\widehat{\mathcal{C}} = (C_m, C_{\text{start},1}, C_{\text{start},2}, \{C_{i,1}, C_{i,2}\}_{i\in[1,\widehat{\ell}]}, C_{\text{end},1}, C_{\text{end},2}, w)$. $\mathscr{B}$ returns $\mathsf{ReRandCT}(\widehat{\mathcal{C}})$ to $\mathscr{A}$.

Guess: $\mathscr{A}$ returns its guess β'.

If $\theta_2 = 0$, then $\widehat{\mathcal{C}}$ is a normal encryption of m_β. Otherwise $\theta_2 \xleftarrow{\text{U}} \mathbb{Z}_{p_2}$ making $\widehat{\mathcal{C}}$ a semi-functional ciphertext for m_β with $\gamma_i = s'_i\theta_2$ for $i = 1, \ldots, \widehat{\ell}$, $\pi_{\text{start}} = v_{\text{start}}$, $\pi_{\text{end}} = s_{\ell'} v_{\text{end}}$, $\pi_{u,\sigma} = v_{u,\sigma}$ and $\pi_{h,\sigma} = v_{h,\sigma}$ for all $\sigma \in \Sigma$. The ciphertext is well-formed. For instance,

$$\begin{aligned} C_{i,2} &= (s'_i v_{h,w_i} + s'_{i-1} v_{u,w_i})T \\ &= s'_i v_{h,w_i}\theta P + s'_{i-1} v_{u,w_i}\theta P + s'_i v_{h,w_i}\theta_2 P_2 + s'_{i-1} v_{u,w_i}\theta_2 P_2 \\ &= s_i H_{w_i} + s_{i-1} U_{w_i} + (\gamma_i \pi_{h,w_i} + \gamma_{i-1}\pi_{u,w_i})P_2 \end{aligned}$$

The rest of the components can be shown to be well-formed in a similar way. The v's are embedded in the public parameters and hence their values modulo p_1 are revealed to the adversary in an information theoretic sense. However their values modulo p_2 remain hidden (by Chinese remainder theorem) thus resulting in the proper distribution of the π's. The s_i's are merely scaled by θ_2 to obtain γ_i's and hence the γ_i's are uniformly and independently distributed. The randomisers for the ciphertext's normal components are also properly distributed since it is rerandomised.

If the adversary wins the game then $\mathscr{B}$ returns 1; otherwise it returns 0. Therefore, we have

$$\begin{aligned} \varepsilon_1 \geq \mathsf{Adv}_{\mathcal{G}}^{\text{DSG1}}(\mathscr{B}) &= |\Pr[\mathscr{B} \text{ returns } 1 \mid T \xleftarrow{\text{U}} \mathbb{G}_{p_1}] - \Pr[\mathscr{B} \text{ returns } 1 \mid T \xleftarrow{\text{U}} \mathbb{G}_{p_1p_2}]| \\ &= |\Pr[\mathscr{A} \text{ wins} \mid T \xleftarrow{\text{U}} \mathbb{G}_{p_1}] - \Pr[\mathscr{A} \text{ wins} \mid T \xleftarrow{\text{U}} \mathbb{G}_{p_1p_2}]| \\ &= |\Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{actual}] - \Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{0,1}]| \\ &= |\Pr[\mathcal{E}_{actual}] - \Pr[\mathcal{E}_{0,1}]| \end{aligned}$$

as required. □

Lemma 8.3.2. *$|\Pr[\mathcal{E}_{k-1,1}] - \Pr[\mathcal{E}_{k,0}]| \leq \varepsilon_2$ for $1 \leq k \leq \nu$.*

Proof. An $(\mathcal{G}_{\text{pub}}, P, P_3, X + P_2, X_2 + X_3, T)$ of DSG2 is given to $\mathscr{B}$ and the goal is to decide whether $T \xleftarrow{\text{U}} \mathbb{G}_{p_1p_3}$ or $T \xleftarrow{\text{U}} \mathbb{G}$. In other words, if $T = \theta P + \theta_2 P_2 + \theta_3 P_3$ then $\mathscr{B}$ has to determine whether $\theta_2 = 0$ or $\theta_2 \xleftarrow{\text{U}} \mathbb{Z}_{p_2}$.

Setup: Scalars $\alpha, v_{\text{start}}, v_{\text{end}}, \{v_{u,\sigma}, v_{h,\sigma}\}_{\sigma\in\Sigma}$ are chosen from $\mathbb{Z}_N$ independently according to the uniform distribution. Parameters are set as follows: $H_{\text{start}} = v_{\text{start}}P$, $H_{\text{end}} = v_{\text{end}}P$, $H_\sigma = v_{h,\sigma}P$ and $U_\sigma = v_{u,\sigma}P$. $\mathcal{PP}$ is given to $\mathscr{A}$ and $\mathscr{B}$ keeps $\mathcal{MSK}$.

Key extraction queries: Suppose $\mathscr{A}$ makes key extraction queries on $\mathcal{M}_1, \ldots, \mathcal{M}_\nu$. $\mathscr{B}$ generates key for $\mathcal{M}_i$ depending on i as follows.

Case $i > k$: $\mathscr{B}$ runs the $\mathcal{B}$-$\mathcal{ABE}$.KeyGen algorithm and returns the resulting (normal) key to $\mathscr{A}$.

Case $i < k$: $\mathscr{B}$ first obtains $\mathcal{SK}_{\mathcal{M}_i} \longleftarrow \mathcal{B}$-$\mathcal{ABE}$.$\mathsf{KeyGen}(\mathcal{MSK}, \mathcal{M}_i)$ and then modifies its components to obtain a Type-2 semi-functional key for $\mathcal{M}_i$ as follows. Since a generator of $\mathbb{G}_{p_2}$ is not available, $\mathscr{B}$ uses element $X_2 + X_3$ to construct the semi-functional components.

$$\mu'_{\text{end}}, \tau'_{\text{end}} \xleftarrow{\text{U}} \mathbb{Z}_N,$$
$$K_{\text{end},1} \leftarrow K_{\text{end},1} + \tau'_{\text{end}}(X_2 + X_3), \quad K_{\text{end},2} \leftarrow K_{\text{end},2} + \mu'_{\text{end}}(X_2 + X_3).$$

The term $\mu_{\text{end}}P_2$ is set to $\mu'_{\text{end}}X_2$. Similarly, $\tau_{\text{end}}P_2 = \tau'_{\text{end}}X_2$. The components $K_{\text{end},1}, K_{\text{end},2}$ already have uniform random elements of $\mathbb{G}_{p_3}$ embedded in them. Hence adding multiples of X_3 will not change the distribution of the $\mathbb{G}_{p_3}$ components.

Case $i = k$: $\mathscr{B}$ computes $\mathcal{SK}_{\mathcal{M}_k}$ embedding the challenge T from the instance.

$$\text{For each } x \in \mathbb{Z}_{|Q|},\ d_x \xleftarrow{\text{U}} \mathbb{Z}_N$$
$$r'_{\text{start}}, r'_{\text{end}}, \{r'_t\}_{t\in\mathcal{T}} \xleftarrow{\text{U}} \mathbb{Z}_N$$

$$K_{\text{start},1} = (d_0 + r'_{\text{start}}v_{\text{start}})T, \quad K_{\text{start},2} = r'_{\text{start}}T,$$

$$\text{For all } t \in \mathcal{T} \text{ with } t = (q_x, q_y, \sigma) \text{ and } \sigma \in \Sigma,$$
$$K_{t,1} = (-d_x + r'_t v_{u,\sigma})T, \quad K_{t,2} = r'_t T, \quad K_{t,3} = (d_y + r'_t v_{h,\sigma})T,$$

$$K_{\text{end},1} = -\alpha P + (d_f + r'_{\text{end}}v_{\text{end}})T, \quad K_{\text{end},2} = r'_{\text{end}}T.$$

Let $\mathcal{SK}_{\mathcal{M}} = (K_{\text{start},1}, K_{\text{start},2}, \{K_{t,1}, K_{t,2}, K_{t,3}\}_{t\in\mathcal{T}}, K_{\text{end},1}, K_{\text{end},2})$. $\mathscr{B}$ returns $\mathsf{ReRandK}(\mathcal{SK}_{\mathcal{M}_k})$ to $\mathscr{A}$. We have $T = \theta P + \theta_2 P_2 + \theta_3 P_3$ where θ_2 could be zero. Hence every component is made up of elements of $\mathbb{G}_{p_1}$, $\mathbb{G}_{p_3}$ and possibly elements of $\mathbb{G}_{p_2}$. The $\mathbb{G}_{p_1}$ and $\mathbb{G}_{p_3}$ elements are properly distributed due to the invocation of $\mathsf{ReRandK}$ algorithm. If $\theta_2 = 0$, $\mathcal{SK}_{\mathcal{M}_k}$ is normal. Otherwise, $\theta_2 \xleftarrow{\text{U}} \mathbb{G}_{p_2}$ making $\mathcal{SK}_{\mathcal{M}_k}$ Type-1 semi-functional. The randomisers for the semi-functional components are set as: $z_x = d_x\theta_2$ for all $q_x \in Q$, $\mu_{\text{start}} = r'_{\text{start}}\theta_2$, $\mu_{\text{end}} = r'_{\text{end}}\theta_2$, $\mu_t = r'_t\theta_2$ for all $t \in \mathcal{T}$; $\pi_{\text{start}} = v_{\text{start}}$, $\pi_{u,\sigma} = v_{u,\sigma}$, $\pi_{h,\sigma} = v_{h,\sigma}$ for each $\sigma \in \Sigma$ and $\tau_{\text{end}} = r'_{\text{end}}v_{\text{end}}\theta_2$. Although v's are provided to the adversary via the public parameters, their values modulo p_2 remain hidden from the adversary (by Chinese remainder theorem). The μ's are uniformly distributed by the choice of r''s. Hence the π's and τ_{end} are uniformly distributed in $\mathscr{A}$'s view.

Challenge: $\mathscr{B}$ receives messages m_0, m_1 and challenge string $\widehat{w} = \widehat{w}_1 \cdots \widehat{w}_{\widehat{\ell}}$ from $\mathscr{A}$. It chooses $\beta \xleftarrow{\text{U}} \{0, 1\}$ and constructs ciphertext $\widehat{\mathcal{C}}$ as follows.

$$\gamma_0, \ldots, \gamma_{\widehat{\ell}} \xleftarrow{\text{U}} \mathbb{Z}_N$$

$$C_m = m_\beta \cdot e(P, X + P_2)^{\alpha\gamma_{\widehat{\ell}}},$$

$$C_{0,1} = \gamma_0(X + P_2), \quad C_{\text{start},2} = \gamma_0 v_{\text{start}}(X + P_2),$$

For $i = 1, \ldots, \widehat{\ell}$,

$$C_{i,1} = \gamma_i(X + P_2), \quad C_{i,2} = (\gamma_i v_{h,w_i} + \gamma_{i-1} v_{u,w_i})(X + P_2),$$

$$C_{\text{end},1} = C_{\ell,1}, \quad C_{\text{end},2} = \gamma_{\widehat{\ell}} v_{\text{end}}(X + P_2),$$

setting $s_i = \theta\gamma_i$ for $i \in [0, \widehat{\ell}]$. The output of $\mathsf{ReRandCT}(\widehat{\mathcal{C}})$ is returned to $\mathscr{A}$. The π values (except π_{end}) are set to the corresponding v's modulo p_2. These are equal to the π-values of the k-th key thus satisfying the requirements for Type-1 semi-functionality. Note that after calling $\mathsf{ReRandCT}$ the randomisers for the $\mathbb{G}_{p_1}$ components will have the proper distribution.

Guess: $\mathscr{A}$ sends $\mathscr{B}$ its guess β'.

We now show that the challenge ciphertext and k-th key are properly distributed in $\mathscr{A}$'s view with all but negligible probability. The following holds for the k-th key and the challenge ciphertext.

$$\mu_{\text{end}}\pi_{\text{end}} - \tau_{\text{end}}\gamma_{\widehat{\ell}} = (r'_{\text{end}}\theta_2)(\gamma_{\widehat{\ell}} v_{\text{end}}) - (r'_{\text{end}} v_{\text{end}}\theta_2)\gamma_{\widehat{\ell}} = 0 \pmod{p_2}.$$

The ciphertext-key pair will turn out to be nominally semi-functional. This is to ensure that $\mathscr{B}$ itself cannot create a semi-functional ciphertext for a string w' accepted by $\mathcal{M}_k$ that assists in determining whether $\mathcal{SK}_{\mathcal{M}_k}$ is semi-functional or not. Decryption succeeds and provides no information to $\mathscr{B}$ about the distribution of $\mathcal{SK}_{\mathcal{M}_k}$ and hence T. On the other hand, it is required to prove that this relation between the k-key and $\widehat{\mathcal{C}}$ is hidden from the adversary. The argument follows from three facts:

1. $\mathscr{A}$ cannot request keys for any automaton $\mathcal{M}$ that accepts $\widehat{w}$
2. the final state of any automaton $\mathcal{M}$ on which a query is made is not reachable on input $\widehat{w}$ (any automaton that is queried has a unique final state and hence a special symbol \$ based on which a transition to the final state is made only in case of acceptance)
3. each symbol appears at most once in strings or descriptions of automata

Consider a transition $t = (q_x, q_y, \sigma)$ in $\mathcal{M}$ and suppose the i-th set of components in $\widehat{\mathcal{C}}$ are for the symbol σ (i.e., $\widehat{w}_i = \sigma$). Then $C_{i,\cdot}$ and $K_{t,\cdot}$ components will share the same π-values. Assume that the μ_t and γ_i, γ_{i-1} values are statistically revealed to the adversary. It essentially gets hold of 3 equations (corresponding to semi-functional components of $K_{t,1}, K_{t,3}, C_{w,2}$) in 4 unknowns $(\pi_{h,\sigma}, \pi_{u,\sigma}, z_x, z_y)$. Using these the adversary cannot gain any information about these quantities. Thus they appear uniformly distributed in $\mathscr{A}$'s view. What remains is to show that the relation between π_{end} and τ_{end} remains information-theoretically hidden from the adversary. Observe that π_{end} is set to $\gamma_{\widehat{\ell}} v_{\text{end}}$ and τ_{end} to $r'_{\text{end}} v_{\text{end}}\theta_2$. The scalar $\gamma_{\widehat{\ell}}$ has the right distribution due to its choice and so is μ_{end} except when $\theta_2 = 0 \pmod{p_2}$ which occurs with negligible probability. Given that τ_{end} and π_{end} share the value of v_{end} modulo p_2, their value must be shown to be hidden from $\mathscr{A}$. Since $\mathcal{M}_k$ does not accept $\widehat{w}$, the (unique) final state is never reached (see fact 2 above). As a result the adversary cannot get hold of any equation that involves τ_{end} and any of the π-values. This especially holds for π_{end}. Furthermore, the single-occurrence restriction on each symbol implies that there is at most one equation involving π_{end}. Hence the k-th key and $\widehat{\mathcal{C}}$ remain properly distributed in $\mathscr{A}$'s view except with negligible probability.

If the adversary wins the game then $\mathscr{B}$ returns 1; otherwise it returns 0. Therefore, we have

$$\begin{aligned}\varepsilon_2 \geq \mathsf{Adv}_{\mathcal{G}}^{\mathrm{DSG2}}(\mathscr{B}) &= |\Pr[\mathscr{B} \text{ returns } 1 \mid T \xleftarrow{\mathrm{U}} \mathbb{G}_{p_1p_3}] - \Pr[\mathscr{B} \text{ returns } 1 \mid T \xleftarrow{\mathrm{U}} \mathbb{G}]| \\ &= |\Pr[\mathscr{A} \text{ wins} \mid T \xleftarrow{\mathrm{U}} \mathbb{G}_{p_1P_3}] - \Pr[\mathscr{A} \text{ wins} \mid T \xleftarrow{\mathrm{U}} \mathbb{G}]| \\ &= |\Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{k-1,1}] - \Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{k,0}]| \\ &= |\Pr[\mathcal{E}_{k-1,1}] - \Pr[\mathcal{E}_{k,0}]|\end{aligned}$$

as required. □

Lemma 8.3.3. $|\Pr[\mathcal{E}_{k,0}] - \Pr[\mathcal{E}_{k,1}]| \leq \varepsilon_2$ *for* $1 \leq k \leq \nu$.

The proof is similar to that of Lemma 8.3.2 except for the simulation of the k-key. The end components of this key are additionally rerandomised in $\mathbb{G}_{p_2}$ to ensure that it remains semi-functional with its type depending on whether the instance is real or random.

Proof. Let $(\mathcal{G}_{\mathrm{pub}}, P, P_3, X + P_2, X_2 + X_3, T)$ be the instance of DSG2 that $\mathscr{B}$ has to solve i.e., decide whether $\theta_2 = 0$ or $\theta_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_{p_3}$ where $T = \theta P + \theta_2 P_2 + \theta_3 P_3$.

Setup: Scalars $\alpha, v_{\mathrm{start}}, v_{\mathrm{end}}, \{v_{u,\sigma}, v_{h,\sigma}\}_{\sigma\in\Sigma}$ are chosen from $\mathbb{Z}_N$ independently according to the uniform distribution. Parameters are set as follows: $H_{\mathrm{start}} = v_{\mathrm{start}}P$, $H_{\mathrm{end}} = v_{\mathrm{end}}P$, $H_\sigma = v_{h,\sigma}P$ and $U_\sigma = v_{u,\sigma}P$. $\mathcal{PP}$ is given to $\mathscr{A}$ and $\mathscr{B}$ keeps $\mathcal{MSK}$.

Key extraction queries: For key extraction queries on $\mathcal{M}_i$ for $i \neq k$, $\mathscr{B}$ answers the query as in proof of Lemma 8.3.2. The secret key for $\mathcal{M}_k$ is generated as follows.

For each $x \in \mathbb{Z}_{|Q|}$, $d_x \xleftarrow{\mathrm{U}} \mathbb{Z}_N$
$r'_{\mathrm{start}}, r'_{\mathrm{end}}, \{r'_t\}_{t\in\mathcal{T}}, \mu_1, \mu_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_N$

$K_{\mathrm{start},1} = (d_0 + r'_{\mathrm{start}}v_{\mathrm{start}})T, \quad K_{\mathrm{start},2} = r'_{\mathrm{start}}T,$

For all $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,
$K_{t,1} = (-d_x + r'_t v_{u,\sigma})T, \quad K_{t,2} = r'_t T, \quad K_{t,3} = (d_y + r'_t v_\sigma)T,$

$K_{\mathrm{end},1} = -\alpha P + (d_f + r'_{\mathrm{end}}v_{\mathrm{end}})T + \mu_1(X_2 + X_3), \quad K_{\mathrm{end},2} = r'_{\mathrm{end}}T + \mu_2(X_2 + X_3).$

Let $\mathcal{SK}_{\mathcal{M}} = (K_{\mathrm{start},1}, K_{\mathrm{start},2}, \{K_{t,1}, K_{t,2}, K_{t,3}\}_{t\in\mathcal{T}}, K_{\mathrm{end},1}, K_{\mathrm{end},2})$. $\mathscr{B}$ returns $\mathsf{ReRandK}(\mathcal{SK}_{\mathcal{M}_k})$ to $\mathscr{A}$. If $\theta_2 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_2}$, then $\mathcal{SK}_{\mathcal{M}_k}$ is Type-1 semi-functional; otherwise it is a Type-2 semi-functional key. Both τ_{end} and μ_{end} are set to random quantities in either cases to prevent $\mathscr{B}$ from generating a nominally semi-functional ciphertext to test $\mathcal{SK}_{\mathcal{M}_k}$'s type of semi-functionality. The randomisers for the Type-1 semi-functional components are set as: $\mu_{\mathrm{start}} = r'_{\mathrm{start}}\theta_2$, $\mu_t = r'_t\theta_2$ for all $t \in \mathcal{T}$; $\pi_{\mathrm{start}} = v_{\mathrm{start}}$, $\pi_{u,\sigma} = v_{u,\sigma}$ and $\pi_{h,\sigma} = v_\sigma$ for each $\sigma \in \Sigma$. Furthermore, since the key is rerandomised, its $\mathbb{G}_{p_1}$ and $\mathbb{G}_{p_3}$ components are properly distributed.

The **Challenge** and **Guess** phases are identical to Lemma 8.3.2. If the adversary wins ($\beta \neq \beta'$), then $\mathscr{B}$ returns 1; otherwise it returns 0. Therefore, we have $\varepsilon_2 \geq |\Pr[\mathcal{E}_{k,0}] - \Pr[\mathcal{E}_{k,1}]|$. □

Lemma 8.3.4. $|\Pr[\mathcal{E}_{\nu,1}] - \Pr[\mathcal{E}_{final}]| \leq \varepsilon_3$.

The idea of the proof is as follows. Let $(\mathcal{G}_{\text{pub}}, P, P_2, P_3, \alpha P + X_2, sP + Y_2, T)$ be the instance of SGDH using which the game needs to be simulated. α from the instance is the α of the system master secret. The scalar s from the instance will be mapped to the randomiser that is used to mask the message i.e., $s_{\widehat{\ell}}$, where $\widehat{\ell}$ is the length of the challenge string. Since generators of subgroups corresponding to all three primes are known, (semi-functional) keys and ciphertexts can be generated. The main trick lies in generating the $K_{\text{end},1}$ components of the keys since they have α embedded in them and also in computing the ciphertext terms corresponding to the randomiser $s_{\widehat{\ell}}$.

Proof. Given an instance $(\mathcal{G}_{\text{pub}}, P, P_2, P_3, \alpha P + X_2, sP + Y_2, T)$ of SGDH, $\mathscr{B}$ has to decide whether $T = e(P,P)^{\alpha s}$ or $T \xleftarrow{\text{U}} \mathbb{G}_T$. The game is simulated as follows.

Setup: Randomisers $v_{\text{start}}, v_{\text{end}}, \{v_{u,\sigma}, v_{h,\sigma}\}_{\sigma\in\Sigma}$ are sampled uniformly and independently from $\mathbb{Z}_N$. Then set $H_{\text{start}} = v_{\text{start}}P$, $H_{\text{end}} = v_{\text{end}}P$, for all $\sigma \in \Sigma$, $H_\sigma = v_{h,\sigma}P$, $U_\sigma = v_{u,\sigma}P$ and $e(P,P)^\alpha = e(\alpha P + X_2, P)$. The public parameters $\mathcal{PP}$ are provided to $\mathscr{A}$. Note that the simulator does not know the master secret key.

Key extraction queries: Since αP is masked with an element of $\mathbb{G}_{p_2}$, $\mathscr{B}$ can generate only Type-2 semi-functional keys. For a query on an automaton $\mathcal{M} = (Q, \Sigma, q_0, q_f, \delta)$, a key is constructed as follows. Sample $D_x \xleftarrow{\text{U}} \mathbb{G}_{p_1}$ for all $q_x \in Q$. Construct the components $K_{\text{start},1}, K_{\text{start},2}, \{K_{t,1}, K_{t,2}, K_{t,3}\}_{t\in\mathcal{T}}$ just as in the $\mathcal{B}\text{-}\mathcal{ABE}$.KeyGen algorithm. The master secret α is embedded only the term $K_{\text{end},1}$ and the main trick lies in generating this component. The encoding of α in $\mathbb{G}_{p_1}$ is masked by $\mathbb{G}_{p_2}$-component and hence $\mathscr{B}$ cannot prevent $K_{\text{end},1}$ from having semi-functional components. $\mathscr{B}$ chooses $\mu_{\text{end}}, r_{\text{end}} \xleftarrow{\text{U}} \mathbb{Z}_N$, $R_{\text{end},1}, R_{\text{end},2} \xleftarrow{\text{U}} \mathbb{G}_{p_3}$, $Z_2 \xleftarrow{\text{U}} \mathbb{G}_{p_2}$ and computes

$$K_{\text{end},1} = -(\alpha P + X_2) + D_f + r_{\text{end}}H_{\text{end}} + R_{\text{end},1} + Z_2, \quad K_{\text{end},2} = r_{\text{end}}P + R_{\text{end},2} + \mu_{\text{end}}P_2$$

implicitly setting $\tau_{\text{end}}P_2 = X_2 + Z_2$. Scalars μ_{end} and Z_2 are freshly chosen for each key. Therefore, the values of τ_{end} for the keys remain properly distributed.

Challenge: $\mathscr{B}$ receives two messages m_0, m_1 along with a string $\widehat{w} = \widehat{w}_1 \cdots \widehat{w}_{\widehat{\ell}}$ from $\mathscr{A}$; chooses $\beta \xleftarrow{\text{U}} \{0,1\}$ and constructs a ciphertext for m_β and $\widehat{w}$ as described below.

$s_0, \ldots, s_{\widehat{\ell}-1}, \gamma_0, \ldots, \gamma_{\widehat{\ell}-1} \xleftarrow{\text{U}} \mathbb{Z}_N$;
$\pi_{\text{start}} \xleftarrow{\text{U}} \mathbb{Z}_N$, $\pi_{u,\sigma} \xleftarrow{\text{U}} \mathbb{Z}_N$ for all $\sigma \in \Sigma$;
for all $\sigma \in \Sigma \setminus \{\widehat{w}_{\widehat{\ell}}\}$, $\pi_{h,\sigma} \xleftarrow{\text{U}} \mathbb{Z}_N$, set $\pi_{h,\widehat{w}_{\widehat{\ell}}} = v_{h,\widehat{w}_{\widehat{\ell}}}$,

$C_m = m_\beta \cdot T$,
$C_{0,1} = s_0 P + \gamma_0 P_2, \quad C_{\text{start},2} = s_0 H_{\text{start}} + \gamma_0 \pi_{\text{start}} P_2$,

For $i = 1, \ldots, \widehat{\ell} - 1$,
$\quad C_{i,1} = s_i P + \gamma_i P_2, \quad C_{i,2} = s_i H_{w_i} + s_{i-1} U_\sigma + (\gamma_i \pi_{w_i} + \gamma_{i-1}\pi_{u,w_i})P_2$,
$C_{\widehat{\ell},1} = sP + Y_2, \quad C_{\widehat{\ell},2} = v_{\widehat{w}_{\widehat{\ell}}}(sP + Y_2) + s_{i-1}U_\sigma + \gamma_{i-1}\pi_{u,w_i}P_2$,

$C_{\text{end},1} = C_{\widehat{\ell},1}, \quad C_{\text{end},2} = v_{\text{end}}(sP + Y_2)$.

implicitly setting $s_{\widehat{\ell}} = s$, $\gamma_{\widehat{\ell}} P_2 = Y_2$ and $\pi_{\text{end}} P_2 = v_{\text{end}} Y_2$. The values of $v_{h,\widehat{w}_{\widehat{\ell}}}$ and v_{end} modulo p_2 are hidden from the adversary and hence $\pi_{h,w_{\widehat{\ell}}\widehat{w}_{\widehat{\ell}}}, \pi_{\text{end}}$ are uniformly and independently distributed in $\mathscr{A}$'s view. $\mathscr{B}$ returns $\widehat{\mathcal{C}}$ consisting of the above components to $\mathscr{A}$.

Guess: $\mathscr{A}$ makes its guess β' of β.

If $T = e(P,P)^{\alpha s}$ then we have $C_m = m_\beta \cdot T = m_\beta \cdot e(P,P)^{\alpha s_{\widehat{\ell}}}$ making $\widehat{\mathcal{C}}$ a semi-functional encryption of m_β and thus playing $\mathsf{G}_{\nu,1}$. Otherwise $T \overset{\text{U}}{\longleftarrow} \mathbb{G}_T$ and $(C_m = m_\beta \cdot T) \overset{\text{U}}{\longleftarrow} \mathbb{G}_T$. In this case, $\widehat{\mathcal{C}}$ will be a semi-functional encryption of a random message and $\mathscr{B}$ simulates G_{final}. If the adversary wins the game then $\mathscr{B}$ returns 1; otherwise it returns 0. We therefore have,

$$\begin{aligned}
\varepsilon_3 \geq \mathsf{Adv}_{\mathcal{G}}^{\text{SGDH}}(\mathscr{B}) &= |\Pr[\mathscr{B} \text{ returns } 1 \mid T = e(P,P)^{\alpha s}] - \Pr[\mathscr{B} \text{ returns } 1 \mid T \overset{\text{U}}{\longleftarrow} \mathbb{G}_T]| \\
&= |\Pr[\mathscr{A} \text{ wins} \mid T = e(P,P)^{\alpha s}] - \Pr[\mathscr{A} \text{ wins} \mid T \overset{\text{U}}{\longleftarrow} \mathbb{G}_T]| \\
&= |\Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{\nu,1}] - \Pr[\mathscr{A} \text{ wins in } \mathsf{G}_{final}]| \\
&= |\Pr[\mathcal{E}_{\nu,1}] - \Pr[\mathcal{E}_{final}]|
\end{aligned}$$

as required.

□

8.4 Full Construction

The restrictions on $\mathcal{B}$-$\mathcal{ABE}$ scheme confines the functionality support to a small subclass of regular languages. It is possible to expand the supported class of languages via an extension of $\mathcal{B}$-$\mathcal{ABE}$. The extension provides the ability to deal with multiple occurrences of symbols both in the input string and transitions of the automata. The number of occurrences is however bounded at setup time. As a result, the sizes of public parameters, keys and ciphertexts increase by a factor proportional to these bounds.

We shall first define some notation. For a matrix $\mathbf{A} \in \mathbb{Z}_N{}^{m \times n}$, $\mathbf{A}[i,j]$ denotes the entry in i-th row and j-column of $\mathbf{A}$. Let $w = w_1 \ldots w_\ell$ be a string over the alphabet Σ and $\mathcal{T}$ be the (ordered) set of transitions of an automaton $\mathcal{M}$.

- $\mathsf{s_{max}}$: bound on the number of occurrences of each symbol in a string
- $\mathsf{t_{max}}$: the maximum number of transitions on any particular symbol
- $\mathsf{n^c}[w,i]$: contains k if position i is the k-occurrence of the symbol w_i in w
- $\mathsf{n^k}[\sigma,t]$: contains k if t is the k-transition on σ

The extended construction $\mathcal{F}$-$\mathcal{ABE}$ = ($\mathcal{F}$-$\mathcal{ABE}$.Setup, $\mathcal{F}$-$\mathcal{ABE}$.Encrypt, $\mathcal{F}$-$\mathcal{ABE}$.KeyGen, $\mathcal{F}$-$\mathcal{ABE}$.Decrypt) is described below.

$\mathcal{F}$-$\mathcal{ABE}$.Setup(Σ, κ): Generate a composite order pairing $\mathcal{G} = (p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, e, G)$ according to the security parameter κ. Choose elements $P, H_{\text{start}}, H_{\text{end}} \overset{\text{U}}{\longleftarrow} \mathbb{G}_{p_1}$, $P_3 \overset{\text{U}}{\longleftarrow} \mathbb{G}_{p_3}$, $\alpha \overset{\text{U}}{\longleftarrow} \mathbb{Z}_N$ and

$$\mathbf{H}_\sigma, \mathbf{U}_\sigma \overset{\text{U}}{\longleftarrow} (\mathbb{Z}_N)^{\mathsf{s_{max}} \times \mathsf{t_{max}}} \text{ for all } \sigma \in \Sigma.$$

The public parameters and master secret are given by

$\mathcal{PP}$: $(\mathcal{G}_{\text{pub}}, \Sigma, P, H_{\text{start}}, H_{\text{end}}, H_\lambda, (\mathbf{H}_\sigma, \mathbf{U}_\sigma)_{\sigma\in\Sigma}, e(P,P)^\alpha)$,
$\mathcal{MSK}$: $(-\alpha P, P_3)$.

$\mathcal{F}$-$\mathcal{ABE}$.KeyGen$(\mathcal{MSK}, \mathcal{M} = (Q, \Sigma, q_0, q_f, \delta))$: For each $x \in \mathbb{Z}_{|Q|}$, pick $D_x \xleftarrow{\text{U}} \mathbb{G}_{p_1}$. Choose elements r_{start}, for all $t \in \mathcal{T}$, r_t and r_{end} uniformly and independently at random from $\mathbb{Z}_N$. Let $R_{\text{start},1}, R_{\text{start},2}$, $(R_{t,1}, R_{t,2}, R_{t,3})_{t\in\mathcal{T}}$ and $R_{\text{end},1}, R_{\text{end},2}$ be randomly chosen elements of $\mathbb{G}_{p_3}$. Compute the elements of the key as follows.

$$K_{\text{start},1} = D_0 + r_{\text{start}}H_{\text{start}} + R_{\text{start},1}, \quad K_{\text{start},2} = r_{\text{start}}P + R_{\text{start},2},$$

For all $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,

$$K_{t,2} = r_tP + R_{t,2},$$
$$(K_{t,1,i} = -D_x + r_t\mathbf{U}_\sigma[i, \mathsf{n}^\mathsf{k}[\sigma,t]] + R_{t,1}, \quad K_{t,3,i} = D_y + r_t\mathbf{H}_\sigma[i,\mathsf{n}^\mathsf{k}[\sigma,t]] + R_{t,3})_{i\in[1,\mathsf{s}_{\max}]},$$

$$K_{\text{end},1} = -\alpha P + D_f + r_{\text{end}}H_{\text{end}} + R_{\text{end},1}, \quad K_{\text{end},2} = r_{\text{end}}P + R_{\text{end},2}.$$

Here D_f corresponds to the final state q_f. The secret key for automaton $\mathcal{M}$ is given by $\mathcal{SK}_\mathcal{M}$ = $(K_{\text{start},1}, K_{\text{start},2}, (K_{t,1}, K_{t,2}, K_{t,3})_{t\in\mathcal{T}}, K_{\text{end},1}, K_{\text{end},2})$.

$\mathcal{F}$-$\mathcal{ABE}$.Encrypt$(\mathcal{PP}, w = w_1\cdots w_\ell, m)$: Choose randomisers $s_0, s_1, \ldots, s_\ell \xleftarrow{\text{U}} \mathbb{Z}_N$. Compute the ciphertext elements as follows.

$$C_m = m\cdot e(P,P)^{\alpha s_\ell},$$

$$C_{0,0} = C_{\text{start},1} = s_0P, \quad C_{\text{start},2} = s_0H_{\text{start}},$$

For $i = 1,\ldots,\ell$,

$$C_{i,0} = s_iP, \quad (C_{i,j} = s_i\mathbf{H}_{w_i}[\mathsf{n}^\mathsf{c}[w,i],j] + s_{i-1}\mathbf{U}_{w_i}[\mathsf{n}^\mathsf{c}[w,i],j])_{j\in[1,\mathsf{t}_{\max}]},$$

$$C_{\text{end},1} = C_{\ell,0} = s_\ell P, \quad C_{\text{end},2} = s_\ell H_{\text{end}}.$$

The ciphertext is given by $\mathcal{C} = (C_m, C_{\text{start},1}, C_{\text{start},2}, (C_{i,0}, C_{i,j})_{i\in[1,\ell],j\in[1,\mathsf{t}_{\max}]}, C_{\text{end},1}, C_{\text{end},2}, w)$.

$\mathcal{F}$-$\mathcal{ABE}$.Decrypt$(\mathcal{C}, \mathcal{SK}_\mathcal{M})$: Suppose that Accept$(\mathcal{M}, w) = 1$ and $w = w_1\cdots w_\ell$. Then there exists a sequence of transitions $t_1, t_2, \ldots, t_\ell$ with $t_i = (q_{x_{i-1}}, q_{x_i}, w_i)$ where $x_0 = 0$ and $x_\ell = f$. Decryption consists of several stages of computation. First compute

$$\begin{aligned} A_0 &= e(C_{\text{start},1}, K_{\text{start},1})e(C_{\text{start},1}, K_{\text{start},2})^{-1} \\ &= e(P, D_0)^{s_0} \end{aligned}$$

Then compute intermediate values A_i (for $i = 1,\ldots,\ell$) as follows. Pick $C_{i,\mathsf{n}^\mathsf{k}[w_i,t_i]}$ and $K_{t_i,1,\mathsf{n}^\mathsf{c}[w_i,i]}, K_{t_i,3,\mathsf{n}^\mathsf{c}[w_i,i]}$. Such components exist and are unique.

$$\begin{aligned} A_i &= A_{i-1}\cdot e(C_{i-1,0}, K_{t_i,1,\mathsf{n}^\mathsf{c}[w_i,i]})e(C_{i,\mathsf{n}^\mathsf{k}[w_i,t_i]}, K_{t_i,2})^{-1}e(C_{i,0}, K_{t_i,3,\mathsf{n}^\mathsf{c}[w_i,i]}) \\ &= e(P, D_{x_i})^{s_i} \end{aligned}$$

With any other pair of $C_{i,j}$ and $K_{t_i,1,k}, K_{t_i,3,k}$ it is not possible to cancel out $e(P, D_{x_{i-1}})^{s_{i-1}}$. The last intermediate $A_{\ell+1}$ is computed as

$$A_{\ell+1} = e(C_{\text{end},1}, K_{\text{end},1})\cdot e(C_{\text{end},2}, K_{\text{end},2})^{-1} = e(P,P)^{-\alpha s_\ell}e(D_f, P)^{s_\ell}.$$

Using A_ℓ and $A_{\ell+1}$ the message is unmasked as shown below.

$$m = C_m \cdot A_{\ell+1} \cdot A_\ell^{-1}.$$

Discussion. The construction essentially converts a DFA and string to a basic form by mapping each occurrence of a symbol σ to a different representation in the group. Consider a ciphertext for string w and automaton $\mathcal{M}$. In the full ABE scheme, w and $\mathcal{M}$ are encoded so that there exists a unique sequence of decryption operations that result in the correct message if $\mathcal{M}$ accepts w. Given this, correctness of decryption follows.

For example, consider $\Sigma = \{0, 1\}$. Public parameters would then consist of matrices $\mathbf{H}_0[\mathsf{s}_{\mathsf{max}}, \mathsf{t}_{\mathsf{max}}]$ and $\mathbf{H}_1[\mathsf{s}_{\mathsf{max}}, \mathsf{t}_{\mathsf{max}}]$. Suppose encryption is to be done for the string $10100 \in \{0, 1\}^*$. This string is encoded as $1_1 0_1 1_2 0_2 0_3$ so that each occurrence of 0 (or 1) is treated separately. 1_1 is mapped to $\mathbf{H}_1[1, \cdot]$ and 1_2 to $\mathbf{H}_1[2, \cdot]$. Similarly, each occurence of 0 can be mapped to a unique row in $\mathbf{H}_0$. For the key, let $\mathcal{M}$ be an automaton over $\{0, 1\}$. Suppose there are two transitions $t_1 = (q_{x_1}, 1, q_{y_1})$ and $t_2 = (q_{x_2}, 1, q_{y_2})$ appearing in the same order without any other 1-transition in between. Then the group elements used to encode t_1 will correspond to a unique column in $\mathbf{H}_1$ given by $\mathbf{H}_1[\cdot, 1]$. Transition t_2 would correspond to the column $\mathbf{H}_1[\cdot, 2]$. Suppose symbol 1_1 triggers the transition t_2. Decryption is done using the ciphertext and key component pair corresponding to the (unique) entry $\mathbf{H}[1, 2]$ of $\mathbf{H}$ present in both the ciphertext portion for the symbol 1_1 as well as the key portion for transition t_2.

Suppose there were only one public parameter corresponding to σ, say H_σ. Then in the second reduction, H_σ would be computed as $v_\sigma P$ where $P \in \mathbb{G}^\times$ and $v_\sigma \xleftarrow{\text{U}} \mathbb{Z}_N$. The simulator must also be able to create a properly distributed semi-functional ciphertext. For a single occurence of σ in the challenge string $\widehat{w}_1$, the value of $v_\sigma \pmod{p_2}$ can be used to create the semi-functional term in the ciphertext corresponding to σ. But for more than one occurence $v_\sigma \pmod{p_2}$ does not provide sufficient entropy to argue about the independence of semi-functional terms corresponding to all the occurrences of σ. Therefore, while arguing about security, the existence of $\mathsf{s}_{\mathsf{max}} \times \mathsf{t}_{\mathsf{max}}$ distinct representations for a symbol σ in the public parameters ensures that the semi-functional components for all occurrences of σ are independent of each other. Furthermore, the same rerandomisation technique can be employed to ensure proper distribution of keys and ciphertexts in the proof.

Stated formally below is the security guarantee we obtain for $\mathcal{F}$-$\mathcal{ABE}$.

Theorem 8.4.1. *If the* (ε_1, t')-DSG1, (ε_2, t')-DSG2, (ε_3, t')-SGDH *assumptions hold, then* $\mathcal{F}$-$\mathcal{ABE}$ *is* (ε, t, ν)-IND-STR-CPA *secure where*

$$\varepsilon \le \varepsilon_1 + 2\nu\varepsilon_2 + \varepsilon_3$$

and $t = t' - O(\nu|\Sigma|\rho \cdot \max(\mathsf{s}_{\mathsf{max}}, \mathsf{t}_{\mathsf{max}}))$, *where* ρ *is an upper bound on the time required for one scalar multiplication in* $\mathbb{G}$.

Chapter 9

Conclusion

This thesis contains several practical contributions to identity-based cryptography.

We started with converting Waters dual system IBE scheme from the setting of symmetric pairings to that of asymmetric pairings. This has been done in a systematic manner going through several stages of simplifications. The simplification resulted in two IBE schemes ($\mathcal{IBE}1$ and $\mathcal{IBE}6$). Security of $\mathcal{IBE}1$ is based on standard assumptions and reduces the sizes of ciphertexts and keys by 2 elements each from the original scheme of Waters. $\mathcal{IBE}6$ is quite simple and minimal in the sense that both encryption and key generation use one randomiser each. The security of $\mathcal{IBE}6$ is based on two standard assumptions and a natural and minimal extension of the DDH assumption for $\mathbb{G}_2$.

We then considered constant size ciphertext HIBE. The first construction $\mathcal{LW}$-$\mathcal{AHIBE}$ was obtained by extending the Lewko-Waters IBE scheme using asymmetric pairings to a constant-size ciphertext HIBE. In addition to CPA-security the HIBE scheme possesses anonymity. Security is based on the assumptions LW1, LW2, DBDH-3 and a new assumption A1 that we introduce. The assumptions used are static but non-standard. $\mathcal{LW}$-$\mathcal{AHIBE}$ is the first example of an anonymous, adaptively secure, constant-size ciphertext HIBE which can be instantiated using Type-3 pairings. We went on to study the problem of obtaining more efficient constant size ciphertext HIBE scheme based on standard assumption. Starting from the IBE scheme of Jutla and Roy, we obtain two HIBE schemes with constant-size ciphertexts and full security. One achieves anonymity while the other is non-anonymous with shorter keys. Compared to previous HIBE schemes, our constructions provide very good efficiency with just 3 pairings for decryption and 3 group elements in the ciphertext. At the time they were proposed, these were the only CC-HIBEs achieving security under standard assumptions and degradation independent of the HIBE depth. In HIBE-related literature focused on either constant-size ciphertexts or anonymity or both, we believe that our constructions completed the picture.

Another primitive we explored is identity-based broadcast encryption. We presented new IBBE schemes which achieve both theoretically satisfying security (i.e, security against adaptive-identity attacks based on simple assumptions) and practical efficiency at the same time. The new schemes are obtained by developing on the currently known most efficient IBE scheme due to Jutla and Roy [103].

Finally, as an application of dual system techniques, we obtained an attribute-based encryption scheme for DFAs based on composite order pairings that has adaptive security under static assumptions. This was obtained by adopting Waters' DFA-based ABE [160] in the composite order setting. Certain restrictions were imposed on the DFAs in order to obtain a dual system proof. This limited the set of regular languages that can be supported by the ABE scheme. We showed how to relax the restrictions and thus support a larger sub-class of regular languages. The cost of achieving this is an increase in the sizes of the ciphertext and keys.

Open Problems and Future Directions. We point out some interesting problems to pursue for further research. As with most prior work, the new schemes are proved secure against chosen-plaintext attacks. It is of interest to obtain efficient variants of our schemes which are secure against chosen-ciphertext attacks. Also, actual implementation studies will take some of the works further along the path of actual deployment.

Recent works have shown how to construct (hierarchical) identity-based encryption schemes with a tight security reductions based on simple assumptions. An important future direction is to obtain tightly secure anonymous HIBE. Further, the IBE schemes proposed by these works have public parameters of the size $O(n)$ where n is the bit-length of identities. The problem of obtaining a tightly secure IBE based on simple assumptions with constant size public parameters remains wide open. Another important future direction is to obtain an anonymous HIBE with prefix decryption. As we noted in Chapter 6, it seems impossible to build a HIBE scheme with all three of the following properties – anonymity, prefix decryption and constant-size ciphertext. It would be interesting to explore this further and either formally prove that this is indeed the case or find a counter-example.

We proposed an IBBE scheme with $O(m)$-size public parameters (m is the maximum number of privileged users), ciphertexts of length $O(|S|)$ (where S is the set of intended recipients) and constant sized secret keys. An important open problem is to build an IBBE scheme with constant-size ciphertexts and keys and adaptive security under simple assumptions. Let $\#pp$, $\#cpr$ and $\#key$ denote the sizes of public parameters, ciphertexts and keys. Some interesting questions related to PKBE schemes are as follows.

- Is it possible to construct an adaptively secure PKBE scheme under simple assumptions with $\#pp = O(n)$, $\#cpr = O(1)$ and $\#key = O(1)$ (where n is the total number of users)?
- Do BE schemes with $\#pp = O(1)$, $\#cpr = O(|S|)$ and $\#key = O(1)$ and adaptive security from simple assumptions exist?

The most interesting problem is to construct an adaptively secure (IB)BE scheme having $O(\mathsf{poly}(\log n))$ sized public parameters, ciphertexts and keys from pairings. (Here, poly denotes some polynomial and n is the total number of users).

With reference to DFA-based ABE schemes, it would be interesting to obtain adaptive security without restricting the number of occurrences of symbols in either the strings or transitions of automata based on static assumptions. An important future direction is to build ABE schemes supporting more sophisticated functionalities.

Many of the above-mentioned problems may not have solutions based on bilinear maps. They might require richer structures such as multilinear maps. On the other hand, it is of immense interest to know how much can be achieved with pairings.

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 Shoup [150], pages 205–222.

[2] Michel Abdalla, Eike Kiltz, and Gregory Neven. Generalized key delegation for hierarchical identity-based encryption. In Joachim Biskup and Javier Lopez, editors, *ESORICS*, volume 4734 of *Lecture Notes in Computer Science*, pages 139–154. Springer, 2007.

[3] Gora Adj, Alfred Menezes, Thomaz Oliveira, and Francisco Rodríguez-Henríquez. Computing Discrete Logarithms in $\mathbb{F}_{3^{6*137}}$ and $\mathbb{F}_{3^{6*163}}$ using Magma. Cryptology ePrint Archive, Report 2014/057, 2014. `http://eprint.iacr.org/`.

[4] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Gilbert [88], pages 553–572.

[5] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In Rabin [134], pages 98–115.

[6] Sattam S. Al-Riyami and Kenneth G. Paterson. Certificateless public key cryptography. In Chi-Sung Laih, editor, *Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30 - December 4, 2003, Proceedings*, volume 2894 of *Lecture Notes in Computer Science*, pages 452–473. Springer, 2003.

[7] Sattam S. Al-Riyami and Kenneth G. Paterson. CBE from CL-PKE: A generic construction and efficient schemes. In Vaudenay [154], pages 398–415.

[8] Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, volume 8441 of *Lecture Notes in Computer Science*, pages 557–577. Springer, 2014.

[9] Nuttapong Attrapadung. Dual system encryption framework in prime-order groups. Cryptology ePrint Archive, Report 2015/390, 2015. `http://eprint.iacr.org/`.

[10] Joonsang Baek, Reihaneh Safavi-Naini, and Willy Susilo. Efficient multi-receiver identity-based encryption and its application to broadcast encryption. In Vaudenay [154], pages 380–397.

[11] M. Barbosa and P. Farshim. Efficient identity-based key encapsulation to multiple parties. In Nigel P. Smart, editor, *IMA Int. Conf.*, volume 3796 of *Lecture Notes in Computer Science*, pages 428–441. Springer, 2005.

[12] Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé. A quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. *CoRR*, abs/1306.4244, 2013.

[13] Paulo S. L. M. Barreto, Benoît Libert, Noel McCullagh, and Jean-Jacques Quisquater. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In Bimal Roy, editor, *ASIACRYPT*, volume 3788 of *Lecture Notes in Computer Science*, pages 515–532. Springer, 2005.

[14] Mihir Bellare, Alexandra Boldyreva, and Adriana Palacio. An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In Cachin and Camenisch [37], pages 171–188.

[15] Mihir Bellare, Eike Kiltz, Chris Peikert, and Brent Waters. Identity-based (lossy) trapdoor functions and applications. In Pointcheval and Johansson [133], pages 228–245.

[16] Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Security proofs for identity-based identification and signature schemes. *J. Cryptology*, 22(1):1–61, 2009.

[17] Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for waters' IBE scheme. In Joux [100], pages 407–424.

[18] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[19] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-Policy Attribute-Based Encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334. IEEE Computer Society, 2007.

[20] Eli Biham, editor. *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*. Springer, 2003.

[21] Olivier Blazy, Eike Kiltz, and Jiaxin Pan. (hierarchical) identity-based encryption from affine message authentication. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO (1)*, volume 8616 of *Lecture Notes in Computer Science*, pages 408–425. Springer, 2014.

[22] Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity-based encryption without random oracles. In Cachin and Camenisch [37], pages 223–238.

[23] Dan Boneh and Xavier Boyen. Secure identity-based encryption without random oracles. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 443–459. Springer, 2004.

[24] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity-based encryption with constant size ciphertext. In Cramer [60], pages 440–456. Full version available at Cryptology ePrint Archive; Report 2005/015.

[25] 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.

[26] Dan Boneh, Craig Gentry, and Michael Hamburg. Space-efficient identity based encryption without pairings. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*, pages 647–657. IEEE Computer Society, 2007.

[27] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Biham [20], pages 416–432.

[28] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Shoup [150], pages 258–275.

[29] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF formulas on ciphertexts. In Joe Kilian, editor, *TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 325–341. Springer, 2005.

[30] Dan Boneh and Jonathan Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 87–103. Springer, 2005.

[31] 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.

[32] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: a new vision for public-key cryptography. *Commun. ACM*, 55(11):56–64, 2012.

[33] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *IACR Cryptology ePrint Archive*, 2002:80, 2002.

[34] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 280–300. Springer, 2013.

[35] 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.

[36] Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 290–307. Springer, 2006.

[37] 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.

[38] Ran Canetti and Juan A. Garay, editors. *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*. Springer, 2013.

[39] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004.

[40] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In Biham [20], pages 255–271.

[41] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In Cachin and Camenisch [37], pages 207–222.

[42] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. *J. Cryptology*, 25(4):601–639, 2012.

[43] Sanjit Chatterjee, Darrel Hankerson, Edward Knapp, and Alfred Menezes. Comparing two pairing-based aggregate signature schemes. *Des. Codes Cryptography*, 55(2-3):141–167, 2010.

[44] Sanjit Chatterjee and Alfred Menezes. On cryptographic protocols employing asymmetric pairings – the role of ψ revisited. *Discrete Applied Mathematics*, 159(13):1311–1322, 2011.

[45] 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.

[46] 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.

[47] Sanjit Chatterjee and Palash Sarkar. HIBE with short public parameters without random oracle. In X. Lai and K. Chen, editors, *ASIACRYPT*, volume 4284 of *Lecture Notes in Computer Science*, pages 145–160. Springer, 2006. see also Cryptology ePrint Archive, Report 2006/279, http://eprint.iacr.org/.

[48] Sanjit Chatterjee and Palash Sarkar. Multi-receiver identity-based key encapsulation with shortened ciphertext. In Rana Barua and Tanja Lange, editors, *INDOCRYPT*, volume 4329 of *Lecture Notes in Computer Science*, pages 394–408. Springer, 2006.

[49] Sanjit Chatterjee and Palash Sarkar. New constructions of constant size ciphertext HIBE without random oracle. In M.S. Rhee and B. Lee, editors, *ICISC*, volume 4296 of *Lecture Notes in Computer Science*, pages 310–327. Springer, 2006.

[50] Sanjit Chatterjee and Palash Sarkar. Constant size ciphertext HIBE in the augmented selective-id model and its extensions. *J. UCS*, 13(10):1367–1395, 2007.

[51] Sanjit Chatterjee and Palash Sarkar. *Identity-based encryption.* Springer, 2011.

[52] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Hoeteck Wee. Shorter IBE and signatures via asymmetric pairings. *IACR Cryptology ePrint Archive*, 2012:224, 2012.

[53] Jie Chen and Hoeteck Wee. Fully, (almost) tightly secure IBE and dual system groups. In Canetti and Garay [38], pages 435–460. Full version available as IACR Technical Report, 2013/803, `http://eprint.iacr.org/2013/803`.

[54] Jie Chen and Hoeteck Wee. Fully, (almost) tightly secure IBE and dual system groups. `https://sites.google.com/site/jchencrypto/publications`, 2013.

[55] Jie Chen and Hoeteck Wee. Dual system groups and its applications — compact hibe and more. Cryptology ePrint Archive, Report 2014/265, 2014. `http://eprint.iacr.org/`.

[56] Liqun Chen, Zhaohui Cheng, and Nigel P. Smart. Identity-based key agreement protocols from pairings. *Int. J. Inf. Sec.*, 6(4):213–241, 2007.

[57] Jung Hee Cheon. Security Analysis of the Strong Diffie-Hellman Problem. In Vaudenay [155], pages 1–11.

[58] Sherman S. M. Chow. Removing Escrow from Identity-Based Encryption. In Jarecki and Tsudik [98], pages 256–276.

[59] 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.

[60] 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.

[61] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Knudsen [106], pages 45–64.

[62] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Comput.*, 33(1):167–226, 2003.

[63] Angelo De Caro, Vincenzo Iovino, and Giuseppe Persiano. Fully secure anonymous HIBE and secret-key anonymous IBE with short ciphertexts. In Marc Joye, Atsuko Miyaji, and Akira Otsuka, editors, *Pairing-Based Cryptography - Pairing 2010*, volume 6487 of *Lecture Notes in Computer Science*, pages 347–366. Springer Berlin / Heidelberg, 2010.

[64] Cécile Delerablée. Identity-based broadcast encryption with constant size ciphertexts and private keys. In Kaoru Kurosawa, editor, *Advances in Cryptology - ASIACRYPT 2007, 13th International Conference on the Theory and Application of Cryptology and Information Security, Kuching, Malaysia, December 2-6, 2007, Proceedings*, volume 4833 of *Lecture Notes in Computer Science*, pages 200–215. Springer, 2007.

[65] Cécile Delerablée, Pascal Paillier, and David Pointcheval. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In Tsuyoshi Takagi, Tatsuaki Okamoto, Eiji Okamoto, and Takeshi Okamoto, editors, *Pairing*, volume 4575 of *Lecture Notes in Computer Science*, pages 39–59. Springer, 2007.

[66] Alexander W. Dent. A survey of certificateless encryption schemes and security models. *Int. J. Inf. Sec.*, 7(5):349–377, 2008.

[67] Alexander W. Dent, Benoît Libert, and Kenneth G. Paterson. Certificateless encryption schemes strongly secure in the standard model. In Ronald Cramer, editor, *Public Key Cryptography - PKC 2008, 11th International Workshop on Practice and Theory in Public-Key Cryptography, Barcelona, Spain, March 9-12, 2008. Proceedings*, volume 4939 of *Lecture Notes in Computer Science*, pages 344–359. Springer, 2008.

[68] 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.

[69] Yevgeniy Dodis and Nelly Fazio. Public key trace and revoke scheme secure against adaptive chosen ciphertext attack. In Yvo Desmedt, editor, *Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, pages 100–115. Springer, 2003.

[70] Léo Ducas. Anonymity from asymmetry: New constructions for anonymous HIBE. In Josef Pieprzyk, editor, *CT-RSA*, volume 5985 of *Lecture Notes in Computer Science*, pages 148–164. Springer, 2010.

[71] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge L. Villar. An algebraic framework for diffie-hellman assumptions. In Canetti and Garay [38], pages 129–147.

[72] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *CRYPTO*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491. Springer, 1993.

[73] David Mandell Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In Gilbert [88], pages 44–61.

[74] 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.

[75] Steven Galbraith. New discrete logarithm records, and the death of Type 1 pairings. `http://ellipticnews.wordpress.com/2014/02/01/new-discrete-logarithm-records-and-the-death-of-type-1-pairings/#comment-426`, 2014.

[76] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

[77] Steven D. Galbraith and Victor Rotger. Easy decision-Diffie-Hellman groups. *IACR Cryptology ePrint Archive*, 2004:70, 2004.

[78] 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.

[79] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2013.

[80] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 40–49. IEEE Computer Society, 2013.

[81] Sanjam Garg, Craig Gentry, Shai Halevi, Amit Sahai, and Brent Waters. Attribute-based encryption for circuits from multilinear maps. In Canetti and Garay [38], pages 479–499.

[82] Craig Gentry. Certificate-based encryption and the certificate revocation problem. In Biham [20], pages 272–293.

[83] Craig Gentry. Practical identity-based encryption without random oracles. In Vaudenay [155], pages 445–464.

[84] Craig Gentry and Shai Halevi. Hierarchical Identity Based Encryption with Polynomially Many Levels. In Omer Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 437–456. Springer, 2009.

[85] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 197–206. ACM, 2008.

[86] 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.

[87] Craig Gentry and Brent Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In Joux [100], pages 171–188.

[88] Henri Gilbert, editor. *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*. Springer, 2010.

[89] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[90] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 545–554. ACM, 2013.

[91] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security*, pages 89–98. ACM, 2006.

[92] Robert Granger, Thorsten Kleinjung, and Jens Zumbrägel. Breaking '128-bit secure' supersingular binary curves (or how to solve discrete logarithms in $\mathbb{F}_{2^{4\cdot1223}}$ and $\mathbb{F}_{2^{12\cdot367}}$). Cryptology ePrint Archive, Report 2014/119, 2014. `http://eprint.iacr.org/`.

[93] Robert Granger, Thorsten Kleinjung, and Jens Zumbragel. Discrete logarithms in $GF(2^9 234)$. `https://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind1401&L=NMBRTHRY&F=&S=&P=8736`, 2014.

[94] 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.

[95] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison Wesley, 2 edition, 2000.

[96] Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In Knudsen [106], pages 466–481.

[97] Qiong Huang and Duncan S. Wong. Generic certificateless encryption in the standard model. In Atsuko Miyaji, Hiroaki Kikuchi, and Kai Rannenberg, editors, *Advances in Information and Computer Security, Second International Workshop on Security, IWSEC 2007, Nara, Japan, October 29-31, 2007, Proceedings*, volume 4752 of *Lecture Notes in Computer Science*, pages 278–291. Springer, 2007.

[98] Stanislaw Jarecki and Gene Tsudik, editors. *Public Key Cryptography - PKC 2009, 12th International Conference on Practice and Theory in Public Key Cryptography, Irvine, CA, USA, March 18-20, 2009. Proceedings*, volume 5443 of *Lecture Notes in Computer Science.* Springer, 2009.

[99] 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.

[100] Antoine Joux, editor. *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, volume 5479 of *Lecture Notes in Computer Science.* Springer, 2009.

[101] Antoine Joux. A new index calculus algorithm with complexity $L(1/4 + o(1))$ in small characteristic. In Tanja Lange, Kristin Lauter, and Petr Lisonek, editors, *Selected Areas in Cryptography*, volume 8282 of *Lecture Notes in Computer Science*, pages 355–379. Springer, 2013.

[102] Marc Joye and Gregory Neven. *Identity-based cryptography*, volume 2. IOS press, 2009.

[103] Charanjit S. Jutla and Arnab Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT (1)*, volume 8269 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2013.

[104] Charanjit S. Jutla and Arnab Roy. Switching lemma for bilinear tests and constant-size NIZK proofs for linear subspaces. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II*, volume 8617 of *Lecture Notes in Computer Science*, pages 295–312. Springer, 2014.

[105] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, 2008.

[106] Lars R. Knudsen, editor. *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, volume 2332 of *Lecture Notes in Computer Science*. Springer, 2002.

[107] Kwangsu Lee, JongHwan Park, and DongHoon Lee. Anonymous HIBE with short ciphertexts: full security in prime order groups. *Designs, Codes and Cryptography*, pages 1–31, 2013.

[108] Kwangsu Lee, JongHwan Park, and DongHoon Lee. Anonymous HIBE with short ciphertexts: full security in prime order groups. *Designs, Codes and Cryptography*, pages 1–31, 2013.

[109] Allison Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. Cryptology ePrint Archive, Report 2011/490, 2011. `http://eprint.iacr.org/`.

[110] Allison Lewko and Brent Waters. New Proof Methods for Attribute-Based Encryption: Achieving Full Security through Selective Techniques. In Safavi-Naini and Canetti [140], pages 180–198.

[111] Allison B. Lewko. Tools for Simulating Features of Composite Order Bilinear Groups in the Prime Order Setting. In Pointcheval and Johansson [133], pages 318–335.

[112] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In Gilbert [88], pages 62–91.

[113] Allison B. Lewko, Amit Sahai, and Brent Waters. Revocation Systems with Very Small Private Keys. In *IEEE Symposium on Security and Privacy*, pages 273–285. IEEE Computer Society, 2010.

[114] Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *TCC*, volume 5978 of *Lecture Notes in Computer Science*, pages 455–479. Springer, 2010.

[115] Joseph K. Liu, Man Ho Au, and Willy Susilo. Self-generated-certificate public key cryptography and certificateless signature / encryption scheme in the standard model. *IACR Cryptology ePrint Archive*, 2006:373, 2006.

[116] David Naccache. Secure and practical identity-based encryption. *IET Information Security*, 1(2):59–64, 2007.

[117] 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.

[118] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004.

[119] Tatsuaki Okamoto and Katsuyuki Takashima. Homomorphic Encryption and Signatures from Vector Decomposition. In Steven D. Galbraith and Kenneth G. Paterson, editors, *Pairing*, volume 5209 of *Lecture Notes in Computer Science*, pages 57–74. Springer, 2008.

[120] Tatsuaki Okamoto and Katsuyuki Takashima. Hierarchical Predicate Encryption for Inner-Products. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 214–231. Springer, 2009.

[121] Tatsuaki Okamoto and Katsuyuki Takashima. Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In Rabin [134], pages 191–208.

[122] Tatsuaki Okamoto and Katsuyuki Takashima. Achieving Short Ciphertexts or Short Secret-Keys for Adaptively Secure General Inner-Product Encryption. In Dongdai Lin, Gene Tsudik, and Xiaoyun Wang, editors, *CANS*, volume 7092 of *Lecture Notes in Computer Science*, pages 138–159. Springer, 2011.

[123] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner-product encryption. Cryptology ePrint Archive, Report 2011/543, 2011. `http://eprint.iacr.org/`.

[124] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively Attribute-Hiding (Hierarchical) Inner Product Encryption. In Pointcheval and Johansson [133], pages 591–608.

[125] Tatsuaki Okamoto and Katsuyuki Takashima. Fully Secure Unbounded Inner-Product and Attribute-Based Encryption. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT*, volume 7658 of *Lecture Notes in Computer Science*, pages 349–366. Springer, 2012.

[126] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *ACM Conference on Computer and Communications Security*, pages 195–203. ACM, 2007.

[127] Tapas Pandit and Rana Barua. Adaptively Secure Functional Encryption for Finite Languages from DLIN Assumption. Cryptology ePrint Archive, Report 2014/225, 2014. `http://eprint.iacr.org/`.

[128] Jong Hwan Park, Ki Tak Kim, and Dong Hoon Lee. Cryptanalysis and improvement of a multi-receiver identity-based key encapsulation at INDOCRYPT 06. In Masayuki Abe and Virgil D. Gligor, editors, *ASIACCS*, pages 373–380. ACM, 2008.

[129] Jong Hwan Park and Dong Hoon Lee. Anonymous HIBE: Compact construction over prime-order groups. *IEEE Transactions on Information Theory*, 59(4):2531–2541, 2013.

[130] Kenneth G. Paterson and Sriramkrishnan Srinivasan. On the relations between non-interactive key distribution, identity-based encryption and trapdoor discrete log groups. *Des. Codes Cryptography*, 52(2):219–241, 2009.

[131] Duong Hieu Phan, David Pointcheval, Siamak Fayyaz Shahandashti, and Mario Strefler. Adaptive CCA Broadcast Encryption with Constant-Size Secret Keys and Ciphertexts. In Willy Susilo, Yi Mu, and Jennifer Seberry, editors, *ACISP*, volume 7372 of *Lecture Notes in Computer Science*, pages 308–321. Springer, 2012.

[132] Duong Hieu Phan, David Pointcheval, and Mario Strefler. Adaptively secure broadcast encryption with forward secrecy. *IACR Cryptology ePrint Archive*, 2011:463, 2011.

[133] David Pointcheval and Thomas Johansson, editors. *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*. Springer, 2012.

[134] Tal Rabin, editor. *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*. Springer, 2010.

[135] Somindu C. Ramanna. Dfa-based functional encryption: Adaptive security from dual system encryption. Cryptology ePrint Archive, Report 2013/638, 2013. `http://eprint.iacr.org/`.

[136] Somindu C. Ramanna, Sanjit Chatterjee, and Palash Sarkar. Variants of waters' dual system primitives using asymmetric pairings - (extended abstract). In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 298–315. Springer, 2012.

[137] Somindu C. Ramanna and Palash Sarkar. Anonymous constant-size ciphertext HIBE from asymmetric pairings. In Martijn Stam, editor, *IMA Int. Conf.*, volume 8308 of *Lecture Notes in Computer Science*, pages 344–363. Springer, 2013.

[138] Somindu C. Ramanna and Palash Sarkar. Efficient Adaptively Secure IBBE from Standard Assumptions. Cryptology ePrint Archive, Report 2014/380, 2014. `http://eprint.iacr.org/`.

[139] Somindu C. Ramanna and Palash Sarkar. Efficient (anonymous) compact HIBE from standard assumptions. In Sherman S. M. Chow, Joseph K. Liu, Lucas Chi Kwong Hui, and Siu-Ming Yiu, editors, *Provable Security - 8th International Conference, ProvSec 2014, Hong Kong, China, October 9-10, 2014. Proceedings*, volume 8782 of *Lecture Notes in Computer Science*, pages 243–258. Springer, 2014.

[140] Reihaneh Safavi-Naini and Ran Canetti, editors. *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*. Springer, 2012.

[141] Amit Sahai and Brent Waters. Fuzzy Identity-Based Encryption. In Cramer [60], pages 457–473.

[142] 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.

[143] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27:701–717, October 1980.

[144] Jae Hong Seo, Tetsutaro Kobayashi, Miyako Ohkubo, and Koutarou Suzuki. Anonymous hierarchical identity-based encryption with constant size ciphertexts. In Jarecki and Tsudik [98], pages 215–234.

[145] 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.

[146] Elaine Shi and Brent Waters. Delegating capabilities in predicate encryption systems. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 560–578. Springer, 2008.

[147] Elaine Shi and Brent Waters. Delegating capabilities in predicate encryption systems. In *Automata, Languages and Programming*, pages 560–578, 2008.

[148] Victor Shoup. Lower bounds for discrete logarithms and related problems. In *EUROCRYPT*, pages 256–266, 1997.

[149] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. http://eprint.iacr.org/.

[150] Victor Shoup, editor. *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*. Springer, 2005.

[151] 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/.

[152] Nigel P. Smart. Efficient key encapsulation to multiple parties. In Carlo Blundo and Stelvio Cimato, editors, *SCN*, volume 3352 of *Lecture Notes in Computer Science*, pages 208–219. Springer, 2004.

[153] Nigel P. Smart and Frederik Vercauteren. On computable isomorphisms in efficient asymmetric pairing-based systems. *Discrete Applied Mathematics*, 155(4):538–547, 2007.

[154] Serge Vaudenay, editor. *Public Key Cryptography - PKC 2005, 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets, Switzerland, January 23-26, 2005, Proceedings*, volume 3386 of *Lecture Notes in Computer Science*. Springer, 2005.

[155] Serge Vaudenay, editor. *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*. Springer, 2006.

[156] Eric R. Verheul. Evidence that XTR is more secure than supersingular elliptic curve cryptosystems. *J. Cryptology*, 17(4):277–296, 2004.

[157] Brent Waters. Efficient identity-based encryption without random oracles. In Cramer [60], pages 114–127.

[158] Brent Waters. Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.

[159] Brent Waters. Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *Public Key Cryptography*, volume 6571 of *Lecture Notes in Computer Science*, pages 53–70. Springer, 2011.

[160] Brent Waters. Functional Encryption for Regular Languages. In Safavi-Naini and Canetti [140], pages 218–235.

[161] Hoeteck Wee. Dual System Encryption via Predicate Encodings. In Yehuda Lindell, editor, *TCC*, volume 8349 of *Lecture Notes in Computer Science*, pages 616–637. Springer, 2014.

[162] Dae Hyun Yum and Pil Joong Lee. Generic construction of certificateless encryption. In Antonio Laganà, Marina L. Gavrilova, Vipin Kumar, Youngsong Mun, Chih Jeng Kenneth Tan, and Osvaldo Gervasi, editors, *Computational Science and Its Applications - ICCSA 2004, International Conference, Assisi, Italy, May 14-17, 2004, Proceedings, Part I*, volume 3043 of *Lecture Notes in Computer Science*, pages 802–811. Springer, 2004.