Advanced encryption for the sharing of sensitive data

Anaïs Barthoulot

Thèse de Doctorat

Université de Limoges
ED 653 – Sciences et Ingénierie (SI)
Faculté des Sciences et Techniques – Institut de Recherche XLIM

Orange
Orange Innovation – Caen

Thèse pour obtenir le grade de
Docteur de l'Université de Limoges
Informatique

Présentée et soutenue par
Anaïs Barthoulot - Université de Limoges, Orange

Le 18 décembre 2023

CHIFFREMENT AVANCÉ POUR LE PARTAGE DE DONNÉES SENSIBLES

Thèse dirigée par Olivier BLAZY, Sébastien CANARD et Philippe GABORIT

JURY :

Président du jury
Mme. Adeline ROUX-LANGLOIS, Directrice de Recherche CNRS – GREYC – Université de Caen Normandie

Rapporteurs
M. Guilhem CASTAGNOS, Maître de Conférences – IMB – Université de Bordeaux
M. David POINTCHEVAL, Directeur de Recherche CNRS – DI ENS – École Normale Supérieure

Examinateurs
M. Olivier BLAZY, Professeur – LIX – École Polytechnique
M. Sébastien CANARD, Professeur – LTCI – Télécom Paris
M. Philippe GABORIT, Professeur – XLIM – Université de Limoges
Mme. Elizabeth A. QUAGLIA, Associate Professor – Royal Holloway – Université de Londres
Mme. Carla RÀFOLS, Researcher – Université Pompeu-Fabra

J'ai fait des erreurs et j'en referai, j'espère juste ça sera pas les mêmes

Orelsan - Shonen

Remerciements

Bien que l'on m'ait suggéré de très bonnes idées pour écrire mes remerciements, comme utiliser de la stéganographie ou de la cryptographie, l'exercice me paraît suffisamment périlleux en lui-même et je suivrai donc sa forme classique. Il me semble normal de commencer en remerciant Sébastien Canard et Olivier Blazy pour leur confiance, leur accompagnement et leurs nombreux conseils pendant ces trois années et demi. De plus, je tiens à remercier Philippe Gaborit d'avoir accepté de reprendre la direction de cette thèse en début de deuxième année, afin de nous éviter un périple administratif complexe. Je remercie également David Pointcheval et Guilhem Castagnos d'avoir accepté de relire ce manuscrit, et je les remercie pour leurs précieux conseils. I also would like to thank Elizabeth Quaglia, Carla Ràfols, and Adeline Roux-Langlois for accepting the role of examiners for my Ph.D. defense. Merci aussi à Débora Thomas qui m'a beaucoup aidée pour la préparation des différentes missions.
Je souhaite également remercier les membres de l'équipe SPI de Caen, permanents, doctorants, et anciens doctorants, et en particulier Adel (les bons plans), Aïda, Bastien, Ferran, Jacques, Jean-François, Jérémy, Karel, Nicolas (du bas), Nicolas (du haut), Paul, et Olivier. Un remerciement tout particulier à Maxime (le chef du couloir) d'abord doctorant puis permanent, avec qui j'ai partagé ce bureau (toujours avec les radiateurs allumés) pendant 3 ans, et qui a été là pour moi lors de mon arrivée à Caen en pleine période Covid. Merci de m'avoir fait découvrir de nombreux bars de Caen (et l'embuscade qui porte bien son nom), merci pour les après-midi de co-télétravail, les sorties bike and run, et surtout merci pour les chansons qui restent dans la tête.
Merci à mes amis de Montpellier, Anthony, Bérénice, Clément, Fanny, Lisa, Marine, Sélène, Thomas d'avoir été là malgré la distance. Les vendredis soir à jouer à Among Us pendant les confinements / couvre-feux m'ont apporté beaucoup de joie lors de cette période pas évidente. Merci à Sélène d'être venue en vacances deux fois en Normandie (malgré la pluie). Merci à Bérénice d'être venue plusieurs fois (malgré son soi-disant test covid positif). Je te souhaite plein de courage et de réussite pour la fin de ta thèse.
Merci à mes amies de Caen, Aurélie, Calista (qui me fait me sentir moins vieille), Charlotte, Ikbelle, Margaux (avec son fameux nom de famille "Gomar"), Mélanie, Prune et Sophie (les deux autres Totally Spies), et Tiphaine. Mon parcours handballistique en Normandie n'aura pas été très simple, mais il m'aura permis de rencontrer chacune d'entre vous.

Merci à Chris(to) : t'avoir auprès de moi, avec ton côté tête en l'air et ce talent à ne jamais stresser, m'a permis de tenir pendant les moments difficiles. Et que dire de nos soirées jeux de société après le refus de mes papiers... Merci pour tout.
Enfin, merci à ma famille qui m'a soutenue tout au long de mes études (même si j'ai dû très vite arrêter de leur demander de l'aide pour les devoirs). Merci à ma sœur Chloé qui a corrigé mes nombreux CV et lettres de motivation, je n'aurais probablement pas eu ce stage chez Orange sans toi. Also thank you to my extended family - Ian and Christine, Paul and Chris, Anne and Graham, and Bob - who helped improve my English. A special thanks to Bob: you gave me an incredible opportunity by helping me secure this internship at Wilson James. I can certainly say that without it, I would not be here today. Your kindness and support when I arrived alone in London for the first time on June 1, 2019, were invaluable. I enjoyed all our meals during that summer, and I will never forget all the things you did for me.

Table of Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

ABE – Attribute-Based Encryption.
ANO-AugBE – Anonymous security for Augmented Broadcast Encryption scheme.
ANO-WIBE – Anonymous security for Identity-Based Encryption with Wildcard scheme.
AS – Authorization Server.
AugBE – Augmented Broadcast Encryption.
BDHE – Bilinear Diffie-Hellman Exponent.
BE – Broadcast Encryption.
BT – Broadcast and Trace.
CCA1 – Non-Adaptive Chosen-Ciphertext Attack.
CCA2 – Adaptive Chosen-Chiphertext Attack.
CDH – Computational Diffie-Hellman.
CP-ABE – Ciphertext Policy Attribute-Based Encryption.
CPA – Chosen-Plaintext Attack.
CS – Central Server.
DDH – Decisional Diffie-Hellman.
DLin – Decisional Linear.
DO – Data Owner.
DPVS – Dual Pairing Vector Spaces.
DS – Decisional Subspace.
DSE – Dual System Encryption.
eDDH – Extended Decisional Diffie-Hellman.
GGM – Generic Group Model.
GSD – General Subgroup Decision.
i.e. – Id Est, which corresponds to “that is” in English.
IdP – Identity Provider.
iff – If and only if.

IH-AugBE – Index-Hiding security for Augmented Broadcast Encryption scheme.
IND – Indistinguishability.
IND-BE – Indistinguishability security for Broadcast Encryption scheme.
IND-WIBE – Indistinguishability security for Identity-Based Encryption with Wildcard scheme.
IPE – Inner Product Encryption.
KGC – Key Generation Center.
KP-ABE – Key Policy Attribute-Based Encryption.
LVAS – Locally Verifiable Aggregate Signature.
MH-AugBE – Message-Hiding security for Augmented Broadcast Encryption scheme.
NM – Non-Malleability.
PAC – Pattern Certification Center.
PH-IPE – Payload-Hiding security for Inner Product Encryption scheme.
PH-WIBE – Pattern-Hiding security for Identity-Based Encryption with Wildcard scheme.
PKE – Public Key Encryption.
PPKG – Privacy-Preserving Key Generation.
PPT – Probabilistic Polynomial Time.
ROM – Random Oracle Model.
sAH-IPE – strong Attribute-Hiding security for Inner Product Encryption scheme.
SBDH – Strong Bilinear Diffie-Hellman.
SDH – Strong Diffie-Hellam.
SF – Semi-Functional.
sSBDH – Symmetric Strong Bilinear Diffie-Hellman.
SXDH – Symmetric External Diffie-Hellman.
TT – Traitor Tracing.
Unf – Unforgeability.
UPE – Unforgeability of Private Evaluation.
wAH-IPE – weak Attribute-Hiding security for Inner Product Encryption scheme.
WIBE – Identity-Based Encryption with Wildcard.
XDLin – eXternal Decision Linear.

1 Introduction

Contents

SINCE the inception of writing, humans have sought to protect their information from unwanted people. Two famous techniques, renowned since antiquity, are the *steganography* and the *cryptography*. While both terms are composed of the suffix “graphy”, signifying “writing”, their prefixes, “stegano” and “crypto”, carry different meanings: “covered” for the former and “hidden” for the latter. More precisely, the steganography involves concealing the message itself (by covering it most of the time), while cryptography consists in hiding the meaning of the message. In history, we can find a lot of examples for steganography and cryptography: the Greek Histiaeus used to shave his slaves’ heads to write messages on their skin and waited for his slaves’ hair to regrow before to send them (steganography), while the Spartan military employed the Scytale, a cylinder with a strip of parchment wrapped around it, upon which a message was written (cryptography).

Traditionally, cryptography was used for military purposes. The most famous example is the Enigma machine used by the Nazi Germany during World War II. The Enigma machine is composed of a keyboard (for the 26 letters of the alphabet) along with a set of rotating disks, called rotors, that “scramble” the alphabet letters. The settings of the machine, and especially the arrangement of the rotors determines the encryption scheme. The Enigma machine is also famous because it represents the first example of electronic cryptography; prior to its use, cryptography was entirely “handmade”. The security of the machine relied on the fact that at this time, no machine was powerful enough to try all the possible arrangements to decipher a given encryption.

However, today, cryptography is deployed in various domains, not limited to military communications. It is used in electronic commerce, personal communication, and the deployment of secure messaging applications. In short, cryptography is present wherever there is data to protect.

1.1 Foundations of Cryptography

Traditionally, cryptographic primitives used to rely on a single secret key used for both encryption and decryption. Besides still being used nowadays, notably the Advanced Encryption Standard (AES), such cryptographic schemes have some drawbacks, such as the fact that they require an efficient and secure way to share keys among concerned users. In the 1970s, Diffie and Hellman [58] introduced a new kind of cryptography with their key exchange protocol in which cryptographic schemes use a pair of keys instead of only one key: one key of the pair must be kept secret and is used for decryption, while the other is publicly known and used for encryption. Cryptographic schemes with only one key are referred to as *symmetric* or *secret key*, while those with a pair of keys are referred to as *asymmetric* or *public key*. In this thesis, we will mainly use asymmetric cryptographic primitives as they are better suited for messages sharing.

Cryptography provides four guarantees to users.

- **Confidentiality**: only the authorized receiver can learn the content of the message;
- **Integrity**: the receiver can ensure that the message has not been altered;
- **Authenticity**: the receiver can verify the origin of the message (that the sender is who she claims to be);
- **Non-Repudiation**: the sender cannot deny being the origin of the message.

1.2 Context

Nowadays, data are omnipresent in our daily lives, and we deal with a lot of them without even realizing it. For example, a typical workday for an office worker, let us call her Alice, could unfold as follows: first, she wakes up at around 6 a.m. to the sound of her connected alarm clock, then she checks her social media accounts before taking the bus to go to the office. During lunch, she visits a restaurant with her colleagues, and after work, she takes the underground to meet her personal trainer at the gym. While waiting, she books an appointment with her dentist on her phone. On her way back home, she stops by the grocery store to pick up the order she placed during her afternoon break,

and finally, she ends the day by watching TV series online. As we can observe, an average person like Alice deals with and also generates a significant amount of data in a single day. It begins during the night and in the morning when the connected alarm clock analyzes Alice's sleep to determine the perfect time to wake her up. When Alice is active on social media, she accesses others' data while also sharing her own when she posts photos, for example. While taking the bus, Alice uses her travel card, which contains various information about her. At work, Alice handles the company's and its clients' data, and during lunch, when she uses her credit card for payment, sensitive data is involved. When she books her medical appointment on the underground, she deals with sensitive data in an insecure environment. At the gym with her personal trainer, she shares data collected by her connected watch and an associated training application. When collecting her order at the store, she has to pay online, thus sending sensitive data over the internet. Finally, while watching series online, she consumes a significant amount of data and shares some information about her preferences.

Storing data securely and efficiently is one of the most studied and challenging areas today. On one hand, storing an increasing amount of data (and keeping it readily available) demands significant resources, including electricity and water. In an era of environmental concerns, there is a strong motivation to research efficient storage solutions that reduce the size of stored data. On the other hand, privacy concerns are on the rise, with users becoming more cautious about the data they share and with whom they share it, in light of recent laws such as the General Data Protection Regulation (GDPR) established by the European Union in 2016 to protect personal data. There exist several methods to store data while keeping its privacy, including encryption (such as AES for symmetric and RSA for asymmetric encryption), differential privacy (adding noise to data), and data masking techniques like tokenization and pseudonymization.

However, there is also a need to share data, which necessitates the development of advanced data sharing techniques. This forms the scope of this thesis. We condider that there are three types of data sharing: i) when information is shared from one individual to a group of users, ii) when information is shared from one individual to a group with *common* attributes, and iii) when information is shared from one individual to another. In this thesis, we focus on the first two types of data sharing, studying two cryptographic primitives known as *Broadcast Encryption* and *Attribute-Based Encryption*, which respectively fall into the contexts of i) and ii).

Furthermore, it is essential to acknowledge that data are not only stored but also processed by servers to respond to user requests, which sometimes requires substantial computational power from the server. While some promising theoretical techniques for

storage and computations over encrypted data, such as Gentry's fully homomorphic encryption (FHE), have been developed, they are often inefficient and resource-intensive for practical deployment. It is important to note that, although highly challenging and intriguing, FHE is beyond the scope of this thesis.

1.3 Encryption Scheme for Data Sharing

To a group of users. For this type of sharing we focus on *broadcast encryption* (BE) [66] schemes. A broadcast encryption scheme is a public key encryption system that encrypts messages for a specific subset of users, ensuring that only users within that subset can decrypt them. Figure 1.1 provides a brief and informal overview of how a broadcast encryption scheme operates.

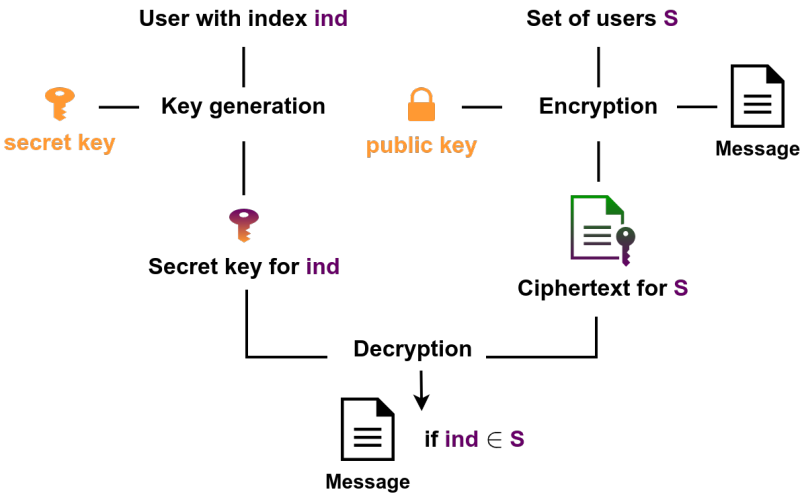


Figure 1.1: Broadcast encryption scheme, simplified.

In our daily lives, such schemes find application in scenarios like paid television or secure data sharing. Another cryptographic primitive, known as *broadcast and trace* (BT), serves a dual purpose by enabling both broadcast encryption and traitor tracing. The latter is a mechanism that helps identify individuals, or at least one of them, responsible for creating a pirate decoder for the broadcast encryption scheme. Broadcast and trace is also employed in the realm of paid television to uncover those responsible for distributing illegal decoders online.

An encryption scheme is considered *efficient* when its parameters remain independent of the number of users within the scheme, even when dealing with large user bases. For broadcast encryption schemes, achieving constant-size ciphertexts has been possible

since 2005 using standard objects and assumptions. However, attaining constant-size public and private keys under *standard assumptions* remains an open challenge. In the context of traitor tracing, there is currently no scheme that offers constant-size ciphertexts, even when considering non-standard assumptions. Currently, the most efficient broadcast encryption and broadcast and trace schemes rely on a combination of pairing and lattice techniques to optimize efficiency. Nevertheless, this approach has the drawback of partial reliance on computational problems that could be compromised by a quantum computer. Therefore, in this thesis, we propose efficient schemes that do not merge these two domains. Whenever feasible, we aim to provide generic constructions of such schemes, enabling implementation with either pairings or lattices.

To a group with common attributes. For this type of sharing we concentrate on *attribute-based encryption* [126] (ABE) schemes. An attribute-based encryption scheme is a public key encryption system in which messages and secret keys are associated with either a set of attributes or an access policy. There are two types of attribute-based encryption schemes: ciphertext policy attribute-based encryption (CP-ABE), where ciphertexts are linked to access policies, and secret keys are linked to sets of attributes; and key policy attribute-based encryption (KP-ABE), where the roles of attributes and access policies are reversed. We provide a brief and informal overview of how the CP-ABE scheme operates in Figure 1.2. A KP-ABE scheme can be described similarly.

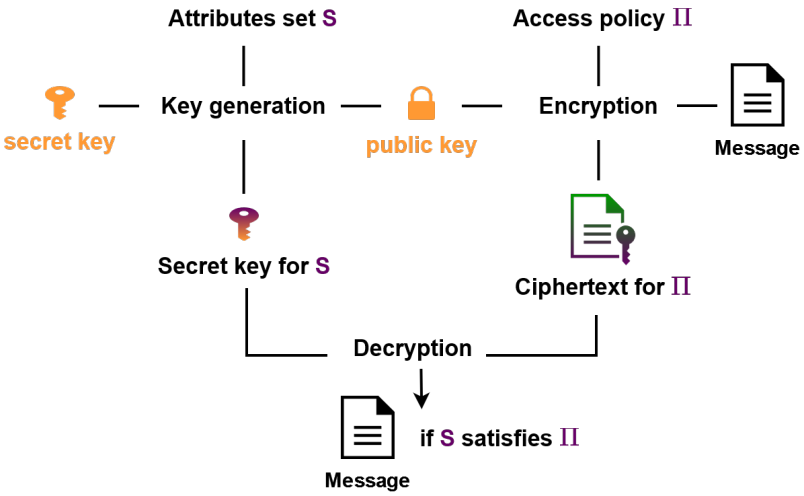


Figure 1.2: Ciphertext policy attribute-based encryption scheme, simplified.

Since both ciphertexts and secret keys in attribute-based encryption schemes depend on attribute sets, designing efficient schemes that keep the size of keys and ciphertexts independent of the maximum number of attributes in the scheme is quite challenging.

While some schemes have achieved either constant size ciphertexts or constant size secret keys, there is currently no scheme that accomplishes both. Additionally, it is important to note that the complexity of the access policies affects the scheme's efficiency: the more complex the policy, the less efficient the scheme becomes. However, some applications, such as developing a sharing platform for connected objects, require fine-grained access control. In this thesis, we have chosen to focus on improving the efficiency of attribute-based encryption schemes by using simple access policies.

1.4 Our Toolbox of Cryptographic Primitives

The goal of our thesis is to propose generic constructions for broadcast encryption and attribute-based encryption schemes. The advantage of these constructions lies in the modularity of our results. However, developing such constructions is not a straightforward task, and we must employ two underlying additional primitives: *identity-based encryption with wildcards* and *cryptographic accumulators*. Furthermore, these primitives require new features or security properties to be effectively utilized in our constructions.

Identity-based encryption with wildcards [4]. An identity-based encryption with wildcards (WIBE) scheme is a public key encryption system wherein ciphertexts are encrypted based on a vector known as a *pattern*. Only users possessing a pattern corresponding to the encryption pattern can decipher the message. Patterns are defined within a set that includes a special symbol, "$\star$", referred to as a "wildcard". A secret key created for pattern $\boldsymbol{P}$ can decrypt a ciphertext associated with pattern $\boldsymbol{P}^{'}$ if and only if, for all $i \in \{1, \cdots, L\}$ (where $L \in \mathbb{N}$ is the length of both patterns), either $P_i = \star$ or $P_i^{'} = \star$ or $P_i = P_i^{'}$. In this scenario, we say that $\boldsymbol{P}$ *matches* $\boldsymbol{P}^{'}$, denoted as $\boldsymbol{P} =_{\star} \boldsymbol{P}^{'}$. For visual clarity, Figure 1.3 provides an example illustrating matching and not matching patterns for patterns defined over $\{0, 1, \star\}^*$.

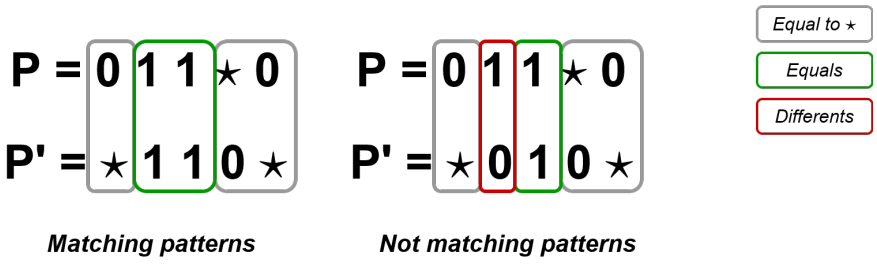


Figure 1.3: Example of matching and not matching patterns.

In Figure 1.4, we provide a brief and informal overview of how an identity-based encryption with wildcards scheme operates.

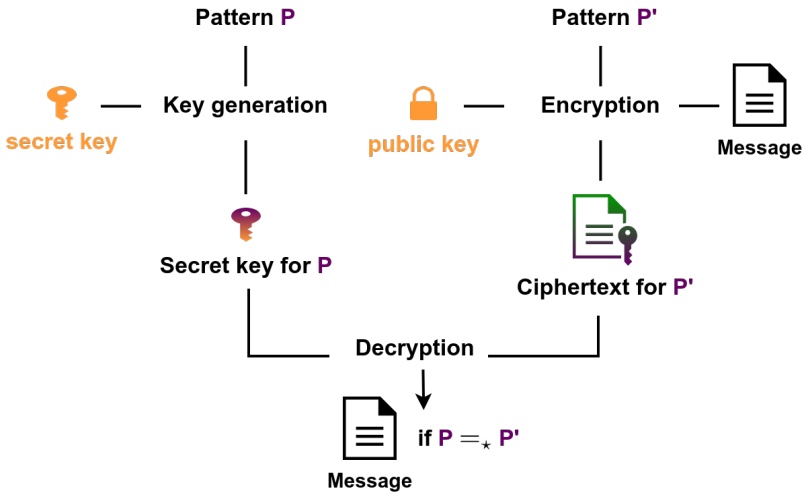


Figure 1.4: Identity-based encryption with wildcards scheme, simplified.

Cryptographic accumulators [27]. A cryptographic accumulator is a system designed to aggregate a set of values into a concise representation while also possessing the capability to prove the membership of any element using a piece of information referred to as a *witness*. This system is parameterized by a tuple consisting of private and secret keys. In Figure 1.5, we provide a brief and informal explanation of how an accumulator scheme functions.

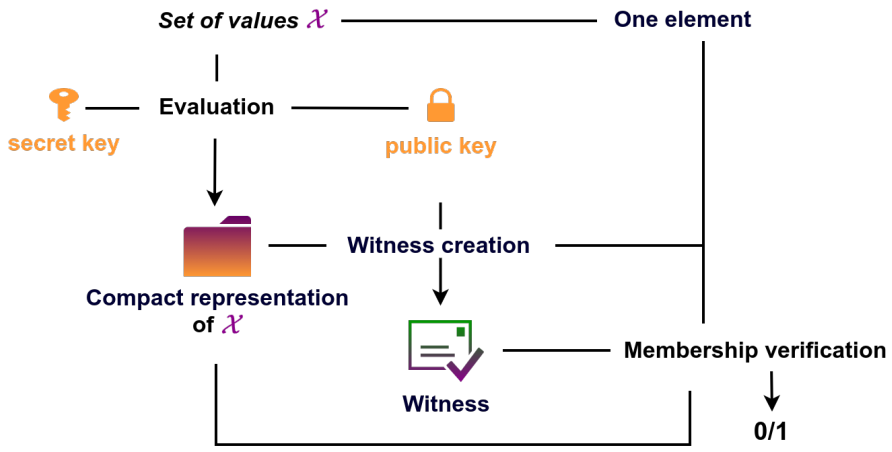


Figure 1.5: Cryptographic accumulator, simplified.

1.5 Contributions

We hereby present all our contributions, categorizing them into two distinct categories: those pertaining to the building blocks (WIBE and accumulators), and those related to

advanced primitives for data sharing (BE and ABE).

We commence by detailing our contributions related to the building blocks.

Contribution 1.1: we define a novel type of identity-based encryption with wildcards scheme known as *privacy-preserving key keneration* (PPKG-WIBE). In this scheme, the key generation algorithm is replaced by an interactive protocol involving three entities: a user requesting a key for a pattern $\boldsymbol{P}$, a pattern certification center certifying that the requesting user indeed possesses $\boldsymbol{P}$, and a key generation center responsible for creating the key for $\boldsymbol{P}$. Additionally, we introduce a new security property for PPKG-WIBE schemes, ensuring that the key generation center remains unaware of any information about $\boldsymbol{P}$.

Contribution 1.2: we introduce a novel security property for identity-based encryption with wildcards schemes, known as *pattern-hiding*. This property safeguards the encryption pattern, ensuring its confidentiality even from users authorized to decrypt. This property supersedes an existing property that protects the encryption pattern but only from users not allowed to decrypt.

Contribution 1.3: we also elaborate two new identity-based encryption with wildcards scheme: the first scheme has constant size ciphertext while the second scheme is pattern-hiding.

Contribution 1.4: we propose the first *privacy-preserving key generation* identity-based encryption with wildcards instantiation.

Contribution 1.5: in the context of cryptographic accumulators, we introduce a novel security property termed *unforgeability of private evaluation*. This property aims to prevent the forgery of the accumulated value when it is computed using the scheme's private key.

Contribution 1.6: we also introduce a novel type of accumulator known as *dually computable* accumulators. These accumulators incorporate two distinct evaluation algorithms: one that requires the scheme's secret key as input and another that relies solely on the public key.

Contribution 1.7: we present two novel accumulator schemes. The first one is original in the literature because it creates the compact representation using the secret key, while the witness is generated using only the public key. Our second scheme is a *dually computable* accumulator.

And here are our contributions in the realm of advanced primitives.

Contribution 2.1: we present a generic construction for broadcast encryption schemes, along with a variant known as *augmented broadcast encryption*, derived from identity-based encryption with wildcards schemes. For the augmented broadcast

encryption variant, we establish the necessity of our newly introduced *pattern-hiding* security property for identity-based encryption with wildcards schemes.

Contribution 2.2: by combining our generic constructions and novel schemes, we introduce a new broadcast encryption scheme featuring ciphertexts of constant size, secure under standard assumptions. Additionally, this marks the first instance of an augmented broadcast encryption scheme secure under standard assumptions.

Contribution 2.3: in the domain of attribute-based encryption schemes, we propose a novel scheme constructed using *dually computable* accumulators. This innovation results in the first attribute-based encryption scheme with constant size ciphertexts and secret keys. It is also noteworthy that this marks the first application of cryptographic accumulators for encryption, extending their use beyond key management.

Contribution 2.4: we explore, through a practical use case involving a sharing platform for connected devices, the utility of ciphertext policy attribute-based encryption for access control. We provide evidence that when this encryption method safeguards both the access policy associated with the ciphertext and the user's attributes during secret key requests, our access control protocol upholds the privacy of all platform participants. Furthermore, we introduce a novel ciphertext policy attribute-based encryption scheme derived from the *privacy-preserving key generation* identity-based encryption with wildcards scheme.

These contributions have led to the acceptance of some articles, while others are currently under submission.

- Our two articles "(Augmented) Broadcast Encryption From Identity-Based Encryption with Wildcards" [20] and "Dually Computable Cryptographic Accumulators and Their Application To Attribute-Based Encryption"[22] were respectively presented at the 21st International Conference on Cryptology and Network (CANS 2022) and the 22st International Conference on Cryptology and Network (CANS 2023).
- Our articles "SoK: Recent Developments in Cryptographic Accumulators - Properties, Security, and Beyond" (presenting Contribution 1.5) and "Adapting Identity-based Encryption with Wildcards to Access Control" (Contributions 1.1, 1.4 and 2.4) are under submission. More details about our publications are given in Section 8.

Table 1.1 provides a list of articles and chapters where you can find our various contributions.

<table>
<tr><th rowspan="2">Contributions</th><th rowspan="2">Chapter</th><th rowspan="2">In submission</th><th colspan="2">Accepted</th></tr>
<tr><th>Proceedings</th><th>Extended Version</th></tr>
<tr><td>Contribution 1.1</td><td rowspan="2">4.2</td><td>√</td><td></td><td></td></tr>
<tr><td>Contribution 1.2</td><td rowspan="2"></td><td rowspan="2">[20]</td><td rowspan="2">[21]</td></tr>
<tr><td>Contribution 1.3</td><td rowspan="2">4.3</td></tr>
<tr><td>Contribution 1.4</td><td rowspan="2">√</td><td></td><td></td></tr>
<tr><td>Contribution 1.5</td><td rowspan="2">5.3</td><td rowspan="5"></td><td rowspan="2">[22]</td><td rowspan="2">[23]</td></tr>
<tr><td>Contribution 1.6</td></tr>
<tr><td>Contribution 1.7</td><td>5.4</td></tr>
<tr><td>Contribution 2.1</td><td rowspan="2">6.1</td><td rowspan="2">[20]</td><td rowspan="2">[21]</td></tr>
<tr><td>Contribution 2.2</td></tr>
<tr><td>Contribution 2.3</td><td>6.2</td><td>[22]</td><td>[23]</td></tr>
<tr><td>Contribution 2.4</td><td>6.3</td><td>√</td><td></td><td></td></tr>
</table>

Table 1.1: Summary of our contributions and publications.

1.6 Organization of this Thesis

This thesis comprises five technical chapters, in addition to this introduction. Chapter 2 provides the necessary mathematical background required for a comprehensive understanding of this manuscript. In Chapter 3, we present some cryptographic preliminaries. Chapter 4 is dedicated to the identity-based encryption with wildcards primitive, featuring a formal definition of the scheme, its associated properties, our contributions to this primitive, and our novel instantiations. Moving on to Chapter 5, we shift our focus to cryptographic accumulators. This chapter formally introduces the primitive, engages in discussions regarding its properties and applications, presents our new functionalities, and unveils our innovative accumulator schemes. In Chapter 6, we explore the applications of the aforementioned primitives in two data sharing schemes: broadcast encryption and attribute-based encryption, complete with a practical use case. Finally, Chapter 7 summarizes the contributions made in this thesis.

2

Mathematical Background

Contents

In this chapter, we introduce the notations used in this manuscript along with the mathematical notions and security assumptions. We also present some cryptographic preliminaries and techniques that we will use in this thesis.

2.1 Notations and Mathematical Background

- Vectors are written with **bold face** lower case letters, patterns and matrices with **bold face** upper case letters.
- $\mathbb{R}, \mathbb{N}, \mathbb{Z}, \mathbb{Z}_N = \mathbb{Z}/N\mathbb{Z}$ respectively represent the usual sets of real numbers, natural numbers, integers and integers modulo N.
- $GL(n, \mathbb{Z}_p)$ for $n \in \mathbb{N}$ and p prime is the set of $n \times n$ invertible matrices over $\mathbb{Z}_p$. It is called the *general linear group*.
- For a set S and an integer l, S^l corresponds to $\underbrace{S \times \cdots \times S}_{l \text{ times}}$.
- For $a, b \in \mathbb{N}$ we denote $\{1, 2, \cdots, a\}$ as $[a]$, and $\{a, a+1, \cdots, b\}$ as $[\![a, b]\!]$ when $a \neq 1$.
- For every finite set S, $x \leftarrow S$ denotes a uniformly random element x from the set S.
- A security parameter is denoted by λ, where $\lambda \in \mathbb{N}$.
- The notation "$\in \text{poly}(\lambda)$" means to be polynomial in the security parameter.
- Unless specified, we consider that any Probabilistic Polynomial Time (PPT) adversary $\mathcal{A}$ has output in $\{0, 1\}$.
- The advantage of an adversary $\mathcal{A}$ to win a security game Game is written $\text{Adv}_{\mathcal{A}}^{\text{Game}}$.

Definition 2.1.1 ***Negligible function.*** *A function $\epsilon : \mathbb{N} \to \mathbb{R}$ is called* negligible *if for all $k \in \mathbb{N}$, there exists $N \in \mathbb{N}$ such that for all $n \geq N$, we have that $|\epsilon(n)| < \frac{1}{n^k}$.*

Definition 2.1.2 ***Characteristic polynomial*** *[73]. A set $\mathcal{X} = \{x_1, \cdots, x_n\}$ with elements $x_i \in \mathbb{Z}_p$ can be represented by a polynomial following an idea introduced in [67]. The polynomial $Ch_{\mathcal{X}}[z] = \prod_{i=1}^{n}(x_i + Z)$ from $\mathbb{Z}_p[Z]$, where Z is a formal variable, is called the* characteristic polynomial *of $\mathcal{X}$. In what follows, we will denote this polynomial simply by $Ch_{\mathcal{X}}$ and its evaluation at a point y as $Ch_{\mathcal{X}}(y)$.*

Definition 2.1.3 ***Elementary symmetric polynomial.*** *The* elementary symmetric polynomial *on $n \in \mathbb{N}$ variables $\{X_i\}$ of degree $k \leq n$ is the polynomial $\sigma_k(X_1, \cdots, X_n) = \sum_{1 \leq i_1 \lneq \cdots \lneq i_k \leq n} X_{i_1} \cdots X_{i_k}$. Notice that $\sigma_1(X_1, \cdots, X_N) = \sum_{i=1}^{n} X_i$ and $\sigma_n = \prod_{i=1}^{n} X_i$.*

Note 2.1.1 *Let $\mathcal{X} = \{X_1, \cdots, X_n\}$. Notice that $Ch_{\mathcal{X}}[Z]$, which is equals to $\prod_{i=1}^{n}(X_i + Z)$ by definition, is also equals to $Z^n + \sigma_1(X_1, \cdots, X_n)Z^{n-1} + \sigma_2(X_1, \cdots, X_n)Z^{n-2} + \cdots + \sigma_n(X_1, \cdots, X_n)$.*

We now recall some algebra definitions.

Definition 2.1.4 ***Group.*** *Let $\mathbb{G}$ be a non-empty set and $\cdot$ be a binary law over $\mathbb{G}$. We say that $(\mathbb{G}, \cdot)$ is a group if it satisfies three requirements:*

- *the law $\cdot$ is* associative*, meaning that for all $g_1, g_2, g_3 \in \mathbb{G}$, we have that $(g_1 \cdot g_2) \cdot g_3 = g_1 \cdot (g_2 \cdot g_3)$;*
- *there exists an* identity element *in $\mathbb{G}$, denoted $\mathbb{1}_{\mathbb{G}}$, such that for all $g \in \mathbb{G}$, we have that $\mathbb{1}_{\mathbb{G}} \cdot g = g \cdot \mathbb{1}_{\mathbb{G}} = g$;*
- *for each $g_1 \in \mathbb{G}$, there exists an element $g_2 \in \mathbb{G}$ such that $g_1 \cdot g_2 = \mathbb{1}_{\mathbb{G}}$ and $g_2 \cdot g_1 = \mathbb{1}_{\mathbb{G}}$, where $\mathbb{1}_{\mathbb{G}}$ is the identity element. For each g_1, the element g_2, called the* inverse *of g_1, is unique and denoted by g_1^{-1}.*

Notation 2.1.1 *For any $g \in \mathbb{G}$ and any $i \in \mathbb{N}$, we write $g^i = \underbrace{g \cdot \cdots \cdot g}_{i \text{ times}}$.*

Notation 2.1.2 *We here decided to define a group with multiplicative notation $\cdot$. We can also define a group with additive notation, where the law, the identity element and the inverse element of g_1 are respectively written $+$, $\mathbb{0}_{\mathbb{G}}$ and $-g_1$.*

Definition 2.1.5 ***Commutative group.*** *The group $(\mathbb{G}, \cdot)$ is said to be* commutative *(or* abelian*) if for all $g_1, g_2 \in \mathbb{G}, g_1 \cdot g_2 = g_2 \cdot g_1$.*

Definition 2.1.6 ***Finite group.*** $(\mathbb{G}, \cdot)$ *is said to be* finite *if* $\mathbb{G}$ *is finite.*

Definition 2.1.7 ***Subgroup.*** *Let* $\mathbb{H}$ *be a subset of* $\mathbb{G}$. $(\mathbb{H}, \cdot)$ *is a subgroup of* $(\mathbb{G}, \cdot)$ *if it satisfies three requirements:*

- $\mathbb{1}_{\mathbb{G}} \in \mathbb{H}$.
- *For all* $h_1, h_2 \in \mathbb{H}$, $h_1 \cdot h_2 \in \mathbb{H}$.
- *For all* $h \in \mathbb{H}$, $h^{-1} \in \mathbb{H}$.

Definition 2.1.8 *For any element* $g \in \mathbb{G}$, *the set* $\left\{g^k | k \in \mathbb{N}\right\}$ *that consists of all integer powers of* g *and denoted by* $\langle g \rangle$ *is the subgroup generated by* g.

Definition 2.1.9 ***Order.*** *The* order *of a (finite) group is the number of elements in the set. The* order *of an element* $g \in \mathbb{G}$, *written* $|g|$, *is the order of the finite subgroup* $\langle g \rangle$ *generated by* g, *i.e. the least positive integer* n *such that* $g^n = 1$ *(if it exists).*

Definition 2.1.10 ***Cyclic group.*** *A group* $(\mathbb{G}, \cdot)$ *is said to be* cyclic *if there exists an element* $g \in \mathbb{G}$ *such that* $\mathbb{G} = \langle g \rangle$. g *is then called the* generator *of the group* $(\mathbb{G}, \cdot)$.

Note 2.1.2 *Any group* $\mathbb{G}$ *of prime order is cyclic, and any element* $g \in \mathbb{G} \setminus \{\mathbb{1}_{\mathbb{G}}\}$ *is a generator of* $\mathbb{G}$.

Definition 2.1.11 ***Direct product of groups.*** *Let* $(\mathbb{G}, *)$ *and* $(\mathbb{H}, \odot)$ *be two groups. The* direct product *of* $(\mathbb{G}, *)$ *and* $(\mathbb{H}, \odot)$ *is the group denoted* $(\mathbb{G}, *) \times (\mathbb{H}, \odot)$ *with elements* (g, h) *where* $g \in \mathbb{G}$ *and* $h \in \mathbb{H}$. *For all* $g_1, g_2 \in \mathbb{G}, h_1, h_2 \in \mathbb{H}$, $(g_1, h_1) \cdot (g_2, h_2) = (g_1 * g_2, h_1 \odot h_2)$ *where* $\cdot$ *denote the group operation of* $(\mathbb{G}, *) \times (\mathbb{H}, \odot)$.

Definition 2.1.12 ***Group isomorphism.*** *Let* $(\mathbb{G}, *)$ *and* $(\mathbb{H}, \odot)$ *be two groups and* $f : \mathbb{G} \to \mathbb{H}$ *be a bijective function (i.e.* injective*, meaning that each element of* $\mathbb{H}$ *is only paired with one element of* $\mathbb{G}$*, and* surjective*, meaning that each element of* $\mathbb{H}$ *has an antecedent by* f *in* $\mathbb{G}$*). If for all* $g_1, g_2 \in \mathbb{G}$, $f(g_1 * g_2) = f(g_1) \odot f(g_2)$ *then* f *is a* group isomorphism.

Definition 2.1.13 ***Groups isomorphic.*** *Two groups* $(\mathbb{G}, *)$ *and* $(\mathbb{H}, \odot)$ *are said to be* isomorphic *if there exists an isomorphism from one to the other. In this case, we write* $(\mathbb{G}, *) \approx (\mathbb{H}, \odot)$.

Note 2.1.3 *A group* $(\mathbb{G}, *)$ *can be isomorphic to the direct product of several groups* $(\mathbb{H}_1, \odot_1), \cdots, (\mathbb{H}_l, \odot_l)$*, for* l *an integer. In this case we write* $(\mathbb{G}, *) \approx (\mathbb{H}_1, \odot_1) \times \cdots \times (\mathbb{H}_l, \odot_l)$*.*

In the following we will refer to the group $(\mathbb{G}, \cdot)$ as $\mathbb{G}$ and $\mathbb{1}_{\mathbb{G}}$ will sometimes be written 1. Thus sometimes $(\mathbb{G}, *) \approx (\mathbb{H}, \odot)$ will be denoted $\mathbb{G} \approx \mathbb{H}$.

Definition 2.1.14 ***Ring.*** *Let* $\mathbb{A}$ *be a non-empty set provided with two internal law* $+$ *(addition) and* $\cdot$ *(multiplication). We say that* $(\mathbb{A}, +, \cdot)$ *is a* ring *if:*

- $(\mathbb{A}, +)$ *is a commutative group, where the (additive) identity element is written* $\mathbb{0}_{\mathbb{A}}$ *and the (additive) inverse of element* $a \in \mathbb{A}$ *is written* $-a$*.*
- *Multiplication is associative: for all* $a_1, a_2, a_3 \in \mathbb{A}$*,* $(a_1 \cdot a_2) \cdot a_3 = a_1 \cdot (a_2 \cdot a_3)$*.*
- *Multiplication is distributive with respect to addition: for all* $a_1, a_2, a_3 \in \mathbb{A}$*,* $a_1 \cdot (a_2 + a_3) = (a_1 \cdot a_2) + (a_1 \cdot a_3)$ *and* $(a_2 + a_3) \cdot a_1 = (a_2 \cdot a_1) + (a_3 \cdot a_1)$*.*

Definition 2.1.15 ***Commutative ring.*** *The ring* $(\mathbb{A}, +, \cdot)$ *is said to be* commutative *if the multiplication is commutative, meaning that for all* $a_1, a_2 \in \mathbb{A}$*,* $a_1 \cdot a_2 = a_2 \cdot a_1$*.*

Definition 2.1.16 ***Multiplicative identity.*** *Let* $(\mathbb{A}, +, \cdot)$ *be a ring. If there exists an element* $\mathbb{1}_{\mathbb{A}} \in \mathbb{A}$ *such that for all* $a \in \mathbb{A}$*,* $a \neq$ *,* $\mathbb{1}_{\mathbb{A}} \cdot a = a \cdot \mathbb{1}_{\mathbb{A}} = a$*, we say that* $\mathbb{1}_{\mathbb{A}}$ *is the multiplicative identity. The ring* $(\mathbb{A}, +, \cdot)$ *is then called* unit ring*.*

Definition 2.1.17 ***Multiplicative inverse.*** *Let* $(\mathbb{A}, +, \cdot)$ *be a ring. For* $a_1 \in \mathbb{A}$*, if there exists an element* $a_2 \in \mathbb{A}$ *such that* $a_1 \cdot a_2 = \mathbb{1}_{\mathbb{A}}$ *and* $a_2 \cdot a_1 = \mathbb{1}_{\mathbb{A}}$ *we say that* a_2 *is the* multiplicative inverse *of* a_1 *and is written* a_1^{-1}*.*

Definition 2.1.18 ***Field.*** *A ring* $(\mathbb{A}, +, \cdot)$ *is called a* field *if all elements in* $\mathbb{A}$ *except* $\mathbb{0}_{\mathbb{A}}$ *have a multiplicative inverse.*

Notation 2.1.3 *The ring* $(\mathbb{A}, +, \cdot)$ *is sometimes refer to as* $\mathbb{A}$ *and a field is often denoted by* $(\mathbb{F}, +, \cdot)$ *or simply by* $\mathbb{F}$*.*

Definition 2.1.19 ***Vector space.*** *Let* $\mathbb{F}$ *be a field, and* $\mathbb{V}$ *be a non-empty set together with two binary operations* $+ : \mathbb{V} \times \mathbb{V} \to \mathbb{V}$ *and* $\cdot : \mathbb{F} \times \mathbb{V} \to \mathbb{V}$*. We say that* $(\mathbb{V}, +, \cdot)_{\mathbb{F}}$ *is a* vector space over a field $\mathbb{F}$ *if it satisfies the following conditions, where elements of* $\mathbb{F}$ *are called* scalar *and elements of* $\mathbb{V}$ *are called* vectors*:*

- $(\mathbb{V}, +)$ *is an abelian group.*
- *Scalar multiplication is* associative*: for any* $\lambda_1, \lambda_2 \in \mathbb{F}, \boldsymbol{v} \in \mathbb{V}$*,* $\lambda_1 \cdot (\lambda_2 \cdot \boldsymbol{v}) = (\lambda_1 \cdot \lambda_2) \cdot \boldsymbol{v}$*.*

- *Scalar multiplication is distributive over vector addition: for any $\boldsymbol{v}_1, \boldsymbol{v}_2 \in \mathbb{V}$ and any $\lambda \in \mathbb{F}$, $\lambda(\boldsymbol{v}_1 + \boldsymbol{v}_2) = \lambda\boldsymbol{v}_1 + \lambda\boldsymbol{v}_2$.*
- *Scalar multiplication is distributive over scalar addition: for any $\lambda_1, \lambda_2 \in \mathbb{F}$ and any $\boldsymbol{v} \in \mathbb{V}$: $(\lambda_1 \oplus \lambda_2) \cdot \boldsymbol{v} = \lambda_1 \cdot \boldsymbol{v} + \lambda_2 \cdot \boldsymbol{v}$, where $\oplus$ is the addition law over $\mathbb{F}$.*
- *There exists an element $\mathbb{1}_{\mathbb{F}} \in \mathbb{F}$ such that for all $\boldsymbol{v} \in \mathbb{V}$, $\mathbb{1}_{\mathbb{F}} \cdot \boldsymbol{v} = \boldsymbol{v} \cdot \mathbb{1}_{\mathbb{F}} = \boldsymbol{v}$. This element is called the* identity element of scalar multiplication.

In the following, we will refer to the vector space $(\mathbb{V}, +, \cdot)_{\mathbb{F}}$ as $\mathbb{V}_{\mathbb{F}}$, or as $\mathbb{V}$ over $\mathbb{F}$.

Definition 2.1.20 ***Linear combination.*** *Let $\mathbb{V}$ be a vector space over field $\mathbb{F}$ and $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\}$ (for $k \in \mathbb{N}$) be a set of elements of $\mathbb{V}$. A* linear combination *of the elements $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k$ is an element of $\mathbb{V}$ of the form*

$$a_1 \cdot \boldsymbol{v}_1 + a_2 \cdot \boldsymbol{v}_2 + \cdots + a_k\boldsymbol{v}_k$$

where the scalar $a_1, \cdots a_k \in \mathbb{F}$ are called the coefficients *of the linear combination.*

Definition 2.1.21 ***Linear independence.*** *Let $\mathbb{V}$ be a vector space over field $\mathbb{F}$ and $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\}$ (for $k \in \mathbb{N}$) be a set of elements of $\mathbb{V}$. The elements $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k$ are said to be* linearly independent *if a linear combination of them results in the vector $\mathbf{0}$ if and only if all its coefficients are zero.*

Definition 2.1.22 ***Basis and dimension.*** *Let $\mathbb{V}$ be a vector space over field $\mathbb{F}$, and B be a subset of $\mathbb{V}$. We say that B is a* basis *of $\mathbb{V}$ if its elements are linearly independent and if every element of $\mathbb{V}$ can be written as a unique finite linear combination of the elements of B. The number of elements in B is called the* dimension *of $\mathbb{V}$ and is the same for all bases of $\mathbb{V}$. If $|B|$ is finite, we say that $\mathbb{V}$ has* finite *dimension.*

Definition 2.1.23 ***Linear isomorphism.*** *Let $\mathbb{V}, \mathbb{W}$ be two vector spaces over the same field $\mathbb{F}$. A function $f : \mathbb{V} \to \mathbb{W}$ is to be a* linear map *if for any two vectors $\boldsymbol{v}_1, \boldsymbol{v}_2 \in \mathbb{V}$ and any scalar $\lambda \in \mathbb{F}$ the following conditions are respected:*

- $f(\boldsymbol{v}_1 + \boldsymbol{v}_2) = f(\boldsymbol{v}_1) + f(\boldsymbol{v}_2)$.
- $f(\lambda\boldsymbol{v}_1) = \lambda f(\boldsymbol{v}_1)$.

If f is bijective *then f is called* linear isomorphism.

Note 2.1.4 *Let $\mathbb{F}$ be a field and $\mathbb{F}^n$ be the set of the n-tuples of elements of $\mathbb{F}$. Then $\mathbb{F}^n$ is a vector space over $\mathbb{F}$.*

Definition 2.1.24 ***Canonical basis.*** *Let $\boldsymbol{e}_i$ for $i \in [n]$ be vectors equals to $_{\mathbb{F}}$ except at position i where they are equals to $\mathbb{1}_{\mathbb{F}}$. Then $\{\boldsymbol{e}_i\}_{i=1}^n$ form an basis of $\mathbb{F}^n$ and is called* standard basis *or* canonical basis*.*

Let $\mathbb{V}$ be a vector space of finite dimension n over field $\mathbb{F}$ and $\phi : \mathbb{F}^n \to \mathbb{V}$ be a linear isomorphism. Then the image of $\{\boldsymbol{e}_i\}_{i=1}^n$ by ϕ forms the canonical *basis of $\mathbb{V}$.*

2.2 Bilinear Pairing

In this thesis, we will focus on a particular branch of cryptography, called *pairing-based cryptography* which relies on a specific function called a *pairing*. We formally define this function, its properties and security assumptions related.

2.2.1 Definitions

Definition 2.2.1 ***Bilinear Pairing.*** *A* bilinear pairing *on $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$, where $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are three groups of same order N, is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ that satisfies the following conditions:*

1. Bilinearity*: for all $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$, $a, b \in \mathbb{Z}_N$, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.*
2. Non-degeneracy*: for all $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$ such that $g_1 \neq \mathbb{1}_{\mathbb{G}_1}$ and $g_2 \neq \mathbb{1}_{\mathbb{G}_2}$, we have that $e(g_1, g_2) \neq \mathbb{1}_{\mathbb{G}_T}$, where $\mathbb{1}_{\mathbb{G}_1}, \mathbb{1}_{\mathbb{G}_2}, \mathbb{1}_{\mathbb{G}_T}$ are respectively the identity element of $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$.*
3. Computability*: e must be efficiently computable.*

Actually there exist three kinds of bilinear pairing:

- *Type 1*: in this kind of paring $\mathbb{G}_1 = \mathbb{G}_2$. The pairing is also said to be *symmetric*. In this case we will denote $\mathbb{G}_1$ by $\mathbb{G}$.
- *Type 2*: in this kind of pairing $\mathbb{G}_1 \neq \mathbb{G}_2$ and there exists an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$.
- *Type 3*: in this kind of pairing $\mathbb{G}_1 \neq \mathbb{G}_2$ and no efficiently computable isomorphism is known from $\mathbb{G}_2$ to $\mathbb{G}_1$ (or from $\mathbb{G}_1$ to $\mathbb{G}_2$).

Note 2.2.1 *In the above definition, the groups order N can be either prime or composite. The group $\mathbb{G}_2$ is written $\mathbb{H}$ sometimes.*

Some works, such as [69], show that Type 3 pairing offers better performance and flexibility. Therefore it this thesis we will try as much as possible to use Type 3 bilinear pairing, when instantiating our cryptographic schemes in the bilinear pairing setting.

Definition 2.2.2 ***Asymmetric bilinear pairing groups*** *[79, 85].* Asymmetric bilinear pairing groups *(of prime order)* $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ *are tuple of prime* p*, cyclic groups* $\mathbb{G}_1, \mathbb{G}_2$*,* $\mathbb{G}_T$ *of order* p*,* $g_1 \in \mathbb{G}_1$*,* $g_2 \in \mathbb{G}_2$ *generators of respectively* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*, and a bilinear pairing* $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$*. An asymmetric bilinear pairing group also possesses efficient algorithms, called generic group operations, for computing group operations, evaluating the bilinear pairing, deciding membership of the groups, equality of group elements and sampling generators of the groups.*

Note 2.2.2 *From this definition, we can easily derived the definition of asymmetric bilinear pairing groups of composite order, and symmetric bilinear pairing groups of prime or composite order.*

In the following, we denote by $\mathcal{G}$ a generator of (a)symmetric bilinear pairing groups. Such generator takes as input a security parameter 1^λ. For short, we will sometimes used the term *bilinear pairing group* to denote (a)symmetric bilinear pairing group for prime (resp. composite) order.

We now present some properties and notations about bilinear pairing groups.

Property 2.2.1 *[90] Let* $\Gamma = (N, \mathbb{G}, \mathbb{G}_T, g, e)$ *be a symmetric bilinear pairing group of composite order. Let* $N = p_1 p_2 \cdots p_m$*, where* $p_1, p_2, \cdots, p_m$ *are distinct primes. For each* p_i*,* $\mathbb{G}$ *has a subgroup of order* p_i *denoted by* $\mathbb{G}_{p_i}$*. We let* $g_1, \cdots, g_m$ *denote generators of* $\mathbb{G}_{p_1}$ *through* $\mathbb{G}_{p_m}$ *respectively. Each element* $g \in \mathbb{G}$ *can be expressed as* $g = g_1^{a_1} g_2^{a_2} \cdots g_m^{a_m}$ *for some* $a_1, \cdots, a_m \in \mathbb{Z}_N$*, where each* a_i *is unique modulo* p_i*. We will refer to* $g_1^{a_i}$ *as the "*$\mathbb{G}_{p_i}$ *component" of* g*. When* a_i *is congruent to zero modulo* p_i*, we say that* g *has no* $\mathbb{G}_{p_i}$ *component. The subgroups* $\mathbb{G}_{p_1}, \cdots, \mathbb{G}_{p_m}$ *are "orthogonal" under the bilinear map* e*, meaning that if* $h \in \mathbb{G}_{p_i}$ *and* $u \in \mathbb{G}_{p_j}$ *for* $i \neq j$*, then* $e(h, u) = \mathbb{1}_{\mathbb{G}_T}$*, where* $\mathbb{1}_{\mathbb{G}_T}$ *denotes the identity element in* $\mathbb{G}_T$*.*

Definition 2.2.3 ***Canceling bilinear maps*** *[90]. We say that a bilinear map* $e : \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$ *is* canceling *if there are subgroups* $\mathbb{G}_1, \cdots, \mathbb{G}_m$ *of* $\mathbb{G}$ *and* $\mathbb{H}_1, \cdots, \mathbb{H}_m$ *of* $\mathbb{H}$ *such that* $\mathbb{G} \cong \mathbb{G}_1 \times \cdots \times \mathbb{G}_m$*,* $\mathbb{H} \cong \mathbb{H}_1 \times \cdots \times \mathbb{H}_m$*, and* $e(g_i, h_j) = \mathbb{1}_T$ *whenever* $g_i \in \mathbb{G}_i, h_j \in \mathbb{H}_j$ *for* $i \neq j$*.*

This structure is achieved naturally when the groups $\mathbb{G}, \mathbb{H}, \mathbb{G}_T$ are of composite order $N = p_1 p_2 \cdots p_m$ (where $p_1, \cdots, p_m$ are distinct primes), since we may set $\mathbb{G}_i = \mathbb{G}_{p_i}$, $\mathbb{H}_i = \mathbb{H}_{p_i}$ to be the subgroups of order p_i for each i.

Notation 2.2.1 *In this work, we will consider individual elements of $\mathbb{G}_1$ or $\mathbb{G}_2$ but also "vectors" of group elements. For any vector $\boldsymbol{v} = (v_1, \cdots, v_l) \in \mathbb{Z}_p^l$ and $g_\beta \in \mathbb{G}_\beta$, $g_\beta^{\boldsymbol{v}}$ denote a l-tuple of elements of $\mathbb{G}_\beta$, for $\beta = 1, 2$:*

$$g_\beta^{\boldsymbol{v}} := (g_\beta^{v_1}, \cdots, g_\beta^{v_l})$$

For any $a \in \mathbb{Z}_p$ and $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{Z}_p^l$, we have:

$$g_\beta^{a\boldsymbol{v}} := (g_\beta^{av_1}, \cdots, g_\beta^{av_l}), \qquad g_\beta^{\boldsymbol{v}+\boldsymbol{u}} := (g_\beta^{v_1+u_1}, \cdots, g_\beta^{v_l+u_l})$$

Then we define

$$e(g_1^{\boldsymbol{v}}, g_2^{\boldsymbol{u}}) := \prod_{i=1}^{l} e(g_1^{v_i}, g_2^{u_i}) = e(g_1, g_2)^{\boldsymbol{v}\cdot\boldsymbol{u}}.$$

Here all the computations are done modulo p.

2.2.2 Security Assumptions and Problems

In this section we present bilinear pairing related security problems and assumptions that we will use in this thesis.

Definition 2.2.4 ***Decisional Diffie-Hellman problem in $\mathbb{G}_1$ (DDH$_1$) [30].*** *Let $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ be an asymmetric bilinear pairing group of prime order and a, b, c be randoms in $\mathbb{Z}_p$. The DDH$_1$ problem consists, given as input $\Delta = (\Gamma, g_1, g_2, g_1^a, g_2^b)$ and t, in deciding if $t = g_1^{ab}$ or $t = g_1^{ab+c}$. An adversary $\mathcal{A}$ solves the DDH problem in $\mathbb{G}_1$ with advantage ϵ if*

$$\mathrm{Adv}_{\mathcal{A}}^{DDH_1} := \left|\Pr\left[\mathcal{A}(\Delta, g_1^{ab}) = 1\right] - \Pr\left[\mathcal{A}(D, g_1^{ab+c}) = 1\right]\right| \geq \epsilon$$

where the probability is taken over the random choice of generators $(g_1, g_2) \in \mathbb{G}_1 \times \mathbb{G}_2$, the random choice of $a, b, c \in \mathbb{Z}_p$ and the random bits consumed by $\mathcal{A}$.

The dual of above problem is Decisional Diffie-Hellman problem in $\mathbb{G}_2$ (denoted as DDH$_2$), which is identical to DDH$_1$ with the roles of $\mathbb{G}_1$ and $\mathbb{G}_2$ reversed.

Definition 2.2.5 ***Computational Diffie-Hellman problem in*** $\mathbb{G}_2$ *(CDH). Let* $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ *be an asymmetric bilinear pairing group of prime order and* a, b *be randoms in* $\mathbb{Z}_p$*. The CDH problem in* $\mathbb{G}_2$ *consists, given as input* (Γ, g_2^a, g_2^b)*, in computing* g_2^{ab}*. An adversary* $\mathcal{A}$ *wins the CDH problem in* $\mathbb{G}_2$ *with advantage* ϵ *if*

$$\mathsf{Adv}_{\mathcal{A}}^{CDH} := \Pr\left[\mathcal{A}(\Gamma, g_2^a, g_2^b) = g_2^{ab}\right] \geq \epsilon$$

where the probability is taken over the random choice of generators $(g_1, g_2) \in \mathbb{G}_1 \times \mathbb{G}_2$*, the random choice of* $a, b \in \mathbb{Z}_p$ *and the random bits consumed by* $\mathcal{A}$*.*

Definition 2.2.6 ***Symmetric External Diffie-Hellman assumption*** *(SXDH) [34]. The SXDH assumption holds if DDH problems are intractable in both* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*.*

Definition 2.2.7 ***Decisional Linear problem*** *(DLin) [34]. Let* $\Gamma = (p, \mathbb{G}, \mathbb{G}_T, g, e)$ *be a symmetric bilinear pairing group of prime order,* u, v, h *be arbitrary generators of* $\mathbb{G}$ *and* a, b, c *be randoms in* $\mathbb{Z}_p$*. The DLin problem consists, given as input* $\Delta = (\Gamma, u, v, h, u^a, v^b)$ *and* t*, in deciding if* $t = h^{a+b}$ *or* $t = h^c$*. An adversary* $\mathcal{A}$ *solves the DLin problem with advantage* ϵ *if*

$$\mathsf{Adv}_{\mathcal{A}}^{DLin} := \left|\Pr\left[\mathcal{A}(\Delta, h^{a+b}) = 1\right] - \Pr\left[\mathcal{A}(\Delta, h^c) = 1\right]\right| \geq \epsilon$$

where the probability is taken over the random choice of generators $g, u, v, h \in \mathbb{G}$*, the random choice of* $a, b, c \in \mathbb{Z}_p$ *and the random bits consumed by* $\mathcal{A}$*.*

Definition 2.2.8 ***eXternal Decision Linear 1 problem*** *(XDLin1) [6]. Let* $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ *be an asymmetric bilinear pairing group group of prime order and* x, y, a, b, c *be randoms in* $\mathbb{Z}_p$*. The XDLin*1 *problem consists, given as input a tuple* $\Delta = (g_1, g_1^x, g_1^y, g_1^{ax}, g_1^{by}, g_2, g_2^x, g_2^y, g_2^{ax}, g_2^{by})$ *and* t*, in deciding if* $t = g_1^{a+b}$ *or* $t = g_1^c$*. An adversary* $\mathcal{A}$ *solves the XDLin*1 *problem with advantage* ϵ *if*

$$\mathsf{Adv}_{\mathcal{A}}^{XDLin1} := \left|\Pr\left[\mathcal{A}(\Delta, g_1^{a+b}) = 1\right] - \Pr\left[\mathcal{A}(\Delta, g_1^c) = 1\right]\right| \geq \epsilon$$

where the probability is taken over the random choice of generators $(g_1, g_2) \in \mathbb{G}_1 \times \mathbb{G}_2$*, the random choice of* x, y, a, b *in* $\mathbb{Z}_p$ *and the random bits consumed by* $\mathcal{A}$*.*

The **eXternal Decision Linear 2 Assumption** (XDLin2) is defined similarly, except that t is equal either to g_2^{a+b}, or to g_2^c.

Definition 2.2.9 ***ℓ-Bilinear Diffie-Hellman Exponent problem*** *(ℓ-BDHE) [33]. Let $\Gamma = (p, \mathbb{G}, \mathbb{G}_T, g, e)$ be a symmetric bilinear pairing group of prime order, h be a generator of $\mathbb{G}$ and α be a random in $\mathbb{Z}_p^*$. The ℓ-BDHE problem consists, given $(g, h, g^{\alpha}, g^{\alpha^2}, \cdots, g^{\alpha^{l-1}}, g^{\alpha^{l+1}}, \cdots, g^{\alpha^{2l}})$, in computing $e(g,h)^{\alpha^l}$. An adversary $\mathcal{A}$ solves the ℓ-BDHE problem with advantage ϵ if*

$$\mathsf{Adv}_{\mathcal{A}}^{\ell\text{-}BDHE} := \Pr\left[\mathcal{A}(g, h, g^{\alpha}, g^{\alpha^2}, \cdots, g^{\alpha^{l-1}}, g^{\alpha^{l+1}}, \cdots, g^{\alpha^{2l}}) = e(g,h)^{\alpha^l}\right] \geq \epsilon$$

where the probability is taken over the random choice of generators g, h in $\mathbb{G}$, the random choice of α in $\mathbb{Z}_p^*$, and the random bits used by $\mathcal{A}$.

Definition 2.2.10 ***q-strong Diffie-Hellman Problem*** *(q-SDH) [32]. Let $\mathbb{G}_1, \mathbb{G}_2$ be two cyclic groups of prime order p, where possibly $\mathbb{G}_1 = \mathbb{G}_2$. Let g_1 be a generator of $\mathbb{G}_1$, g_2 a generator of $\mathbb{G}_2$ and x be random in $\mathbb{Z}_p^*$. The q-SDH problem consists, given a $(q+3)$-tuple $(g_1, g_1^x, g_1^{x^2}, \cdots, g_1^{x^q}, g_2, g_2^x) \in \mathbb{G}_1^{q+1} \times \mathbb{G}_2^2$, in computing a pair $(g_1^{1/(x+c)}, c) \in \mathbb{G}_1 \times \mathbb{Z}_p$. An adversary $\mathcal{A}$ solves the q-SDH problem with advantage ϵ if*

$$\mathsf{Adv}_{\mathcal{A}}^{q-SDH} := \Pr\left[\mathcal{A}(g_1, g_1^x, \cdots, g_1^{x^q}, g_2, g_2^x) = (g_1^{1/(x+c)}, c)\right] \geq \epsilon$$

where the probability is taken over the random choice of generators $(g_1, g_2) \in \mathbb{G}_1 \times \mathbb{G}_2$, the random choice of $x \in \mathbb{Z}_p^*$, and the random bits consumed by $\mathcal{A}$.

Note that when $\mathbb{G}_1 = \mathbb{G}_2$, the pair (g_2, g_2^x) is redundant since in that case this pair can be generated by raising (g_1, g_1^x) to a random power.

Note 2.2.3 *This problem is a stronger version of the original q-SDH problem, introduced by Boneh and Boyen in 2004 [31]. The reduction of the 2004 version to the 2008 version can be done easily, as explained in [133].*

We now present a modified version of this problem, introduced in 2016 by Ghosh *et al.* [73].

Definition 2.2.11 ***(symmetric) q-strong Bilinear Diffie Hellman problem*** *(q-sSBDH) [73]. Let $\Gamma = (p, \mathbb{G}, \mathbb{G}_T, g, e)$ be a symmetric bilinear pairing group of prime order and x be random in $\mathbb{Z}_p^*$. The q-sSBDH problem consists, given as input a $(q+1)$-tuple of elements $(g, g^x, g^{(x^2)}, \cdots, g^{(x^q)}) \in \mathbb{G}^{q+1}$, in computing a pair $(c, e(g,g)^{1/(x+c)}) \in \mathbb{Z}_p \times \mathbb{G}_T$ for a freely chosen value $c \in \mathbb{Z}_p \setminus \{-x\}$. An adversary $\mathcal{A}$ solves the q-sSBDH problem*

with advantage ϵ if

$$\mathsf{Adv}_{\mathcal{A}}^{q-sSBDH} := \Pr\left[\mathcal{A}(g, g^x, g^{(x^2)}, \cdots, g^{(x^q)}) = (c, e(g,g)^{1/(x+c)})\right] \geq \epsilon$$

where the probability is taken over the random choice of generator $g \in \mathbb{G}_1$, the random choice of $x \in \mathbb{Z}_p^*$, and the random bits consumed by $\mathcal{A}$.

In this thesis, we will use the asymmetric different version of the above problem, that we present below.

Definition 2.2.12 ***(asymmetric) q-strong Bilinear Diffie Hellman problem*** *(q-SBDH) . Let $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, p)$ be an asymmetric bilinear pairing group of prime order and x be random in $\mathbb{Z}_p^*$. The q-SBDH problem consists, given as input a $(2q+2)$-tuple of elements $(g_1, g_1^x, g_1^{(x^2)}, \cdots, g_1^{(x^q)}, g_2, g_2^x, g_2^{(x^2)}, \cdots, g_2^{(x^q)}) \in \mathbb{G}_1^{q+1} \times \mathbb{G}_2^{q+1}$, in computing a pair $(c, e(g_1, g_2)^{1/(x+c)}) \in \mathbb{Z}_p \times \mathbb{G}_T$ for a freely chosen value $c \in \mathbb{Z}_p \setminus \{-x\}$. An adversary $\mathcal{A}$ solves the q-SBDH problem with advantage ϵ if*

$$\mathsf{Adv}_{\mathcal{A}}^{q-SBDH} := \Pr\left[\begin{array}{c}\mathcal{A}(g_1, g_1^x, g_1^{(x^2)}, \cdots, g_1^{(x^q)}, g_2, g_2^x, g_2^{(x^2)}, \cdots, g_2^{(x^q)}) \\ = (c, e(g_1, g_2)^{1/(x+c)})\end{array}\right] \geq \epsilon$$

where the probability is taken over the random choice of generators $(g_1, g_2) \in \mathbb{G}_1 \times \mathbb{G}_2$, the random choice of $x \in \mathbb{Z}_p^*$, and the random bits consumed by $\mathcal{A}$.

We easily see that the q-SBDH problem is reductible to the q-SDH problem: for a given q-SBDH tuple $(g_1, g_1^x, \cdots, g_1^{x^q}, g_2, g_2^x, \cdots, g_2^{x^q})$, one can create the truncated tuple $(g_1, g_1^x, \cdots, g_1^{x^q}, g_2, g_2^x)$ and inputs it to the oracle of q-SDH problem, to obtain $(g_1^{1/(x+c)}, c)$ and finally $(c, e(g_1^{1/(x+c)}, g_2)) = (c, e(g_1, g_2)^{1/(x+c)})$.

Definition 2.2.13 ***n-Extended Decisional Diffie-Hellman problem in $\mathbb{G}_1$*** *(n-eDDH) [91]. Let $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ be an asymmetric bilinear pairing group, κ be a random in $\mathbb{Z}_p^*$ and $\omega, \{h_i, \gamma_i\}_{i=1}^n$ be randoms in $\mathbb{Z}_p$. The n-eDDH problem consists, given as input $\Delta = (\Gamma, g_1^\kappa, \left\{g_1^{\omega+\gamma_i h_i}, g_1^{\gamma_i}, g_1^{h_i}\right\}_{i=1}^n)$ and t in deciding if $t = g_1^{\kappa\omega}$ or t is a random element y of $\mathbb{G}_1$. A PPT adversary $\mathcal{A}$ solves the DDH problem in $\mathbb{G}$ with advantage ϵ if*

$$\mathsf{Adv}_{\mathcal{A}}^{n-eDDH} := \left|\Pr\left[\mathcal{A}(\Delta, g_1^{\kappa\omega})\right] - \Pr\left[\mathcal{A}(\Delta, y)\right]\right| \geq \epsilon$$

where the probability is taken over the random choice of generator $g_1 \in \mathbb{G}_1$, $\kappa \in \mathbb{Z}_p^*$, $\omega, \{h_i, \gamma_i\}_{i=1}^n \in \mathbb{Z}_p$ and the random bits consumed by $\mathcal{A}$.

Definition 2.2.14 ***General Subgroup Decision problem*** *(GSD) [26, 90]. Let* $S_0, S_1, S_2, \cdots, S_k$ *be non-empty subsets of* $[m]$ *such that for each* $2 \leq j \leq k$*, either* $S_j \cap S_0 = \varnothing = S_j \cap S_1$ *or* $S_j \cap S_0 \neq \varnothing \neq S_j \cap S_1$*. Let* $\Gamma = (N = p_1 \cdots p_m, \mathbb{G}, \mathbb{G}_T, e$ *be a symmetric bilinear pairing composite order group and randoms* $Z_0 \leftarrow \mathbb{G}_{S_0}$*,* $Z_1 \leftarrow \mathbb{G}_{S_1}$*,* $Z_2 \leftarrow \mathbb{G}_{S_2}, \cdots, Z_k \leftarrow \mathbb{G}_{S_k}$*. The general subgroup decision problem consists, given* $\Delta = (\Gamma, Z_2, \cdots, Z_k)$ *and* t*, in deciding if* $t = Z_0$ *or* $t = Z_1$*. An adversary* $\mathcal{A}$ *solves the general subgroup decision problem with advantage* ϵ *if*

$$\mathsf{Adv}_{\mathcal{A}}^{GSD} := |\Pr[\mathcal{A}(\Delta, Z_0) = 1] - \Pr[\mathcal{A}(\Delta, Z_1) = 1]| \geq \epsilon$$

where the probability is taken over the random choice of $Z_0, \cdots, Z_k$ *and the random bits consumed by* $\mathcal{A}$*.*

2.3 Dual Pairing Vector Spaces

Dual Pairing Vector Spaces is a concept of pairing-based cryptography, introduced by Okamoto and Takashima [118, 117] that we will use in this thesis to build our cryptographic schemes and prove their security. In this section we present this concept along with its properties and security assumptions.

2.3.1 Definitions and Properties

Definition 2.3.1 ***Dual Pairing Vector Spaces*** *(DPVS) [117]. Dual pairing vector spaces* $(p, \mathbb{V}, \mathbb{V}^*, \mathbb{G}_T, \mathbb{A}, \mathbb{A}^*)$ *are a tuple of a prime* p*, two* N*-dimensional vector spaces* $\mathbb{V}, \mathbb{V}^*$ *over* $\mathbb{Z}_p$*, a cyclic group* $\mathbb{G}_T$ *of order* p*, and their canonical bases i.e.,* $\mathbb{A} = (\boldsymbol{a}_1, \cdots, \boldsymbol{a}_N)$ *of* $\mathbb{V}$ *and* $\mathbb{A}^* = (\boldsymbol{a}_1^*, \cdots, \boldsymbol{a}_N^*)$ *of* $\mathbb{V}^*$ *that satisfy the following conditions:*

- *Non-degenerate bilinear pairing: there exists a polynomial-time computable non-degenerate bilinear pairing* $e : \mathbb{V} \times \mathbb{V}^* \to \mathbb{G}_T$ *i.e.,* $e(s\boldsymbol{x}, t\boldsymbol{y}) = e(\boldsymbol{x}, \boldsymbol{y})^{st}$ *and if* $e(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{1}_T$ *for all* $\boldsymbol{y} \in \mathbb{V}^*$*, then* $\boldsymbol{x} = \boldsymbol{0}$*.*
- *Dual orthonormal bases:* $\mathbb{A}, \mathbb{A}^*$*, and* e *satisfy* $e(\boldsymbol{a}_i, \boldsymbol{a}_j^*) = g_T^{\delta_{i,j}}$ *for all* i *and* j*, where* $\delta_{i,j} = 1$ *if* $i = j$*, and* 0 *otherwise, and* $g_T \neq \mathbb{1}_T \in \mathbb{G}_T$*.*
- *Distorsion maps: endomorphisms* $\phi_{i,j}$ *of* $\mathbb{V}$ *such that* $\phi_{i,j}(\boldsymbol{a}_j) = \boldsymbol{a}_i$ *and* $\phi_{i,j}(\boldsymbol{a}_k) = \boldsymbol{0}$ *if* $k \neq j$ *are polynomial-time computable. Moreover, endomorphisms* $\phi_{i,j}^*$ *of* $\mathbb{V}^*$ *such that* $\phi_{i,j}(\boldsymbol{a}_j^*) = \boldsymbol{a}_i^*$ *and* $\phi_{i,j}(\boldsymbol{a}_k^*) = \boldsymbol{0}$ *if* $k \neq j$ *are also polynomial-time computable. We call* $\phi_{i,j}$ *and* $\phi_{i,j}^*$ *"distortion maps".*

We now present a typical construction of dual pairing vector space as a product of bilinear pairing group, that was first presented in [118]. Let $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ be an asymmetric bilinear pairing group of prime order. The construction (where the description of distortion maps is omitted) is the following:

- **Vector spaces $\mathbb{V}$ and $\mathbb{V}^*$**: $\mathbb{V} := \overbrace{\mathbb{G}_1 \times \cdots \times \mathbb{G}_1}^{N}$ and $\mathbb{V}^* := \overbrace{\mathbb{G}_2 \times \cdots \times \mathbb{G}_2}^{N}$, whose elements are expressed by N-dimensional vectors, $\boldsymbol{x} := (g_1^{x_1}, \cdots, g_1^{x_N})$ and $\boldsymbol{y} := (g_2^{y_1}, \cdots, g_2^{y_N})$, respectively ($x_i, y_i \in \mathbb{F}_q$ for $i = 1, \cdots, N$).
- **Canonical bases $\mathbb{A}$ and $\mathbb{A}^*$**: $\mathbb{A} := (\boldsymbol{a}_1, \cdots, \boldsymbol{a}_N)$ of $\mathbb{V}$, where $\boldsymbol{a}_1 := (g_1, 1, \cdots, 1)$, $\boldsymbol{a}_2 := (1, g_1, 1, \cdots, 1)$, $\cdots$, $\boldsymbol{a}_N := (1, \cdots, 1, g_1)$. $\mathbb{A}^* := (\boldsymbol{a}_1^*, \cdots, \boldsymbol{a}_N^*)$ of $\mathbb{V}^*$, where $\boldsymbol{a}_1^* := (g_2, 1, \cdots, 1)$, $\boldsymbol{a}_2^* := (1, g_2, 1, \cdots, 1)$, $\cdots$, $\boldsymbol{a}_N^* := (1, \cdots, 1, g_2)$.
- **Pairing operation**: $e(\boldsymbol{x}, \boldsymbol{y}) := \prod_{i=1}^{N} e(g_1^{x_i}, g_2^{y_i}) = e(g_1, g_2)^{\sum_{i=1}^{N} x_i y_i} = e(g_1, g_2)^{\boldsymbol{x} \cdot \boldsymbol{y}} \in \mathbb{G}_T$ for the above $\boldsymbol{x} \in \mathbb{V}$ and $\boldsymbol{y} \in \mathbb{V}^*$.
- **Base change**: canonical basis $\mathbb{A}$ is changed to basis $\mathbb{B} := (\boldsymbol{b}_1, \cdots, \boldsymbol{b}_N)$ of $\mathbb{V}$ using a uniformly chosen (regular) linear transformation, $X := (\chi_{i,j}) \leftarrow GL(N, \mathbb{F}_p)$, such that $\boldsymbol{b}_i = \sum_{j=1}^{N} \chi_{i,j} \boldsymbol{a}_j$, $(i = 1, \cdots, N)$. $\mathbb{A}^*$ is also changed to basis $\mathbb{B}^* := (\boldsymbol{b}_1^*, \cdots, \boldsymbol{b}_N^*)$ of $\mathbb{V}^*$, such that $(\vartheta)_{i,j} := (X^\top)^{-1}$, $\boldsymbol{b}_i^* = \sum_{j=1}^{N} \vartheta_{i,j} \boldsymbol{a}_j^*$, $(i = 1, \cdots, N)$. We see that $e(\boldsymbol{b}_i, \boldsymbol{b}_j^*) = e(g_1, g_2)^{\delta_{i,j}}$, ($\delta_{i,j} = 1$ if $i = j$, and $\delta_{i,j} = 0$ if $i \neq j$), i.e. $\mathbb{B}$ and $\mathbb{B}^*$ are dual orthonormal basis of $\mathbb{V}$ and $\mathbb{V}^*$.
- **Intractable Problem**: one of the most natural *decisional* problems is the *decisional subspace problem* (DSP). The DSP_{N_1,N_2} assumption is: it is hard to distinguish $\boldsymbol{v} := v_{N_2+1}\boldsymbol{b}_{N_2+1} + \cdots + v_{N_1}\boldsymbol{b}_{N_1}$ from $\boldsymbol{u} := (v_1\boldsymbol{b}_1 + \cdots + v_{N_1}\boldsymbol{b}_{N_1})$, where $(v_1, \cdots, v_{N_1}) \leftarrow \mathbb{F}_q^{N_1}$ and $N_2 + 1 < N_1$. DSP is intractable if the generalized DDH or DLin problems are intractable.
- **Trapdoor**: although the DSP problem is assumed to be intractable, it can be efficiently solve by using trapdoor $\boldsymbol{t}^* \in \mathrm{span}\langle \boldsymbol{b}_1^*, \cdots, \boldsymbol{b}_{N_2}^* \rangle$. Given $\boldsymbol{v} := v_{N_2+1}\boldsymbol{b}_{N_2+1} + \cdots + v_{N_1}\boldsymbol{b}_{N_1}$ from $\boldsymbol{u} := (v_1\boldsymbol{b}_1 + \cdots + v_{N_1}\boldsymbol{b}_{N_1})$, we can tell $\boldsymbol{v}$ from $\boldsymbol{u}$ using $\boldsymbol{t}^*$ since $e(\boldsymbol{v}, \boldsymbol{t}^*) = 1$ and $e(\boldsymbol{u}, \boldsymbol{t}^*) \neq 1$ with high probability.

In our work, for simplicity of the reading, we will use the following (simplified) definition of DPVS.

Definition 2.3.2 ***Dual pairing vector spaces*** *(DPVS) [51]. For a prime p and a fixed (constant) dimension n, two random bases $\mathbb{B} = (\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n)$ and $\mathbb{B}^* = (\boldsymbol{b}_1^*, \cdots, \boldsymbol{b}_n^*)$ of $\mathbb{Z}_p^n$ are said to be* dual orthonormal*, if $\boldsymbol{b}_i \cdot \boldsymbol{b}_j^* = 0 \pmod p$ whenever $i \neq j$, and $\boldsymbol{b}_i \cdot \boldsymbol{b}_i^* = \psi \pmod p$ for all i, where ψ is a uniformly random element of $\mathbb{Z}_p^*$. For generators $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$, notice that $e(g_1^{\boldsymbol{b}_i}, g_2^{\boldsymbol{b}_j^*}) = 1$ whenever $i \neq j$, where 1 here denotes the identity element in $\mathbb{G}_T$. We denote by* $\mathrm{Dual}(\mathbb{Z}_p^n)$ *an algorithm that generates dual pairing vector*

spaces.

For $n_1, \cdots, n_d$ fixed (constant) dimension, d tuples of two random bases $\mathbb{B}_i, \mathbb{B}_i^*$ of $\mathbb{Z}_p^{n_i}$ for $i = 1, \cdots, d$, are said to be dual orthonormal *if $\boldsymbol{b}_{j,i} \cdot \boldsymbol{b}_{k,i}^* = 0 \mod p$ whenever $j \neq k$, and $\boldsymbol{b}_{j,i} \cdot \boldsymbol{b}_{j,i}^* = \psi \mod p$ for all j and i, where ψ is a random element of $\mathbb{Z}_p^*$. A generation algorithm of such tuples is denoted by* $\mathsf{Dual}(\mathbb{Z}_p^{n_1}, \cdots, \mathbb{Z}_p^{n_d})$.

Note 2.3.1 *In the above definition, one can choose, for convenience, to use $\psi = 1$ as in the work of Lewko [89]. In our work, we do not enforce the condition $\psi = 1$, though it remains a possibility.*

We now present a way to produce new dual orthonormal bases from randomly sampled dual orthonormal bases. This construction was presented by Lewko in [90].

Let $\mathbb{B}, \mathbb{B}^*$ be dual orthonormal bases, $m \leq n$ be fixed positive integers and $\boldsymbol{A} \in \mathbb{Z}_p^{m \times m}$ be an invertible matrix. We let $S_m \subseteq [n]$ be a subset of size m, and define new dual orthonormal bases $\mathbb{B}_{\boldsymbol{A}}, \mathbb{B}_{\boldsymbol{A}^*}$ as follows. We let $\boldsymbol{B}_m$ denote the $n \times m$ matrix over $\mathbb{Z}_p$ whose columns are the vectors $\boldsymbol{b}_i \in \mathbb{B}$ such that $i \in S_m$. Then $\boldsymbol{B}_m \boldsymbol{A}$ is also an $n \times m$ matrix. We form $\mathbb{B}_{\boldsymbol{A}}$ by retaining all of the vectors $\boldsymbol{b}_i \in \mathbb{B}$ for $i \notin S_m$ and exchanging the $\boldsymbol{b}_i$ for $i \in S_m$ with the columns of $\boldsymbol{B}_m \boldsymbol{A}$. To define $\mathbb{B}_{\boldsymbol{A}}^*$, we similarly let $\boldsymbol{B}^*$ denote the $n \times m$ matrix over $\mathbb{Z}_p$ whose columns are the vectors of $\boldsymbol{b}_i^* \in \mathbb{B}^*$ such that $i \in S_m$. Then $\boldsymbol{B}_m^* (\boldsymbol{A}^{-1})^\top$ is also an $n \times m$ matrix, where $(\boldsymbol{A}^{-1})^\top$ denotes the transpose of $\boldsymbol{A}^{-1}$. We form $\mathbb{B}_{\boldsymbol{A}}^*$ by retaining all of the vector $\boldsymbol{b}_i^* \in \mathbb{B}^*$ for $i \notin S_m$ and exchanging the $\boldsymbol{b}_i^*$ for $i \in S_m$ with the columns of $\boldsymbol{B}_m^* (\boldsymbol{A}^{-1})^\top$.

The following lemma formalized the fact that the above constructed bases $(\mathbb{B}_{\boldsymbol{A}}, \mathbb{B}_{\boldsymbol{A}}^*)$ are dual orthonormal.

Lemma 2.3.1 *[90] For any fixed positive integers $m \leq n$, any fixed invertible matrix $\boldsymbol{A} \in \mathbb{Z}_p^{m \times m}$ and set $S_m \subseteq [n]$ of size m, if $(\mathbb{B}, \mathbb{B}^*) \leftarrow \mathsf{Dual}(\mathbb{Z}_p^d)$, then $(\mathbb{B}_{\boldsymbol{A}}, \mathbb{B}_{\boldsymbol{A}^*})$ is also distributed as a random sample from $\mathsf{Dual}(\mathbb{Z}_p^d)$. In particular, the distribution of $(\mathbb{B}_{\boldsymbol{A}}, \mathbb{B}_{\boldsymbol{A}^*})$ is independent of $\boldsymbol{A}$.*

Definition 2.3.3 ***Parameter hiding*** *[90]. In the above construction, if the distribution of the final bases $(\mathbb{B}_{\boldsymbol{A}}, \mathbb{B}_{\boldsymbol{A}^*})$ reveals nothing about the matrix $\boldsymbol{A}$ employed, we say that the construction is* parameter hiding.

Note 2.3.2 *Dual orthonormal bases* $(\mathbb{B}, \mathbb{B}^*)$ *in prime order groups achieved canceling bilinear maps. Each subgroup* $\mathbb{G}_i$ *corresponds to the span of vector* $\boldsymbol{b}_i$ *in the exponent of group* $\mathbb{G}_1$*, and each subgroup* $\mathbb{H}_i$ *corresponds to the span of the vector* $\boldsymbol{b}_i^*$ *in the exponent of group* $\mathbb{G}_2$*.*

2.3.2 Security Assumptions and Problems

In this subsection, we introduce dual pairing vector spaces security assumptions and problems that we will use in our security proofs.

Definition 2.3.4 ***Decisional subspace problem in*** $\mathbb{G}_1$ *(DS1) [51]. Let* $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ *be an asymmetric bilinear pairing group of prime order,* k, n *be fixed positive integers that satisfy* $2k \leq n$*,* $(\mathbb{B}, \mathbb{B}^*)$ *be two random dual orthonormal bases of* $\mathbb{Z}_p^n$ *and* $\tau_1, \tau_2, \mu_1, \mu_2$ *be randoms of* $\mathbb{Z}_p$*. Define the following elements*

$$\begin{gathered}\boldsymbol{u}_1 = g_2^{\mu_1.\boldsymbol{b}_1^* + \mu_2.\boldsymbol{b}_{k+1}^*}, \cdots, \boldsymbol{u}_k = g_2^{\mu_1.\boldsymbol{b}_k^* + \mu_2 \boldsymbol{b}_{2k}^*}, \\ \boldsymbol{v}_1 = g_1^{\tau_1.\boldsymbol{b}_1}, \cdots, \boldsymbol{v}_k = g_1^{\tau_1.\boldsymbol{b}_k}, \\ \boldsymbol{w}_1 = g_1^{\tau_1.\boldsymbol{b}_1 + \tau_2 \boldsymbol{b}_{k+1}}, \cdots, \boldsymbol{w}_k = g_1^{\tau_1.\boldsymbol{b}_k + \mu_2 \boldsymbol{b}_{2k}}.\end{gathered}$$

The decisional subspace problem in $\mathbb{G}_1$ *consists, given tuple* $\Delta = (\Gamma, g_2^{\boldsymbol{b}_1^*}, \cdots, g_2^{\boldsymbol{b}_k^*}, g_2^{\boldsymbol{b}_{2k+1}^*}, \cdots, g_2^{\boldsymbol{b}_n^*}, g_1^{\boldsymbol{b}_1}, \cdots, g_1^{\boldsymbol{b}_n}, \boldsymbol{u}_1, \cdots, \boldsymbol{u}_k, \mu_2)$ *and* $(\boldsymbol{t}_1, \cdots, \boldsymbol{t}_k)$*, in deciding if* $(\boldsymbol{t}_1, \cdots, \boldsymbol{t}_k) = (\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k)$ *or* $(\boldsymbol{t}_1, \cdots, \boldsymbol{t}_k) = (\boldsymbol{w}_1, \cdots, \boldsymbol{w}_k)$*. An adversary* $\mathcal{A}$ *solves the decisional subspace problem in* $\mathbb{G}_1$ *with advantage* ϵ *if*

$$\mathrm{Adv}_{\mathcal{A}}^{DS1} := \left|\Pr\left[\mathcal{A}(\Delta, \boldsymbol{v}_1, \cdots, \boldsymbol{v}_k) = 1\right] - \Pr\left[\mathcal{A}(D, \boldsymbol{w}_1, \cdots, \boldsymbol{w}_k) = 1\right]\right| \geq \epsilon$$

where the probability is taken over the random choice of generators $(g_1, g_2) \in \mathbb{G}_1 \times \mathbb{G}_2$*, the random choice of* $\mathbb{B}, \mathbb{B}^*, \tau_1, \tau_2, \mu_1, \mu_2$ *and the random bits consumed by* $\mathcal{A}$*.*

Lemma 2.3.2 *If the decisional Diffie Hellman problem (DDH) in* $\mathbb{G}_1$ *holds, then the decisional subspace problem in* $\mathbb{G}_1$ *(DS1) also holds.*

The idea of the proof is that if there exists an adversary that breaks the **decisional subspace** problem in $\mathbb{G}_1$, then one can create an adversary against the DDH problem in $\mathbb{G}_1$. For more details on the proof, refer to [51]. The **decisional subspace** problem in $\mathbb{G}_2$ is defined as identical to DS1 with the roles of $\mathbb{G}_1$ and $\mathbb{G}_2$ reversed. DS2 holds if DDH in $\mathbb{G}_2$ holds. The proof is done as for $\mathbb{G}_1$.

We now present three problems, **Problem 1**, **Problem 2** and **Problem 3**, useful for our security proofs. These three problems were first defined in [114] for the symmetric bilinear setting, and proven to hold if DLin holds. We adapt them to the asymmetric bilinear setting, and prove that they hold if XDLin_1 and XDLin_2 hold.

Definition 2.3.5 ***Problem 1.*** *Let $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ be an asymmetric bilinear pairing group of prime order, $(\mathbb{D}, \mathbb{D}^*)$ be random dual orthonormal bases of $\mathbb{Z}_p^{4n+2}$, $\hat{\mathbb{D}}^* = (\boldsymbol{d}_0^*, \cdots, \boldsymbol{d}_n^*, \boldsymbol{d}_{2n+1}^*, \cdots, \boldsymbol{d}_{4n+1}^*)$ and ω, γ, z be randoms of $\mathbb{Z}_p$. For $i \in [\![2, n]\!]$ define the following elements*

$$\boldsymbol{e}_{0,1} = g_1^{\omega \boldsymbol{d}_1 + \gamma \boldsymbol{d}_{4n+1}}, \qquad \boldsymbol{e}_{1,1} = g_1^{\omega \boldsymbol{d}_1 + z \boldsymbol{d}_{n+1} + \gamma \boldsymbol{d}_{4n+1}}, \qquad \boldsymbol{e}_i = g_1^{\omega \boldsymbol{d}_i}.$$

Problem 1 consists, given $\Delta = (\Gamma, \mathbb{D}, \hat{\mathbb{D}}^*, \{\boldsymbol{e}_i\}_{i \in [\![2,n]\!]})$ and $\boldsymbol{t}$, in deciding if $\boldsymbol{t} = \boldsymbol{e}_{0,1}$ or $\boldsymbol{t} = \boldsymbol{e}_{1,1}$. An adversary $\mathcal{A}$ solves Problem 1 with advantage ϵ if

$$\mathsf{Adv}_{\mathcal{A}}^{P1} := |\Pr[\mathcal{A}(\Delta, \boldsymbol{e}_{0,1}) = 1] - \Pr[\mathcal{A}(\Delta, \boldsymbol{e}_{1,1}) = 1]| \geq \epsilon$$

where the probability is taken over the random choice of generator $g_1 \in \mathbb{G}_1$, random choice of $(\mathbb{D}, \mathbb{D}^*)$, ω, γ, z in $\mathbb{Z}_p$, and the random bits consumed by $\mathcal{A}$.

Lemma 2.3.3 *For any adversary $\mathcal{A}$, there is a probabilistic adversary $\mathcal{B}$, whose running time is essentially the same as that of $\mathcal{A}$, such that for any security parameter λ, $\mathsf{Adv}_{\mathcal{A}}^{P1}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{XDLin1}}(\lambda) + 5/p$.*

Definition 2.3.6 ***Problem 2.*** *Let $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ be an asymmetric bilinear pairing group of prime order, $(\mathbb{D}, \mathbb{D}^*)$ be random dual orthonormal bases of $\mathbb{Z}_p^{4n+2}$, $\hat{\mathbb{D}} = (\boldsymbol{d}_0, \cdots, \boldsymbol{d}_n, \boldsymbol{d}_{2n+1}, \cdots, \boldsymbol{d}_{4n+1})$ and $\delta, \tau, \delta_0, \omega, \sigma$ be random elements of $\mathbb{Z}_p$. For $i \in [n]$ define the following elements*

$$\boldsymbol{h}_{0,i}^* = g_2^{\delta \boldsymbol{d}_i^* + \delta_0 \boldsymbol{d}_{3n+i}^*}, \qquad \boldsymbol{h}_{1,i}^* = g_2^{\delta \boldsymbol{d}_i^* + \tau \boldsymbol{d}_{n+i}^* + \delta_0 \boldsymbol{d}_{3n+i}^*}, \qquad \boldsymbol{e}_i = g_1^{\omega \boldsymbol{d}_i + \sigma \boldsymbol{d}_{n+i}}.$$

Problem 2 consists, given $\Delta = (\Gamma, \hat{\mathbb{D}}, \mathbb{D}^*, \{\boldsymbol{e}_i\}_{i \in [n]})$ and $\{\boldsymbol{t}_\mathbb{B}\}_{i \in [n]}$, in deciding if $(\boldsymbol{t}_1, \cdots, \boldsymbol{t}_n) = (\boldsymbol{h}_{0,1}^*, \cdots, \boldsymbol{h}_{0,n}^*)$ or $(\boldsymbol{t}_1, \cdots, \boldsymbol{t}_n) = (\boldsymbol{h}_{1,1}^*, \cdots, \boldsymbol{h}_{1,n}^*)$. An adversary $\mathcal{A}$ solves Problem 2 with advantage ϵ if

$$\mathsf{Adv}_{\mathcal{A}}^{P2} := \left| \Pr\left[\mathcal{A}(\Delta, \{\boldsymbol{e}_{0,i}\}_{i \in [n]}) = 1\right] - \Pr\left[\mathcal{A}(\Delta, \{\boldsymbol{e}_{1,i}\}_{i \in [n]}) = 1\right] \right| \geq \epsilon$$

where the probability is taken over the random choice of generators $(g_1, g_2) \in \mathbb{G}_1 \times \mathbb{G}_2$, random choice of $(\mathbb{D}, \mathbb{D}^*)$, $\delta, \tau, \delta_0, \omega, \sigma$ in $\mathbb{Z}_p$, and the random bits consumed by $\mathcal{A}$.

Lemma 2.3.4 *For any adversary $\mathcal{A}$, there is a probabilistic adversary $\mathcal{B}$, whose running time is essentially the same as that of $\mathcal{A}$, such that for any security parameter λ,* $\mathsf{Adv}^{P2}_{\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{XDLin2}}_{\mathcal{B}}(\lambda) + 5/p$.

Definition 2.3.7 ***Problem 3.*** *Let $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ be an asymmetric bilinear pairing group of prime order, $(\mathbb{D}, \mathbb{D}^*)$ be two random dual orthonormal bases of $\mathbb{Z}_p^{4n+2}$, $\hat{\mathbb{D}} = (\boldsymbol{d}_0, \cdots, \boldsymbol{d}_n, \boldsymbol{d}_{2n+1}, \cdots, \boldsymbol{d}_{4n+1})$, $\hat{\mathbb{D}}^* = (\boldsymbol{d}^*_0, \cdots, \boldsymbol{d}^*_n, \boldsymbol{d}^*_{2n+1}, \cdots, \boldsymbol{d}^*_{4n+1})$ and $\tau, \delta_0, \omega, \omega', \omega'', \kappa', \kappa''$ be randoms in $\mathbb{Z}_p$. For $i \in [n]$ define the following elements*

$$\begin{aligned} \boldsymbol{h}^*_{0,i} &= g_2^{\tau \boldsymbol{d}^*_{n+i} + \delta_0 \boldsymbol{d}^*_{3n+i}}, & \boldsymbol{h}^*_{1,i} &= g_2^{\tau \boldsymbol{d}^*_{2n+i} + \delta_0 \boldsymbol{d}^*_{3n+i}}, \\ \boldsymbol{e}_i &= g_1^{\omega' \boldsymbol{d}_{n+i} + \omega'' \boldsymbol{d}_{2n+i}}, & \boldsymbol{f}_i &= g_1^{\kappa' \boldsymbol{d}_{n+i} + \kappa'' \boldsymbol{d}_{2n+i}}. \end{aligned}$$

Problem 3 consists, given $\Delta = (\Gamma, \hat{\mathbb{D}}, \hat{\mathbb{D}}^*, \{\boldsymbol{e}_i, \boldsymbol{f}_i\}_{i\in[n]})$ and $\{\boldsymbol{t}_i\}_{i\in[n]}$, in deciding if $(\boldsymbol{t}_1, \cdots, \boldsymbol{t}_n) = (\boldsymbol{h}^*_{0,1}, \cdots, \boldsymbol{h}^*_{0,n})$ or $(\boldsymbol{t}_1, \cdots, \boldsymbol{t}_n) = (\boldsymbol{h}^*_{1,1}, \cdots, \boldsymbol{h}^*_{1,n})$. An adversary $\mathcal{A}$ solves Problem 3 with advantage ϵ if

$$\mathsf{Adv}^{P3}_{\mathcal{A}} := \left| \Pr\left[\mathcal{A}(\Delta, \{\boldsymbol{h}^*_{0,i}\}_{i\in[n]}) = 1\right] - \Pr\left[\mathcal{A}(\Delta, \{\boldsymbol{h}^*_{1,i}\}_{i\in[n]}) = 1\right] \right| \geq \epsilon$$

where the probability is taken over the random choice of generators $(g_1, g_2) \in \mathbb{G}_1 \times \mathbb{G}_2$, random choice of $(\mathbb{D}, \mathbb{D}^*)$, $\tau, \delta_0, \omega, \omega', \omega'', \kappa', \kappa''$ in $\mathbb{Z}_p$, and the random bits consumed by $\mathcal{A}$.

Lemma 2.3.5 *For any adversary $\mathcal{A}$, there is a probabilistic adversary $\mathcal{B}$, whose running time is essentially the same as that of $\mathcal{A}$, such that for any security parameter λ,* $\mathsf{Adv}^{P3}_{\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{XDLin2}}_{\mathcal{B}}(\lambda) + 7/p$.

The proofs of Lemmas 2.3.3, 2.3.4 and 2.3.5 are done in Appendix C of our paper [21], which is the full version of our work [20]. It follows the reductions made for [114]'s problems to DLin (which themselves relies on [115]'s proofs as [114]'s problems 1 and 2 are essentially the same as [115]'s basic problems 1 and 2.)

We also introduce two problems, **Problem 1 bis** and **Problem 2 bis**, inspired by [115]'s Problem 1 and Problem 2 respectively. We moved [115] problems from symmetric to asymmetric bilinear pairings setting, and prove that they hold if XDLin_1 and XDLin_2 hold.

Definition 2.3.8 ***Problem 1 bis.*** *Let $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ be an asymmetric bilinear pairing group of prime order, $(\mathbb{D}, \mathbb{D}^*)$ be two random dual orthonormal bases of*

$\mathbb{Z}_p^{4n+2}$, $\hat{\mathbb{D}}^* = (\boldsymbol{d}_0^*, \cdots, \boldsymbol{d}_n^*, \boldsymbol{d}_{2n+1}^*, \cdots, \boldsymbol{d}_{4n+1}^*)$, ω, γ be random elements of $\mathbb{Z}_p$ and $\{\boldsymbol{z}_i\}_{i=1}^n$ be random vector of $\mathbb{Z}_p^n$. For $i \in [\![2, n]\!]$ define the following elements

$$\boldsymbol{e}_{0,1} = g_1^{\omega \boldsymbol{d}_1 + \gamma \boldsymbol{d}_{4n+1}}, \qquad \boldsymbol{e}_{1,1} = g_1^{\omega \boldsymbol{d}_1 + \sum_{l=1}^n \sum_{j=1}^n z_{l,j} \boldsymbol{d}_{n+l} + \gamma \boldsymbol{d}_{4n+1}}, \qquad \boldsymbol{e}_i = g_1^{\omega \boldsymbol{d}_i}.$$

Problem 1 bis consists, given $\Delta = (\Gamma, \mathbb{D}, \hat{\mathbb{D}}^*, \{\boldsymbol{e}_i\}_{i \in [\![2,n]\!]})$ and $\boldsymbol{t}$, in deciding if $\boldsymbol{t} = \boldsymbol{e}_{0,1}$ or $\boldsymbol{t} = \boldsymbol{e}_{1,1}$. An adversary $\mathcal{A}$ solves Problem 1 bis with advantage ϵ if

$$\mathsf{Adv}_{\mathcal{A}}^{P1b} := |\Pr[\mathcal{A}(\Delta, \boldsymbol{t}) = 1] - \Pr[\mathcal{A}(\Delta, \boldsymbol{t}) = 1]| \geq \epsilon$$

where the probability is taken over the random choice of generator $g_1 \in \mathbb{G}_1$, random choice of $(\mathbb{D}, \mathbb{D}^*)$, ω, γ in $\mathbb{Z}_p$, $\{\boldsymbol{z}_i\}_{i=1}^n$ in $\mathbb{Z}_p^n$ and the random bits consumed by $\mathcal{A}$.

Lemma 2.3.6 *For any adversary $\mathcal{A}$, there is a probabilistic adversary $\mathcal{B}$, whose running time is essentially the same as that of $\mathcal{A}$, such that for any security parameter λ, $\mathsf{Adv}_{\mathcal{A}}^{P1b}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}}^{XDLin1}(\lambda) + 5/p$.*

Definition 2.3.9 ***Problem 2 bis.*** *Let $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ be an asymmetric bilinear pairing group of prime order, $(\mathbb{D}, \mathbb{D}^*)$ be random dual orthonormal bases of $\mathbb{Z}_p^{4n+2}$, $\hat{\mathbb{D}} = (\boldsymbol{d}_0, \cdots, \boldsymbol{d}_n, \boldsymbol{d}_{2n+1}, \cdots, \boldsymbol{d}_{4n+1})$, $\delta, \tau, \delta_0, \omega, \sigma$ be random elements of $\mathbb{Z}_p$, $\{\delta_i\}_{i=1}^n$ be random vectors of $\mathbb{Z}_p^n$, $\boldsymbol{Z}$ be a $n \times n$ invertible matrix over $\mathbb{Z}_p$, and $\boldsymbol{U} = (\boldsymbol{Z}^{-1})^\top$. For $i \in [n]$ define the following elements*

$$\boldsymbol{h}_{0,i}^* = g_2^{\delta \boldsymbol{d}_i^* + \sum_{j=1}^n \delta_{i,j} \boldsymbol{d}_{3n+i}^*}, \qquad \boldsymbol{h}_{1,i}^* = g_2^{\delta \boldsymbol{d}_i^* + \sum_{j=1}^n u_{i,j} \boldsymbol{d}_{n+i}^* + \sum_{j=1}^n \delta_{i,j} \boldsymbol{d}_{3n+i}^*}, \qquad \boldsymbol{e}_i = g_1^{\omega \boldsymbol{d}_i + \tau \sum_{j=1}^n z_{i,j} \boldsymbol{d}_{n+i}}.$$

Problem 2 bis consists, given $\Delta = (\Gamma, \hat{\mathbb{D}}, \mathbb{D}^*, \{\boldsymbol{e}_i\}_{i \in [n]})$ and $\{\boldsymbol{t}_i\}_{i \in [n]}$, in deciding if $(\boldsymbol{t}_1, \cdots, \boldsymbol{t}_n) = (\boldsymbol{h}_{0,1}^*, \cdots, \boldsymbol{h}_{0,n}^*)$ or $(\boldsymbol{t}_1, \cdots, \boldsymbol{t}_n) = (\boldsymbol{h}_{1,1}^*, \cdots, \boldsymbol{h}_{1,n}^*)$. An adversary $\mathcal{A}$ solves Problem 2 bis with advantage ϵ if

$$\mathsf{Adv}_{\mathcal{A}}^{P2b} := \left| \Pr\left[\mathcal{A}(\Delta, \{\boldsymbol{h}_{0,i}^*\}_{i \in [n]}) = 1\right] - \Pr\left[\mathcal{A}(\Delta, \{\boldsymbol{h}_{1,i}^*\}_{i \in [n]}) = 1\right] \right| \geq \epsilon$$

where the probability is taken over the random choice of generators $(g_1, g_2) \in \mathbb{G}_1 \times \mathbb{G}_2$, random choice of $(\mathbb{D}, \mathbb{D}^*)$, $\delta, \tau, \delta_0, \omega, \sigma$ in $\mathbb{Z}_p$, $\{\delta_i\}_{i=1}^n$ of $\mathbb{Z}_p^n$, $\boldsymbol{Z}$ in $GL(n, \mathbb{Z}_p)$ and the random bits consumed by $\mathcal{A}$.

Lemma 2.3.7 *For any adversary $\mathcal{A}$, there is a probabilistic adversary $\mathcal{B}$, whose running time is essentially the same as that of $\mathcal{A}$, such that for any security parameter λ, $\mathsf{Adv}_{\mathcal{A}}^{P2b}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}}^{XDLin2}(\lambda) + 5/p$.*

Proofs of Lemmas 2.3.6 and 2.3.7 are done in Appendix C of our paper [21]. They are based on [115]'s proofs, and the idea is to prove that **Problem 1 bis** and **Problem 2 bis** hold if **Problem 1** and **Problem 2** hold respectively.

We now introduce a new security problem for dual pairing vector spaces: the fixed argument dual pairing vector spaces inversion problem. This problem is the first *computational* problem for dual pairing vector spaces, and can be reduced to CDH in $\mathbb{G}_2$ (Definition 2.2.5).

Definition 2.3.10 ***Fixed argument dual pairing vector spaces inversion problem (FA-DPVS-I).*** *Let $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ be an asymmetric bilinear pairing group, $(\mathbb{D}, \mathbb{D}^*) \leftarrow \mathsf{Dual}(\mathbb{Z}_p^2)$ be two dual orthonormal bases where $\boldsymbol{d}_i \cdot \boldsymbol{d}_i^* = \psi$ for $i \in \{1, 2\}$, $\psi \in \mathbb{Z}_p^*$ and where $\mathbb{1}$ denotes the identity element of $\mathbb{G}_T$. The FA-DPVS-I problem consists, given $(\Gamma, g_1^{\boldsymbol{d}_2}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_2^*})$, in computing $g_1^{\boldsymbol{d}_1}$. An adversary $\mathcal{A}$ solves FA-DPVS-I problem with advantage ϵ if*

$$\mathsf{Adv}_{\mathcal{A}}^{FA-DPVS-I} := \left| \Pr\left[\mathcal{A}(\Gamma, g_1^{\boldsymbol{d}_2}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_2^*}) = g_1^{\boldsymbol{d}_1} \right] \right| \geq \epsilon$$

where the probability is taken over the random choice of generators $(g_1, g_2) \in \mathbb{G}_1 \times \mathbb{G}_2$, the random choice of $(\mathbb{D}, \mathbb{D}^*)$ and the random bits consumed by $\mathcal{A}$.

Lemma 2.3.8 *The fixed argument dual pairing vector spaces inversion assumption has a unique solution.*

Proof 2.3.1 *Breaking the assumption means to find an element $\boldsymbol{t} = (g_1^{t_1}, g_1^{t_2}) \in \mathbb{G}_1^2$ for $t_1, t_2 \in \mathbb{Z}_p$ such that*

$$\begin{cases} e(\boldsymbol{t}, g_2^{\boldsymbol{d}_1^*}) = e(g_1, g_2)^{\psi} \\ e(\boldsymbol{t}, g_2^{\boldsymbol{d}_2^*}) = \mathbb{1} \end{cases} \tag{2.1}$$

where $\psi \in \mathbb{Z}_p^*$ and $\mathbb{1}$ is the identity element of $\mathbb{G}_T$. The above system can be rewritten as

$$\begin{cases} e(g_1^{t_1}, g_2^{d_{1,1}^*}) \cdot e(g_1^{t_2}, g_2^{d_{1,2}^*}) = e(g_1, g_2)^{\psi} \\ e(g_1^{t_1}, g_2^{d_{2,1}^*}) \cdot e(g_1^{t_2}, g_2^{d_{2,2}^*}) = \mathbb{1} \end{cases} \tag{2.2}$$

In the exponent, the system becomes

$$\begin{cases} t_1 d_{1,1}^* + t_2 d_{1,2}^* = \psi \\ t_1 d_{2,1}^* + t_2 d_{2,2}^* = 0 \end{cases} \tag{2.3}$$

and has a unique solution if $d^*_{1,1}d^*_{2,2} - d^*_{2,1}d^*_{1,2} \neq 0$*. By case-based reasoning we have that* $d^*_{1,1}d^*_{2,2} - d^*_{2,1}d^*_{1,2} = 0$ *if*

- $(d^*_{1,1}, d^*_{1,2})$ *or* $(d^*_{2,1}, d^*_{2,2})$ *is equal to* $(0,0)$*. This is not possible by dual pairing vector spaces definition.*
- $(d^*_{1,1}, d^*_{1,2})$ *and* $(d^*_{2,1}, d^*_{2,2})$ *are respectively equals to either* $(d^*_{1,1}, 0)$ *and* $(d^*_{2,1}, 0)$ *or* $(0, d^*_{1,2})$ *and* $(0, d^*_{2,2})$*. In these cases the system (2.3) becomes* $\begin{cases} t_1 d^*_{1,1} = \psi \\ t_1 d^*_{2,1} = 0 \end{cases}$ *or* $\begin{cases} t_2 d^*_{1,2} = \psi \\ t_2 d^*_{2,2} = 0 \end{cases}$ *, which does not have a solution. By definition of dual pairing vector spaces, this is not possible.*
- $(d^*_{1,1}, d^*_{1,2}) = (d^*_{2,1}, d^*_{2,2})$ *which is not possible by definition of dual pairing vector spaces.*
- $(d^*_{2,1}, d^*_{2,2}) = (d^*_{1,2}, d^*_{1,1})$*. In this case the equation* $d^*_{1,1}d^*_{2,2} - d^*_{2,1}d^*_{1,2} = 0$ *that can be rewritten as* $d^*_{1,1}d^*_{2,2} = d^*_{2,1}d^*_{1,2}$ *becomes* $d^{*2}_{1,1} = d^{*2}_{1,2}$*. We now have two cases:*
  - *either* $d^*_{1,1} = d^*_{1,2}$ *thus in this case* $(d^*_{1,1}, d^*_{1,2}) = (d^*_{2,1}, d^*_{2,2})$ *which is not possible by definition of dual pairing vector spaces;*
  - *or* $d^*_{1,1} = -d^*_{1,2}$ *thus in this case the system 2.3 becomes* $\begin{cases} -d^*_{1,2}t_1 + d^*_{1,2}t_2 = 0 \\ d^*_{1,2}t_1 - d^*_{1,2}t_2 = 0 \end{cases}$ *and thus* $\begin{cases} 0 = \psi \\ d^*_{1,2}t_1 = d^*_{1,2}t_2 \end{cases}$ *which is not possible as* $\psi \neq 0$*.*

Thus, by construction $d^*_{1,1}d^*_{2,2} - d^*_{2,1}d^*_{1,2} \neq 0$ *and the above system as a unique solution:* $\boldsymbol{t} = g_1^{\boldsymbol{d}_1}$*.* □

Theorem 2.3.1 *If the computational Diffie-Hellman assumption in* $\mathbb{G}_2$ *holds, then the fixed argument dual pairing vector spaces inversion assumption holds.*

To do the proof of the above theorem, we need an intermediate problem.

Definition 2.3.11 ***Intermediate problem*** *(IP). Let* $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ *be an asymmetric bilinear pairing group and* a, b *be randoms in* $\mathbb{Z}_p$*. The intermediate problem consists,* $(\Gamma, g_1^a, g_2^a, g_2^b)$*, in computing* g_1^{ab}*. An adversary* $\mathcal{A}$ *solves the intermediate problem with advantage* ϵ *if*

$$\mathsf{Adv}^{IP}_{\mathcal{A}} := \left|\Pr\left[\mathcal{A}(\Gamma, g_1^a, g_2^a, g_2^b) = g_1^{ab}\right]\right| \geq \epsilon$$

where the probability is taken over the random choice of generators $(g_1, g_2) \in \mathbb{G}_1 \times \mathbb{G}_2$*, the random choice of* $a, b \in \mathbb{Z}_p$ *and the random bits consumed by* $\mathcal{A}$*.*

Theorem 2.3.2 *If CDH in* $\mathbb{G}_2$ *holds, then the intermediate problem holds.*

Proof 2.3.2 *We prove the contrapositive. Let* $\mathcal{B}$ *be an adversary that breaks the intermediate problem with non-negligible advantage. We build* $\mathcal{A}$ *that uses* $\mathcal{B}$ *to break CDH in* $\mathbb{G}_2$*.*
$\mathcal{A}$ *is given* (Γ, g_2^a, g_2^b)*. Using type 2 pairings, there exists* ϕ *from* $\mathbb{G}_2$ *to* $\mathbb{G}_1$*. Then* $\mathcal{A}$ *gives to* $\mathcal{B}$*:* $(\Gamma, \phi(g_2^a) = g_1^a, g_2^a, g_2^b)$*. The latter answers with* g_1^{ab} *and* $\mathcal{A}$ *returns* $\phi(g_1^{ab}) = g_2^{ab}$ *as her answers. Notice that* $\mathcal{A}$*'s advantage is equal to* $\mathcal{B}$*'s advantage, therefore it is non-negligible.* □

Note 2.3.3 *Notice that the above proof is possible only when considering type 2 pairings (Section 2.2.2).*

Note 2.3.4 *The intermediate problem has a unique solution. Indeed we can rewrite the problem as a fixed pairing inversion problem [68] as its aim is to find, given* $g_2 \in \mathbb{G}_2$ *and* $e(g_1, g_2)^{ab} \in \mathbb{G}_T$ *(computed from the other inputs of the problem),* $t \in \mathbb{G}_1$ *such that* $e(t, g_2) = e(g_1, g_2)^{ab}$*. Then as stated in [68], as* e *is non-degenerate and the groups* $\mathbb{G}_1, \mathbb{G}_2$ *and* $\mathbb{G}_T$ *are cyclic of prime order* p*, then solution to the above problem is unique.*

Now we reduce the fixed argument dual pairing vector spaces inversion problem to the intermediate problem.

Theorem 2.3.3 *If the intermediate problem holds, then fixed argument dual pairing vector spaces inversion problem holds.*

Proof 2.3.3 *We prove the contrapositive. Let* $\mathcal{B}$ *be an adversary that breaks the FA-DPVS-I problem with non-negligible advantage. We build* $\mathcal{A}$ *that uses* $\mathcal{B}$ *to break the intermediate problem.*
$\mathcal{A}$ *is given* $(\Gamma, g_1^a, g_2^a, g_2^b)$*.* $\mathcal{A}$ *runs* $\mathsf{Dual}(\mathbb{Z}_p^2)$ *to get* $(\mathbb{B}, \mathbb{B}^*)$ *dual orthonormal bases of dimension* 2*. Then* $\mathcal{A}$ *defines new orthonormal bases* $(\mathbb{D}, \mathbb{D}^*)$ *as follows:* $\boldsymbol{d}_1 = ab\boldsymbol{b}_1$*,* $\boldsymbol{d}_2 = a\boldsymbol{b}_2$*,* $\boldsymbol{d}_1^* = \boldsymbol{b}_1^*$ *and* $\boldsymbol{d}_2^* = b\boldsymbol{b}_2^*$*. We easily notice that* $(\mathbb{D}, \mathbb{D}^*)$ *are dual orthonormal.*
Then $\mathcal{A}$ *gives* $(\Gamma, g_1^{\boldsymbol{d}_2}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_2^*})$ *to* $\mathcal{B}$*. Notice that* $g_1^{\boldsymbol{d}_2} = (g_1^a)^{\boldsymbol{b}_2}$*,* $g_2^{\boldsymbol{d}_1^*} = g_2^{\boldsymbol{b}_1^*}$ *and* $g_2^{\boldsymbol{d}_2^*} = (g_2^b)^{\boldsymbol{d}_2^*}$*.*
$\mathcal{B}$ *answers with* $\boldsymbol{t} = (t_1, t_2) \in \mathbb{G}_1^2$*, which is equal to* $g_1^{\boldsymbol{d}_1} \in \mathbb{G}_1^2$ *by uniqueness of the solution in FAP-DPVS-I assumption, and can be rewrite as* $(g_1^{\boldsymbol{d}_{1,1}}, g_1^{\boldsymbol{d}_{1,2}})$*. By construction, it is equal to* $(g_1^{ab\boldsymbol{b}_{1,1}}, g_1^{ab\boldsymbol{b}_{1,2}})$ *and* $\mathcal{A}$ *can returns* $t_1^{(\boldsymbol{b}_{1,1})^{-1}}$ *as her answers as it is equal to* g_1^{ab}*.*□

The proof of the theorem 2.3.1 is done by combining Theorem 2.3.2 and Theorem 2.3.3.

3

Cryptographic Preliminaries

Contents

THIS chapter introduces formal methods for establishing the security of cryptographic schemes. Additionally, it presents three cryptographic primitives that we will use in this thesis.

3.1 Proving Security

When it comes to security, cryptography follows the principle established in the 19th century by *Kerckhoffs*: to ensure the security of an encryption scheme, the method must be public (it will inevitably become public one day). Only a small part of the method, known as the key, should remain secret and be easily changeable.

Traditionally, proving the security of a cryptographic scheme consisted of searching for attacks on this scheme. If no attack that contradicts the scheme's security property was found, the scheme was considered secure. However, we can never be certain that an attack does not exist, and another method is needed to prove the security of cryptographic primitives. In this section, we present such a method, called *provable security*, along with two ways to realize proofs in this paradigm. We also introduce different types of attacks an adversary can launch and security models.

3.1.1 Provable Security

Provable security is an approach for demonstrating the security of a cryptographic scheme, introduced in the 1980s. Its principle is to relate the security of a cryptographic scheme to that of its underlying primitives or (well-established) computational or decisional problems. To prove security in this paradigm, one must first identify the security goals and the adversary's capabilities of the cryptographic scheme. Then, do the same for the underlying primitives and computational problems. Finally, one provides a *reduction* showing how to transform an adversary that breaks the security goals of the scheme into an adversary that breaks the security goals of the underlying primitives and problems [1]. In short, provable security guarantees that a scheme is secure relative to a specific security definition against a given adversarial model and under a particular assumption. The adversary's capabilities are defined with respect to the kinds of attacks and according to a model that defines the attack environment.

Therefore, the security of cryptographic schemes relies on the difficulty of the underlying mathematical problems. If these problems are successfully solved, the security of the schemes is compromised. That is why cryptographic schemes are built upon mathematical problems that are presumed to be "hard" to solve. An example of such

a “hard” problem is the factorization of an integer N into two large primes p and q. Here “hard” actually means that breaking the problem requires performing so many operations that, without unlimited computational power, it is not possible to complete all the computations in a reasonable time. The notion of “reasonable time” has evolved over the years. For instance, when Alan Turing attempted to break Enigma, he had less than a day to find the machine settings, as new settings were used each day.

3.1.2 Proofs Methods

In provable security, the security proofs can be done in two ways: *game-based* and *simulation-based*.

Game-based security. In the game-based approach, security property of the scheme is linked to a particular event and security is demonstrated in the form of a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ (usually modeled as a PPT algorithm). The adversary's goal is to solve the game with non-negligible probability, using a set of query oracles modeling her capabilities according to the security model and the type of attacks. Her advantage in winning the game is determined based on the type of the associated problem. If the problem is *decisional*, then $\mathcal{A}$'s advantage is defined as the difference between her probability of winning the game and the probability of winning the game through random guessing. On the other hand, if the problem is *computational*, $\mathcal{A}$'s advantage is defined as her probability of winning the game with a random proposal. If the adversary's advantage is negligible, we say that the primitive satisfies the security property. However, in some cases, reductions are too complicated to be be accomplished in a single step. That is why Shoup [130] formalized the *game-hopping* technique in 2004. In this methodology, a proof starts with an initial game that comes from the security property to prove. From the initial game, a sequence of subsequent games is constructed, with the final one being simple enough for direct analysis. Importantly, it must be proven that the adversary $\mathcal{A}$ cannot detect any difference between two consecutive games in the sequence. Ultimately, the probability of an adversary winning the first security game is reduced to her advantage in winning the last security game. In this manuscript, we will utilize this method for our security proofs.

Simulation-based security. In the simulation-based approach, the security of a scheme is established by demonstrating that an adversary's capabilities in the real execution of the primitive are no greater than what they can achieve in an ideal scenario, which is inherently secure by definition. This concept is often referred to as the *real world/ideal world* paradigm. To demonstrate the indistinguishability of these two worlds,

the objective is to present an ideal world adversary, commonly known as a simulator, for each real world adversary. The key criterion is that the output generated by the simulator should be indistinguishable from the output produced by the real world adversary.

We now present possible attacks by the adversary and the environments in which these attacks can occur.

3.1.3 Attacks by the Adversary

In the sequel, we employ the *game-based security* approach and address a *decisional* problem. The adversary is given a ciphertext from which she must solve the problem; this ciphertext is referred to as the *challenge* ciphertext. The adversary is also granted the ability to submit queries to the challenger. For any cryptographic encryption scheme, we consider three types of attacks in which the adversary has a varying levels of attacking capability.

- *Chosen-Plaintext Attack* (CPA): in this scenario, the adversary can acquire ciphertexts of their choice. It is worth noting that for public-key schemes, providing the adversary with the public key is enough to facilitate these attacks.
- *Non-Adaptive Chosen-Ciphertext Attack* (CCA1) [111]: in this scenario, the adversary possesses the scheme's public key and has access to an oracle for the decryption function. However, the adversary can only make queries to this oracle before receiving the challenge ciphertext.
- *Adaptive Chosen-Ciphertext Attack* (CCA2) [122]: in this scenario, the adversary, in addition to the public key, also has access to an oracle for the decryption function. Importantly, the adversary can make queries to this oracle even after receiving the challenge ciphertext. However, there is a restriction: the adversary cannot request the decryption of the challenge ciphertext itself.

CPA attacks are less powerful than CCA1 attacks, and CCA1 attacks, in turn, are less powerful than CCA2 attacks. In this thesis, we prioritize the efficiency of schemes over their security levels, so we will primarily focus on CPA security.

3.1.4 Attack Environments

Attack environments can be defined according to several models. We here present some of them.

Computational model [74]. In the computational model, an adversary is an arbitrary probabilistic algorithm with limitations, and cryptographic primitives are represented as (tuples of) algorithms that adhere to security and constructive assumptions (for example, encryption and decryption are inversely related).

Standard model. In a computational model, if the adversary's limitations are solely related to time and computing power, we refer to this as the *standard model*. In this model, the security of cryptographic schemes is established solely based on complexity assumptions. It is important to note that proofs in the standard model are recognized to be complex and challenging to attain, and schemes proven to be secure in this model often encounter efficiency

There are additional security models in which cryptographic primitives are substituted with idealized versions, with the aim of simplifying the proof.

Random oracle model (ROM) [25]. Within this computational model, a theoretical black box known as the random oracle is presumed to exist and be accessible to the adversary. Random oracles are commonly employed as ideal substitutes for cryptographic hash functions when robust randomness within their output domain is necessary. When queried at a new domain point, the random oracle produces a randomly selected value from its range. For previously queried points, it retrieves the same value that it provided initially, as it retains all of its responses in memory.

Generic group model (GGM) [105]. Within this computational model, the adversary gains access to an oracle responsible for executing group operations. In this context, efficient group encodings are substituted with randomly chosen encodings, thereby preventing the adversary from exploiting any specific group structure.

Other models. Other, less frequently employed models exist, such as the *weak multilinear maps model* [108] which encompass all known categories of attacks on multilinear map[1] or the *ideal cipher* model [128] where block ciphers are substituted with a random permutation for every key.

[1] A multilinear map is defined a bilinear map except that the map e now takes as input n elements instead of 2.

3.1.5 Computing Power

Computers are getting more and more powerful following the observation of Moore from 1965 [109], which states that computer processing speeds roughly double every two years. Therefore, while in the 1980s we considered that 2^{60} operations were computationally infeasible, a few years ago in 2008, the bound was 2^{80}, and we are now going to consider that 2^{100} operations are not possible. To maintain the security of existing cryptographic schemes, the sizes of their parameters (*i.e.* the keys) are increased to overcome the growing power of computers.

For several years, the possibility of deploying super powerful computers, called *quantum* computers, has made the community wonder about the robustness of existing cryptographic schemes. This is due to the fact that in 1994 Shor [129] proposed an algorithm for quantum computers that reduces algorithms complexity, especially the complexity of "hard" problems such as factorization or discrete logarithm. On the other hand, problems relying on error-correcting codes or a mathematical object called *lattices* are supposed to be resistant against a quantum computer.

3.2 Public Key Encryption Schemes

A public key encryption scheme is an encryption system parameterized by a secret key and a public key. Encryption is performed using the scheme's public key, and decryption is only possible with knowledge of the scheme's secret key.

Definition 3.2.1 ***Public key encryption scheme*** *(PKE). A public key encryption scheme $\mathcal{E}$ consists of three algorithms:*

- KeyGen*: the key generation algorithm takes as input a security parameter λ, and returns a pair of secret key sk and public key pk.*
- Encrypt*: the encryption algorithm takes as input a public key pk along with a message m and returns ct the encryption of m under pk.*
- Decrypt*: the decryption algorithm takes as input a secret key sk along with a ciphertext ct and returns a message m'.*

Definition 3.2.2 ***Correctness.*** *A public key encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is said to be* correct *if for all security parameter 1^λ, all honestly generated key pair $(sk, pk) \leftarrow \mathsf{KeyGen}(1^\lambda)$, and all message m, $\Pr\left[\mathsf{Decrypt}(\mathsf{sk}, \mathsf{Encrypt}(\mathsf{pk}, \mathsf{m})) = \mathsf{m}\right] = 1$.*

An encryption scheme will be consider *secure* if it preserves the privacy of messages, meaning that from a ciphertext no adversary can learn information about its plaintext, except its length. This security notion is referred to as *indistinguishability* [74]. We now formally present it.

Definition 3.2.3 ***Adaptive indistinguishability security***. *A public key encryption scheme* $\mathcal{E} = (\text{KeyGen}, \text{Encrypt}, \text{Decrypt})$ *is said to satisfy* adaptive indistinguishability security *if the advantage of any adversary* $\mathcal{A}$ *of winning the security game presented in Figure 3.1 is negligible. Let* $\mathcal{C}$ *be a challenger.*

- `SETUP`: $\mathcal{C}$ takes as input a security parameter λ, runs $\text{KeyGen}(\lambda)$ to get (sk, pk) and gives pk to $\mathcal{A}$.
- `CHALLENGE`: $\mathcal{A}$ chooses two challenge messages m_0, m_1 of same length and sends them to $\mathcal{C}$. The latter picks $b \leftarrow \{0,1\}$ and returns to $\mathcal{A}$ the challenge ciphertext $\text{ct}_b \leftarrow \text{Encrypt}(\text{pk}, \text{m}_b)$.
- `GUESS`: $\mathcal{A}$ outputs a guess bit $b' \in \{0,1\}$. $\mathcal{A}$ wins the security game if $b' = b$.

Figure 3.1: Adaptive indistinguishability security game for public key encryption schemes.

In the above definition, adversary $\mathcal{A}$ gives to challenger $\mathcal{C}$ the two challenge messages m_0, m_1 after seeing the scheme's public key. There exists a weaker notion of indistinguishability security, called *selective indistinguishability* in which the adversary must choose challenge messages before seeing the scheme's public key. The formal definition can easily be derived from the above definition. Selective security, while perhaps justified in some cases, is too restrictive for realistic applications. That is why in this thesis we will focus on adaptive security.

Note 3.2.1 *There exists another required security for encryption scheme,* non-malleability *[60] which requires the inability of an adversary, given a challenge ciphertext, to output a different ciphertext such that the plaintexts underlying both ciphertexts are "meaningfully related".*

In this thesis, as we focus on privacy concerns we will mainly focus on indistinguishability security.

These security goals IND (indistinguishability) and NM (non-malleability) can be "mixed" with the types of attacks seen in Section 3.1. It gives us the following security notions: IND-CPA, IND-CCA1, IND-CCA2, NM-CPA, ...

Combining this with what we said in the Section 3.1, we get that in this manuscript we will focus on *adaptive IND-CPA* security.

Based on the definition of public encryption schemes, the literature has introduced numerous encryption schemes, often referred to as “advanced”. These innovative schemes not only enhance the confidentiality of public key encryption but also introduce new functionalities. In Chapters 4 and 6, we will introduce several such advanced encryption schemes. We now present another public encryption scheme, called *inner product encryption scheme*. Although we do not make any contributions to this primitive, it will serve as a foundational component for our research.

3.3 Inner Product Encryption Schemes

Informally, in a inner product encryption scheme, a secret key is associated to a vector, let us say $\boldsymbol{u}$ while a ciphertext is associated to a vector denoted $\boldsymbol{v}$ and the former can decrypt the latter if and only if $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = 0$.

Definition 3.3.1 ***Inner product encryption scheme*** *(IPE) [84]. An inner product encryption scheme consists of four algorithms:*

- Setup*: the setup algorithm takes as input a security parameter λ and a vector length $n \in \mathbb{N}$ and outputs a master secret key msk along with a public key pk.*
- KeyGen*: the key generation algorithm takes as input a master secret key msk and a vector $\boldsymbol{u}$ and returns a secret key $\mathsf{sk}_{\boldsymbol{u}}$ created for $\boldsymbol{u}$.*
- Encrypt*: the encryption algorithm takes as input a public key pk, a vector $\boldsymbol{v}$ and a message m to encrypt. It returns a ciphertext $\mathsf{ct}_{\boldsymbol{v}}$ of message m, according to $\boldsymbol{v}$.*
- Decrypt*: the decryption algorithm takes as input a secret key $\mathsf{sk}_{\boldsymbol{u}}$ and a ciphertext $\mathsf{ct}_{\boldsymbol{v}}$ and outputs a message m'.*

Definition 3.3.2 ***Correctness.*** *An inner product encryption scheme is said to be* correct *if for all security parameter λ, all integer n, every honestly generated key pairs $(msk, pk) \leftarrow \mathsf{Setup}(\lambda, n)$, every messages m and every vectors $\boldsymbol{u}, \boldsymbol{v}$ such that $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = 0$,*

$$\Pr\left[\mathsf{Decrypt}(\mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{u}), \mathsf{Encrypt}(\mathsf{pk}, \boldsymbol{v}, \mathsf{m})) = \mathsf{m}\right] = 1.$$

Regarding security, IPE schemes can satisfy two indistinguishability security properties: the first one, called *payload-hiding* security, prevents any adversary to learn information about the plaintext from the ciphertext; and the second, called *attribute-hiding* security,

prevents any adversary to learn information about the encryption vector, from the ciphertext. The latter property is said to be *weak* if the adversary is not allowed to query secret keys that can decrypt the challenge ciphertext, otherwise it is said to be *fully* or *strong*.

Definition 3.3.3 ***Adaptive payload-hiding security*** *(PH-IPE) [84]. An inner product encryption scheme is said to satisfy adaptive* payload-hiding *security (or to be adaptively payload-hiding) if all PPT adversaries $\mathcal{A}$ have at most negligible advantage in winning the game presented in Figure 3.2, where $\mathcal{A}$'s advantage is defined as* $\mathsf{Adv}_{\mathcal{A}}^{PH-IPE}(\lambda) := \Pr\left[b' = b\right] - 1/2$ *for any $\lambda \in \mathbb{N}$. Let $\mathcal{C}$ be a challenger.*

- SETUP: $\mathcal{C}$ on input (λ, n) runs $\mathsf{Setup}(\lambda, n)$ to get $(\mathsf{msk}, \mathsf{pk})$ and gives pk to $\mathcal{A}$.
- KEY QUERY: $\mathcal{A}$ may adaptively query a key for vector $\boldsymbol{u}$. In response, $\mathcal{A}$ is given by $\mathcal{C}$ the corresponding secret key $\mathsf{sk}_{\boldsymbol{u}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{u})$.
- CHALLENGE: $\mathcal{A}$ sends to $\mathcal{C}$ challenge pattern $\boldsymbol{v}$ along with two challenge messages $\mathsf{m}_0, \mathsf{m}_1$. The latter randomly picks $b \leftarrow \{0,1\}$ and returns to $\mathcal{A}$ $\mathsf{ct}^b \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \boldsymbol{v}, \mathsf{m}_b)$.
- KEY QUERY: $\mathcal{A}$ may continue to issue key query for vector $\boldsymbol{u}$ and is given $\mathsf{sk}_{\boldsymbol{u}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{u})$.
- GUESS: $\mathcal{A}$ outputs a bit b' and wins if $b' = b$ and if, for all $\boldsymbol{u}$ for which a key was queried, the condition $\langle \boldsymbol{u}, \boldsymbol{v} \rangle \neq 0$ holds.

Figure 3.2: Adaptive payload-hiding security game for inner product encryption schemes.

We now present the definition of adaptive strong attribute-hiding security.

Definition 3.3.4 ***Adaptive strong attribute-hiding security*** *(sAH-IPE) [84]. An inner product encryption scheme is said to satisfy adaptive strong* attribute-hiding *security (or to be adaptively strong attribute-hiding) if all PPT adversaries $\mathcal{A}$ have at most negligible advantage in winning the game presented in Figure 3.3, where $\mathcal{A}$'s advantage is defined as* $\mathsf{Adv}_{\mathcal{A}}^{sAH-IPE}(\lambda) := \Pr\left[b' = b\right] - 1/2$ *for any $\lambda \in \mathbb{N}$. Let $\mathcal{C}$ be a challenger.*

The definition of adaptive weak attribute-hiding security (wAH-IPE) can easily be derived from the one above, by changing the restriction on vectors for which secret key as queried. The new restriction is that any queried vector $\boldsymbol{u}$ must satisfy $\langle \boldsymbol{u}, \boldsymbol{v}_0 \rangle = \langle \boldsymbol{u}, \boldsymbol{v}_1 \rangle = 1$.

Note 3.3.1 *We can combine simultaneously weak attribute-hiding security and payload-hiding security as in the former the adversary cannot query keys that decrypt the challenge ciphertext, thus two different challenge messages can be used; but we cannot*

- SETUP: $\mathcal{C}$ on input (λ, n) runs Setup(λ, n) to get $(\mathsf{msk}, \mathsf{pk})$ and gives pk to $\mathcal{A}$.
- KEY QUERY: $\mathcal{A}$ may adaptively query a key for vector $\boldsymbol{u}$. In response, $\mathcal{A}$ is given by $\mathcal{C}$ the corresponding secret key $\mathsf{sk}_{\boldsymbol{u}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{u})$.
- CHALLENGE: $\mathcal{A}$ sends to $\mathcal{C}$ two challenge patterns $\boldsymbol{v}_0$ and $\boldsymbol{v}_1$ along with a challenge message m. The latter randomly picks $b \leftarrow \{0, 1\}$ and returns to $\mathcal{A}$ $\mathsf{ct}^b \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \boldsymbol{v}_b, \mathsf{m})$.
- KEY QUERY: $\mathcal{A}$ may continue to issue key query for vector $\boldsymbol{u}$. $\mathcal{A}$ is then given $\mathsf{sk}_{\boldsymbol{u}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{u})$.
- GUESS: $\mathcal{A}$ outputs a bit b' and wins if $b' = b$ and if, for all $\boldsymbol{u}$ for which a key was queried, the condition $\langle \boldsymbol{u}, \boldsymbol{v} \rangle \neq 0$ holds.

Figure 3.3: Adaptive strong attribute-hiding security game for inner product schemes.

do the same with strong attribute-hiding. However this does not mean that an IPE scheme cannot be both payload-hiding and strong attribute-hiding, it only means that both property must be proven separately.

Note 3.3.2 *Payload-hiding security is weaker that weak attribute-hiding security, which is itself weaker than strong attribute-hiding security.*

3.4 Signature Schemes

A digital signature scheme, also known simply as a signature scheme, is a primitive used to verify the authenticity and integrity of messages. Briefly, it is parameterized by both a signing key, which is kept secret, and a verification key, which is made public and each user possesses her own pair of keys. The signature of a message is done with the signing key and the verification is done using the verification key.

Definition 3.4.1 ***Digital signature scheme.*** *A digital signature scheme consists of three algorithms:*

- Setup(λ)*: the setup algorithm takes as input a security parameter λ and outputs a signing key sk along with a verification key vk.*
- Sign$(\mathsf{sk}, \mathsf{m})$*: the signature algorithm takes as input a signing key sk and a message m and returns a signature σ.*
- Verify$(\mathsf{vk}, \sigma, \mathsf{m})$*: the verification algorithm takes as input a verification key vk, a signature σ and a message m. It returns 1 if the signature σ is a valid signature of message m, and return 0 otherwise.*

Definition 3.4.2 ***Correctness.*** *A signature scheme is said to be* correct *if for all security parameter λ, every honestly generated key pair $(sk, vk) \leftarrow \mathsf{Setup}(\lambda)$, every message m:*

$$\Pr\left[\mathsf{Verify}(\mathsf{vk}, \mathsf{Sign}(\mathsf{sk}, \mathsf{m}), \mathsf{m}) = 1\right] = 1.$$

Definition 3.4.3 ***Adaptive unforgeability*** *(Unf). A signature scheme is said to satisfy adaptive* unforgeability *security if all PPT adversaries $\mathcal{A}$ have at most negligible advantage in the game presented in Figure 3.4, where $\mathcal{A}$'s advantage is defined as*

$$\mathsf{Adv}^{\mathsf{Unf}}_{\mathcal{A}}(\lambda) := \Pr\left[\mathsf{Verify}(\mathsf{vk}, \sigma^*, \mathsf{m}^*) = 1 | (\mathsf{m}^*, \sigma^*) \leftarrow \mathcal{A}(\mathsf{vk})\right] = 1.$$

Let $\mathcal{C}$ be a challenger.

`SETUP`: $\mathcal{C}$ on input λ runs $\mathsf{Setup}(\lambda)$ to get $(\mathsf{sk}, \mathsf{vk})$ and sends vk to $\mathcal{A}$.
`SIGNATURE QUERY`: $\mathcal{A}$ chooses a message m and sends it to $\mathcal{C}$, who responds with $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{m})$.
`GUESS`: $\mathcal{A}$ outputs a message m^* and a signature σ^* and wins the game if m^* was not queried to $\mathcal{C}$ and $\mathsf{Verify}(\mathsf{vk}, \sigma^*, \mathsf{m}^*) = 1$.

Figure 3.4: Adaptive unforgeability security game.

3.5 Dual System Encryption Framework

The dual system encryption (DSE) framework is a novel approach within the provable security paradigm. It was introduced by Waters in 2009 [137] to establish the (adaptive) security of public key encryption schemes. This framework operates within the context of game-based security and will be employed in this thesis to demonstrate the security of our encryption schemes.

Within this framework, encryption schemes are designated as *dual system encryption* schemes if both ciphertexts and private keys can assume two indistinguishable forms: *normal* or *semi-functional* (SF). Normal secret keys and ciphertexts are generated through the system's key generation or encryption algorithm and behave as expected in a typical encryption scheme. Semi-functional ciphertexts and secret keys are solely utilized in the security proof, not in the actual system. An important characteristic is that a normal key can decrypt both normal or semi-functional ciphertexts, and a normal ciphertext can be decrypted by both normal or semi-functional secret keys. However,

attempting to decrypt a semi-functional ciphertext with a semi-functional secret key will result in failure.

In this framework, a security proof is established through a sequence of games, which are demonstrated to be indistinguishable one to each other. The initial game is the scheme's original security game, involving normal secret keys and ciphertexts. The subsequent game closely resembles the first, with the exception that the challenge ciphertext is now semi-functional. We contend that no adversary can discern this alteration (except by violating a security assumption) because all provided secret keys are normal and, therefore, capable of decrypting the challenge ciphertext (assuming the decryption condition is met), regardless of whether it is normal or semi-functional. Let $q \in \mathbb{N}$ represents the number of key requests that an attacker is permitted to make. Accordingly, we define games 1 to q as follows: in Game$_k$, the first k keys are semi-functional, while the remaining keys are normal; the challenge ciphertext remains semi-functional. In Game$_q$ both the keys and the challenge ciphertext are semi-functional. As none of the provided keys can effectively decrypt the challenge ciphertext (since all keys are semi-functional, the challenge ciphertext is as well), demonstrating security at this stage becomes straightforward.

When proving indistinguishability of Game$_k$ and Game$_{k-1}$, we actually create a simulator who is prepared to generate a semi-functional challenge ciphertext and is also ready to make the k-th key either normal or semi-functional. This situation presents a potential problem. Indeed, the simulator can determine whether key k is semi-functional by testing decryption with a semi-functional ciphertext (created by her) that should be decrypted by the normal form of the k-th secret key. To address this issue, Waters proposed the use of "tag" in dual-system encryption schemes: random tag values are associated with each ciphertext and secret key, and decryption only succeeds when the tag values of the ciphertext and the decrypting key are different. If the simulator creates a semi-functional ciphertext for herself that should be decrypted by the k-th key assuming it is normal, she would only be able to create one with an identical tag, and thus decryption will fail even if the secret key is normal. Since an adversary can only query secret keys that cannot decrypt the challenge ciphertext, this correlation of tags is hidden from her, and the tags appear randomly distributed from the adversary's perspective.

This solution introduces additional complications to the security proof of the scheme and increases the size of its parameters. That is why Lewko and Waters [95] proposed a different approach to address the aforementioned paradox. Instead of causing decryption to fail when the simulator attempts to test the semi-functionality of the k-th key, they ensure that decryption succeeds even when the key is semi-functional. To achieve this, they introduce a variant of semi-functional keys, which they refer to as "nominally" semi-functional keys. These keys are distributed similarly to semi-functional

keys but are, in fact, correlated with semi-functional ciphertexts. Consequently, when a nominally semi-functional key is employed to decrypt a semi-functional ciphertext, the interaction of these two semi-functional components leads to cancellation, resulting in successful decryption.

Composite order groups and dual system encryption framework. Lewko and Waters [95] provided encryption schemes in the (symmetric) bilinear pairing groups of composite order setting, with security proofs made with the dual system encryption framework. Their idea is to take the group order N equals to $p_1p_2p_3$, where p_1, p_2, p_3 a three different primes. They define the subgroups $\mathbb{G}_{p_1}, \mathbb{G}_{p_3}$ as normal space and the subgroup $\mathbb{G}_{p_2}$ as semi-functional space. As composite order groups achieve the structure of canceling bilinear maps (Definition 2.2.3), it is easy to see that a normal key will decrypt both a normal and a semi-functional ciphertext, and that a normal ciphertext can be decrypted by both a normal and a semi-functional key. However, decryption of a semi-functional ciphertext by a semi-functional key is not possible (unless if the key is nominally) as there would be an extra component in $\mathbb{G}_{p_2}$. The indistinguishability (from the adversary point of view) between normal and semi-functional secret keys and ciphertexts is guaranteed by the general subgroup decision assumption (Definition 2.2.14).

Dual pairing vector spaces and dual system encryption framework. Lewko [90] propose encryption schemes, proven to be secure with the dual system encryption framework, in the bilinear pairing prime order groups setting. Her idea is to exploit the fact that dual pairing vector spaces satisfy the canceling bilinear maps structure. In her scheme, dual orthonormal bases $(\mathbb{B}, \mathbb{B}^*)$ are sampled for a dimension $n \in \mathbb{N}$. Let $k \in \mathbb{N}$ such that $k \leq 2n$. Then normal keys and ciphertexts are defined using vectors $\boldsymbol{b}_1, \cdots, \boldsymbol{b}_k$ and $\boldsymbol{b}_1^*, \cdots, \boldsymbol{b}_k^*$ while semi-functional keys and ciphertexts are defined using vectors $\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n$ and $\boldsymbol{b}_1^*, \cdots, \boldsymbol{b}_n^*$. With this definition, it is clear that normal keys will decrypt normal or semi-functional ciphertexts, that normal ciphertexts will be decrypted by normal or semi-functional keys but semi-functional keys will not decrypt semi-functional ciphertext (unless if the key is nominally semi-functional). From the adversary point of view, normal and semi-functional secret keys and ciphertexts are indistinguishable, thanks to the parameter hiding property of DPVS (Definition 2.3.3).

4

First Cryptographic Tool: Identity-Based Encryption with Wildcards

Contents

THE first cryptographic tool we used in this thesis is *identity-based encryption with wildcards* (WIBE), introduced in 2006 by Abdalla *et al.* [4]. Informally, in such primitive secret keys and ciphertexts are associated with vectors, over a set that contains a "wildcard" symbol denoted "$\star$". Decryption is possible if and only if the secret key vector and the ciphertext vector are equals at all positions different from the wildcard $\star$.
In the sequel, we first give the definition of identity-based encryption with wildcards and its properties, such as the *anonymity* property that states that a ciphertext does not reveal any information about its associated pattern. In Section 4.2 we present a generic construction of an anonymous WIBE scheme, proposed by Abdalla *et al.* [2]. Then in Section 4.2.2 we present two of our contributions which are two new security properties for identity-based encryption with wildcards schemes, called *privacy-preserving key generation* and *pattern-hiding* security: the former transforms the key generation of a WIBE into an interactive protocol between a key generation center (KGC), a pattern audit center (PAC) and a user that is requesting a secret key, and states the KGC center does not learn any information about the user's pattern when creating a key; the latter is an extension of the anonymity property. Finally, in Section 4.3 we propose three new identity-based encryption with wildcards instantiations: the first scheme is a normal WIBE scheme with constant size ciphertext, the second scheme is pattern-hiding and the last scheme has privacy-preserving key generation.

4.1 Identity-Based Encryption with Wildcards (WIBE)

4.1.1 Definitions

Definition 4.1.1 ***Pattern*** *[4, 85]. A pattern $\boldsymbol{P}$ is a vector $(P_1, \cdots, P_L) \in \mathcal{U}^L$, where $\mathcal{U}$ is a set with a special wildcard symbol "$\star$", and $L \in \mathbb{N}$.*

Let $\boldsymbol{P}' = (P'_1, \cdots, P'_L)$ *and* $\boldsymbol{P} = (P_1, \cdots, P_L)$ *be two patterns.* $\boldsymbol{P}'$ belongs *to* $\boldsymbol{P}$*, denoted* $\boldsymbol{P}' \in_\star \boldsymbol{P}$*, if and only if* $\forall i \in \{1, \cdots, L\}$*,* $(P'_i = P_i) \vee (P_i = \star)$*.* $\boldsymbol{P}'$ matches $\boldsymbol{P}$*, denoted* $\boldsymbol{P}' =_\star \boldsymbol{P}$*, if and only if* $\forall i \in \{1, \cdots, L\}$*,* $(P'_i = P_i) \vee (P_i = \star) \vee (P'_i = \star)$*. Notice that if* $\boldsymbol{P}' \in_\star \boldsymbol{P}$ *then* $\boldsymbol{P}' =_\star \boldsymbol{P}$*.*

Notation 4.1.1 *For a pattern* $\boldsymbol{P} \in \mathcal{U}^L$*,* $W(\boldsymbol{P})$ *denoted the set of all indices* $i \in \{1, \cdots, L\}$ *such that* $P_i = \star$*, and* $\bar{W}(\boldsymbol{P})$ *is the complementary set. Clearly* $W(\boldsymbol{P}) \cap \bar{W}(\boldsymbol{P}) = \varnothing$ *and* $W(\boldsymbol{P}) \cup \bar{W}(\boldsymbol{P}) = \{1, \cdots, L\}$*.*

Definition 4.1.2 ***Identity-based encryption with wildcards*** *(WIBE) [4, 85]. An identity-based encryption with wildcards scheme consists of four algorithms:*

- Setup(λ, L)*: the setup algorithm takes as input a security parameter* $\lambda \in \mathbb{N}$ *and pattern length* $L \in \mathbb{N}$*. It outputs a public key pk and a master secret key msk.*
- KeyDer$(\mathsf{msk}, \boldsymbol{P})$*: the key derivation algorithm takes as input a master secret key msk and a pattern* $\boldsymbol{P}$ *and creates a secret key* $\mathsf{sk}_{\boldsymbol{P}}$ *for* $\boldsymbol{P}$*. It can also take as input a secret key* $\mathsf{sk}_{\boldsymbol{P}'}$ *for a pattern* $\boldsymbol{P}'$ *instead of msk and then derives a secret key for any pattern* $\boldsymbol{P} \in_\star \boldsymbol{P}'$*.*
- Encrypt$(\mathsf{pk}, \boldsymbol{P}, \mathsf{m})$*: the encryption algorithm takes as input a public key pk, a pattern* $\boldsymbol{P}$ *and a message m. It outputs ciphertext ct of message m, for pattern* $\boldsymbol{P}$*.*
- Decrypt$(\mathsf{sk}_{\boldsymbol{P}}, \mathsf{ct}, \boldsymbol{P}')$*: the decryption algorithm takes as input a user secret key* $\mathsf{sk}_{\boldsymbol{P}}$ *for a pattern* $\boldsymbol{P}$ *and a ciphertext ct for a pattern* $\boldsymbol{P}'$*, and it returns a message* m'*.*

Definition 4.1.3 ***Correctness*** *[85]. An identity-based encryption with wildcards scheme is said to be* correct *if for all security parameter* $\lambda \in \mathbb{N}$*, all integer* $L \in \mathbb{N}$*, every honestly generated key pair* $(pk, msk) \leftarrow \mathsf{Setup}(\lambda, L)$*, every messages m, and every patterns* $\boldsymbol{P}, \boldsymbol{P}' \in \mathcal{U}^L$*, such that* $\boldsymbol{P}' =_\star \boldsymbol{P}$*:*

$$\Pr\Big[\mathsf{Decrypt}(\mathsf{KeyDer}(\mathsf{msk}, \boldsymbol{P}'), \mathsf{Encrypt}(\mathsf{pk}, \boldsymbol{P}, \mathsf{m})) = \mathsf{m}\Big] = 1.$$

Definition 4.1.4 ***Adaptive indistinguishability security*** *(IND-WIBE) [4, 85]. An identity-based encryption with wildcards scheme is said to satisfy* adaptive indistinguishability *security if all PPT adversaries* $\mathcal{A}$ *have at most negligible advantage in winning the game presented in Figure 4.1, where* $\mathcal{A}$*'s advantage is defined as* $\mathsf{Adv}_{\mathcal{A}}^{IND-WIBE}(\lambda) := \Pr\left[b' = b\right] - 1/2$ *for any* $\lambda \in \mathbb{N}$*.*

SETUP: challenger $\mathcal{C}$ on input (λ, L) runs Setup(λ, L) to generate pk and msk, and gives pk to $\mathcal{A}$.
KEY QUERY: adversary $\mathcal{A}$ may adaptively query a key for pattern $\boldsymbol{P}$. In response, $\mathcal{A}$ is given the corresponding secret key $\mathsf{sk}_{\boldsymbol{P}} \leftarrow$ KeyGen$(\mathsf{msk}, \boldsymbol{P})$.
CHALLENGE: $\mathcal{A}$ chooses challenge pattern $\boldsymbol{P}^*$ and challenge messages $\mathsf{m}_0, \mathsf{m}_1$, and sends them to $\mathcal{C}$. The latter picks a random bit b and gives to $\mathcal{A}$ $\mathsf{ct}^* \leftarrow$ Encrypt$(\mathsf{pk}, \boldsymbol{P}^*, \mathsf{m}_b)$.
KEY QUERY: The adversary may continue to issue key queries for additional pattern $\boldsymbol{P}$. $\mathcal{A}$ is given the corresponding key $\mathsf{sk}_{\boldsymbol{P}} \leftarrow$ KeyGen$(\mathsf{msk}, \boldsymbol{P})$.
GUESS: $\mathcal{A}$ outputs a bit b', and wins the game if $b' = b$ and if, for all $\boldsymbol{P}$ for which a key was queried, the condition $\boldsymbol{P} \neq_\star \boldsymbol{P}^*$ holds.

Figure 4.1: Adaptive indistinguishability security game for identity-based encryption with wildcards schemes.

The next definition presents the anonymous security property of WIBE schemes, introduced by [2], that states that it is hard for an adversary given a ciphertext to guess which previously chosen message was encrypted.

Definition 4.1.5 ***Adaptive anonymous security*** *(ANO-WIBE) [2]. An identity-based encryption with wildcards scheme is said to satisfy* adaptive anonymous *security if all PPT adversaries $\mathcal{A}$ have at most negligible advantage in winning the game presented in Figure 4.2, where $\mathcal{A}$'s advantage is defined as* $\mathsf{Adv}_{\mathcal{A}}^{ANO-WIBE}(\lambda) := \Pr\left[b' = b\right] - 1/2$ *for any $\lambda \in \mathbb{N}$.*

SETUP: challenger $\mathcal{C}$ on input (λ, L) runs Setup(λ, L) to generate keys pk and msk, and gives pk to $\mathcal{A}$.
KEY QUERY: adversary $\mathcal{A}$ may adaptively query a key for pattern $\boldsymbol{P}$. In response, $\mathcal{A}$ is given the corresponding secret key $\mathsf{sk}_{\boldsymbol{P}} \leftarrow$ KeyGen$(\mathsf{msk}, \boldsymbol{P})$.
CHALLENGE: $\mathcal{A}$ chooses two challenge patterns $\boldsymbol{P}^0, \boldsymbol{P}^1$ and challenge messages $\mathsf{m}_0, \mathsf{m}_1$, and sends them to $\mathcal{C}$. The latter picks a random bit b and gives to $\mathcal{A}$ $\mathsf{ct}_b \leftarrow$ Encrypt$(\mathsf{pk}, \boldsymbol{P}^b, \mathsf{m}_b)$.
KEY QUERY: The adversary may continue to issue key queries for additional pattern $\boldsymbol{P}$. $\mathcal{A}$ is given the corresponding key $\mathsf{sk}_{\boldsymbol{P}} \leftarrow$ KeyGen$(\mathsf{msk}, \boldsymbol{P})$.
GUESS: $\mathcal{A}$ outputs a bit b', and wins if $b' = b$ and if, for all pattern $\boldsymbol{P}$ for which a key was queried, the condition $\boldsymbol{P} \neq_\star \boldsymbol{P}^0$ and $\boldsymbol{P} \neq_\star \boldsymbol{P}^1$ holds.

Figure 4.2: Adaptive anonymous security game for identity-based encryption with wildcards schemes.

Note 4.1.1 *In the anonymous security game for identity-based encryption with wildcards, the adversary is allowed to query secret keys that do not decrypt the challenge*

ciphertext. In Section 4.2.2 we present a stronger security notion in which the adversary is allowed to query secret keys that decrypt the challenge ciphertext, at the condition that decryption is possible no matter the challenge pattern chosen.

4.1.2 Generic Construction of Anonymous WIBE Scheme

Here we present a generic construction, proposed by Abdalla *et al.* [2], of anonymous identity-based encryption with wildcards scheme, from inner product encryption (see Section 3.3). Briefly, in an IPE scheme secret keys and ciphertexts are associated to a vector, and decryption is possible if the inner product of the secret key's vector and the ciphertext's vector is equal to 0. For more details on this primitive, refer to Section 3.3.

In the construction, patterns belong to the set $\{0, 1, \star\}$. The key idea of the construction is to double the pattern length and simulate wildcard positions with 0 positions in the IPE vector. To do so they introduce two algorithms that we will call ExtendingKeyPattern and ExtendingCtPattern, that work as follows.

Algorithm 4.1 ExtendingKeyPattern

Input: key pattern $\boldsymbol{P}$ of length n
Output: pattern $\boldsymbol{u}$ of length $2n$

$i \leftarrow 1, j \leftarrow 1$
while $i \leq n, j \leq 2n$ **do**
  if $P_i \neq \star$ **then**
    $u_j \leftarrow 1$ and $u_{j+1} \leftarrow P_i$
  else
    $u_j \leftarrow 0$ and $u_{j+1} \leftarrow 0$
  end if
  $j \leftarrow j + 2, i \leftarrow i + 1$
end while
return $\boldsymbol{u}$

Algorithm 4.2 ExtendingCtPattern

Input: ciphertext pattern $\boldsymbol{P}$ of length n
Output: pattern $\boldsymbol{v}$ of length $2n$

$i \leftarrow 1, j \leftarrow 1$
while $i \leq n, j \leq 2n$ **do**
  if $P_i \neq \star$ **then**
    $v_j \leftarrow -r_i \cdot P_i, v_{j+1} \leftarrow r_i$ for $r_i \leftarrow \mathbb{Z}_p$
  else
    $v_j \leftarrow 0$ and $v_{j+1} \leftarrow 0$
  end if
  $j \leftarrow j + 2, i \leftarrow i + 1$
end while
return $\boldsymbol{v}$

We present in Figure 4.3 how both algorithm work on an example.

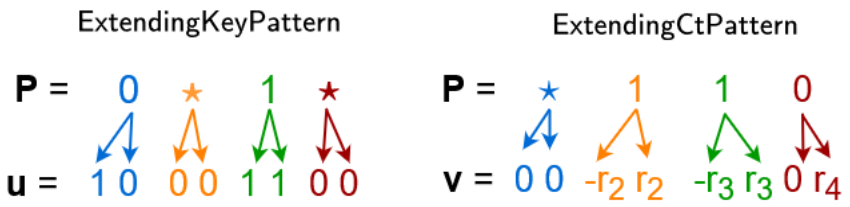


Figure 4.3: ExtendingKeyPattern and ExtendingCtPattern on a example.

We now present the construction in Figure 4.4.

Let $IPE = (Setup, KeyGen, Encrypt, Decrypt)$ be an inner product encryption scheme. We construct WIBE $=$ (Setup, KeyGen, Encrypt, Decrypt) an identity-based encryption with wildcards scheme from IPE.

- Setup(λ, L): the setup algorithm runs $Setup(\lambda, 2L)$ to get $(\mathsf{msk}, \mathsf{pk})$ and outputs them.
- KeyGen$(\mathsf{msk}, \boldsymbol{P})$: the key generation algorithm creates the pattern $\boldsymbol{u}$ of length $2L$ by running ExtendingKeyPattern on input $\boldsymbol{P}$. Then it returns $\mathsf{sk}_{\boldsymbol{P}} \leftarrow KeyGen(\mathsf{msk}, \boldsymbol{u})$.
- Encrypt$(\mathsf{pk}, \boldsymbol{P}', \mathsf{m})$: the encryption algorithm creates the pattern $\boldsymbol{v}$ of length $2L$ by running ExtendingCtPattern on input $\boldsymbol{P}'$. The encryption algorithm returns $\mathsf{ct} \leftarrow Encrypt(\mathsf{pk}, \boldsymbol{v}, \mathsf{m})$.
- Decrypt$(\mathsf{sk}_{\boldsymbol{P}}, \mathsf{ct}, \boldsymbol{P}')$: the decryption algorithm returns $Decrypt(\mathsf{sk}_{\boldsymbol{P}}, \mathsf{ct})$.

Figure 4.4: Generic construction of anonymous identity-based encryption with wildcards scheme from inner product encryption scheme.

Regarding correctness and security Abdalla *et al.* [2] prove that if the underlying IPE scheme is correct and weak attribute-hiding, then the obtained WIBE scheme is respectively correct and anonymous. We refer the interested reader to [2]'s work for more details.

Note 4.1.2 *Actually to be able to make this generic construction, Abdalla* et al. *[2] introduced a new kind of IPE schemes: inner product encryption scheme with generalized key delegation. This new feature of IPE schemes is needed in order to obtain the key delegation algorithm of the WIBE scheme. In this thesis, we will use WIBE scheme without key delegation, therefore we omit in our presentation of the generic construction the delegation part. We also rewrite* KeyDer *algorithm by* KeyGen *in the WIBE scheme for the same reason.*

4.2 Our Contributions to WIBE: New Security Properties

We now present two of our contributions regarding identity-based encryption with wildcards, which are two new security properties. The first one, that we call *privacy-preserving key generation* protects the privacy of pattern used for key generation while the second, called *pattern-hiding* is an extension of the anonymous property.

4.2.1 Introducing Privacy-Preserving Key Generation WIBE

When looking at identity-based encryption with wildcards security properties, we notice that that they either protects the message encrypted (with indistinguishability security, see Definition 4.1.4) or the pattern associated to the ciphertext (anonymous security, see Definition 4.1.5). For future applications, such as one presented in Section 6.3, we decided to introduce a new security property that protects the privacy of the pattern during key generation. In other words, we need a WIBE scheme in which the authority does not learn any information about patterns associated to secret keys.

As such property cannot be included as is in the previously given definition of WIBE (Definition 4.1.2), we need to define a new kind of WIBE: privacy-preserving key generation WIBE scheme. The below definition is based on the work that has been done on e.g., blind signatures [47, 121], or privacy-preserving key generation (PPKG) ABE [131]. In this definition the generation of decryption keys is transform into an interactive protocol between a key generation center (KGC) (that knows the master secret key), a pattern certification center (PAC) and a user, in such a way that the user finally obtain the secret key $\mathsf{sk}_{\mathbf{P}}$, while each party protects its own input. In the next definition, the name of key generation protocol algorithms that are ran by KGC are written in blue, those ran by user in green and in orange those that are run by PAC.

Note 4.2.1 *In the following we use the notation $\tilde{msk}$ to denote a part of msk, given as an optional input some algorithms. Notice that $\tilde{msk}$ is not enough to generate secret keys and might be equals to $\varnothing$ in some cases.*

Definition 4.2.1 ***WIBE scheme with privacy-preserving key generation.*** *(PPKG-WIBE) An identity-based encryption with wildcards scheme with* privacy-preserving key generation *consists of seven algorithms:*

- Setup(λ, L)*: the setup algorithm, run by KGC, takes as input a security parameter $\lambda \in \mathbb{N}$ and the maximal pattern length L. It outputs a public key pk and a master secret key msk.*
- UserTemKeyGen(pk)*: the user's temporary key generation algorithm is run by user. It takes public key pk as input and outputs user's temporary public key tpk_{user} and user's temporary secret key tsk_{user}.*
- BlindTokenGen$(\tilde{\mathsf{msk}}, \mathsf{pk}, \mathbf{P}, \mathsf{tpk}_{user})$*: the blind token generation algorithm is run by PAC. It takes pk, user's pattern $\mathbf{P}$, and user's temporary public key tpk_{user} as input and outputs a blind token $\mathsf{bt}_{\mathbf{P}}$ for pattern $\mathbf{P}$.*

- BlindKeyGen$(\mathsf{msk}, \mathsf{pk}, \mathsf{bt}_{\mathbf{P}})$*: the blind key generation algorithm is run by KGC. It takes msk, pk, and user's blind token* $\mathsf{bt}_{\mathbf{P}}$ *as input and outputs blind secret key bsk for pattern* $\mathbf{P}$*.*
- KeyExtract$(\mathsf{bsk}_{\mathbf{P}}, \mathsf{tsk}_{user})$*: the key extract algorithm is run by user locally. It takes blind secret key* $\mathsf{bsk}_{\mathbf{P}}$ *and user's temporary secret key* tsk_{user} *as input and outputs the final secret key sk for pattern* $\mathbf{P}$*.*
- Encrypt$(\mathsf{pk}, \mathbf{P}', \mathsf{m})$*: the encryption algorithm takes as input the public key pk, a pattern* $\mathbf{P}'$ *and a message m. It outputs ciphertext ct for pattern* $\mathbf{P}'$*.*
- Decrypt$(\mathsf{sk}_{\mathbf{P}}, \mathsf{ct}, \mathbf{P}')$*: the decryption algorithm takes as input a user secret key* $\mathsf{sk}_{\mathbf{P}}$ *for a pattern* $\mathbf{P}$ *and a ciphertext ct for a pattern* $\mathbf{P}'$*, and returns a message* m'*.*

We present in Figure 4.5 the interactive protocol of the key generation.

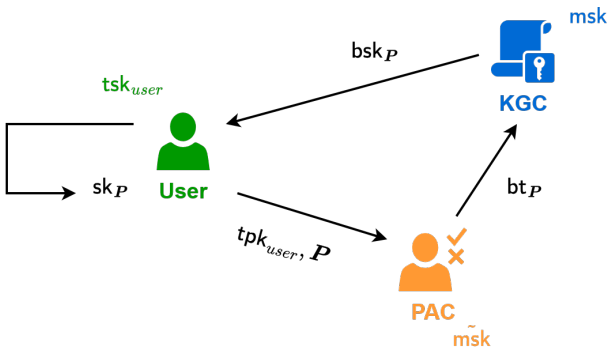


Figure 4.5: The key generation interactive protocol.

With this new definition of WIBE, we easily notice that if we want to protect KGC from learning pattern $\boldsymbol{P}$ for which a key is queried we require that the output of BlindTokenGen does not leak any information about the pattern. We formalize this requirement with the following security definition that states that no adversary can distinguish a blind token for pattern $\boldsymbol{P}^0$ from a blind token for pattern $\boldsymbol{P}^1$, where $\boldsymbol{P}^0, \boldsymbol{P}^1$ are chosen by the adversary.

Definition 4.2.2 ***Adaptive privacy-preserving key generation WIBE security.*** *We define the following oracles: (i) Blind Token Oracle* $\mathcal{O}_{BT}(\mathbf{P}')$ *that takes as input pattern* $\mathbf{P}'$ *and outputs corresponding blind token* $\mathsf{bt}_{\mathbf{P}'}$ *and (ii)* $\mathcal{O}_{BK}(\mathsf{bt}_{\mathbf{P}'})$ *that takes as input blind token* $\mathsf{bt}_{\mathbf{P}'}$ *and outputs corresponding blind key* $\mathsf{bsk}_{\mathbf{P}'}$*. A WIBE scheme is said to satisfy adaptive* privacy-preserving key generation security *if for any PPT adversary* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *to win the game presented in Figure 4.6 is negligible. Let* $\mathcal{C}$ *be a challenger.*

For our future use of such a WIBE, KGC will be playing the adversary's role.

SETUP: $\mathcal{C}$ on input (λ, L) runs Setup(λ, L) to get pk, msk and gives pk to $\mathcal{A}$.
TOKEN or KEYQUERY: $\mathcal{A}$ queries oracles $\mathcal{O}_{BT}$ and $\mathcal{O}_{BK}$ freely.
CHALLENGE: $\mathcal{A}$ submits two patterns $\mathbf{P}^0$ and $\mathbf{P}^1$ where $|\mathbf{P}^0| = |\mathbf{P}^1|$. $\mathcal{C}$ chooses $b \in \{0,1\}$ randomly and queries $\mathcal{O}_{BT}$ on input $\mathbf{P}^b$. It returns blind token $\text{bt}_{\mathbf{P}^b}$ to $\mathcal{A}$.
TOKEN or KEYQUERY: $\mathcal{A}$ queries oracles $\mathcal{O}_{BT}$ and $\mathcal{O}_{BK}$ freely.
GUESS: $\mathcal{A}$ outputs its guess b' and wins if $b = b'$.

Figure 4.6: Adaptive privacy-preserving key generation security game for PPKG-WIBE schemes.

4.2.2 New Security Property of Pattern-Hiding

In this section, we introduce a new security property for WIBE schemes: pattern-hiding security.

Definition 4.2.3 ***Adaptive pattern-hiding security*** *(PH-WIBE). An identity-based encryption with wildcards scheme is said to satisfy* adaptive pattern-hiding *security (or is* adaptively pattern-hiding*) if all PPT adversaries $\mathcal{A}$ have at most negligible advantage in winning the game presented in Figure 4.7, where $\mathcal{A}$'s advantage is defined as* $\text{Adv}_{\mathcal{A}}^{PH-WIBE}(\lambda) := \Pr\left[b' = b\right] - 1/2$ *for any $\lambda \in \mathbb{N}$. Let $\mathcal{C}$ be a challenger.*

SETUP: $\mathcal{C}$ on input (λ, L) runs Setup(λ, L) to generate keys pk and msk, and gives pk to $\mathcal{A}$.
KEY QUERY: adversary $\mathcal{A}$ may adaptively query a key for pattern $\boldsymbol{P}$. In response, $\mathcal{A}$ is given the corresponding secret key $\text{sk}_{\boldsymbol{P}} \leftarrow$ KeyGen(msk, $\boldsymbol{P}$).
CHALLENGE: $\mathcal{A}$ chooses two challenge patterns $\boldsymbol{P}^0, \boldsymbol{P}^1$ and challenge message m, and sends them to $\mathcal{C}$. The latter picks a random bit b and gives to $\mathcal{A}$ $\text{ct}_b \leftarrow$ Encrypt(pk, $\boldsymbol{P}^b$, m).
KEY QUERY: The adversary may continue to issue key queries for additional pattern $\boldsymbol{P}$, and is given the corresponding key $\text{sk}_{\boldsymbol{P}} \leftarrow$ KeyGen(msk, $\boldsymbol{P}$).
GUESS: $\mathcal{A}$ outputs a bit b', and wins if $b' = b$ and if, for all $\boldsymbol{P}$ for which a key was queried, the condition $\boldsymbol{P} =_\star \boldsymbol{P}^0 \wedge \boldsymbol{P} =_\star \boldsymbol{P}^1$ or $\boldsymbol{P} \neq_\star \boldsymbol{P}^0 \wedge \boldsymbol{P} \neq_\star \boldsymbol{P}^1$ holds.

Figure 4.7: Adaptive pattern-hiding security game for identity-based encryption with wildcards schemes.

We now establish the following theorem, that presents the relation between our new security property and existing the one of anonymity.

Theorem 4.2.1 *Pattern-hiding security implies anonymous security for WIBE schemes.*

Proof 4.2.1 *We prove the contrapositive. Let $\mathcal{B}$ be an adversary against anonymous security, that wins with non negligible advantage ϵ and let $\mathcal{C}$ be a challenger for pattern-hiding security. We construct, in Figure 4.8, an adversary $\mathcal{A}$ that wins the pattern-hiding security game, by using $\mathcal{B}$. Let $\mathcal{C}$ be a challenger.*

- `SETUP`: $\mathcal{C}$ on input (λ, L) runs Setup(λ, L) to get pk, msk, and sends pk to $\mathcal{A}$ and $\mathcal{B}$.
- `KEY QUERY`: adversary $\mathcal{B}$ chooses a pattern $\boldsymbol{P}$ for which she requires a key and sends it to $\mathcal{A}$. The latter sends $\boldsymbol{P}$ to $\mathcal{C}$ who answers with $\mathsf{sk}_{\boldsymbol{P}} \leftarrow$ KeyGen$(\mathsf{msk}, \boldsymbol{P})$.
- `CHALLENGE`: $\mathcal{B}$ chooses two challenge patterns $\boldsymbol{P}^0, \boldsymbol{P}^1$ along with challenge messages $\mathsf{m}_0, \mathsf{m}_1$ and sends them to $\mathcal{A}$. The latter picks $\tilde{b} \leftarrow \{0,1\}$ and sends to $\mathcal{C}$ patterns $\boldsymbol{P}^0, \boldsymbol{P}^1$ and message $\mathsf{m}_{\tilde{b}}$. $\mathcal{C}$ chooses $b \leftarrow \{0,1\}$ and returns $\mathsf{ct}_b \leftarrow$ Encrypt$(\mathsf{pk}, \boldsymbol{P}^b, \mathsf{m}_{\tilde{b}})$ to $\mathcal{A}$, who sends it to $\mathcal{B}$.
- `KEY QUERY`: $\mathcal{B}$ may continue to issue key queries for additional pattern $\boldsymbol{P}$. $\mathcal{A}$ queries $\mathcal{C}$ and returns to $\mathcal{B}$ the answer of $\mathsf{sk}_{\boldsymbol{P}} \leftarrow$ KeyGen$(\mathsf{msk}, \boldsymbol{P})$.
- `GUESS`: $\mathcal{B}$ outputs a bit b' to $\mathcal{A}$, who outputs it as her own guess.

Figure 4.8: Construction of PH-WIBE adversary from ANO-WIBE adversary.

$\mathcal{B}$ is an admissible *adversary, meaning that she will always output a key for pattern $\boldsymbol{P}$ such that $\boldsymbol{P} \neq_\star \boldsymbol{P}^0$ and $\boldsymbol{P} \neq_\star \boldsymbol{P}^1$. Thus the pattern-hiding security game restrictions are respected.*
Let us evaluate $\mathcal{A}$'s advantage. If $\tilde{b} = b$, then $\mathcal{B}$ is given a challenge ciphertext as expected and thus wins the security game with advantage ϵ. If $\tilde{b} \neq b$, as the challenge ciphertext is not of the form expected by $\mathcal{B}$ (for example she can receive the result of Encrypt$(\mathsf{pk}, \boldsymbol{P}^0, \mathsf{m}_1)$*), her advantage in winning the security game is equivalent as making a random guess, thus is equal to $1/2$. Therefore,* $\mathsf{Adv}_{\mathcal{A}}^{PH-WIBE}(\lambda) = \epsilon - 1/2$ *which is non negligible.* □

4.3 Another Contribution: Our New WIBE Instantiations

In this section, we present three identity-based encryption with wildcards schemes. The first scheme has constant-size ciphertext but does not provide the pattern-hiding property, while the second scheme does not have constant-size ciphertext but is proved to satisfy pattern-hiding. The third scheme is a privacy-preserving key generation WIBE.

Our three identity-based encryption with wildcards schemes have the particularity that they do not allow key derivation for a pattern from another pattern's key (thus KeyDer

algorithm will be written KeyGen). Plus, we restrict the first and the second scheme key pattern space to $\{0,1\}^L \backslash \{\mathbf{0}^L\}$ and ciphertext pattern spaces to $\{0,\star\}^L \backslash \{\mathbf{0}^L\}$ for the first scheme and to $\{0,\star\}^L \backslash \{\star^L\}$ for our second scheme. These restrictions are required for a future use of our WIBE schemes as building blocks for data sharing primitives. Notice that with this key pattern space, the decryption condition now requires that the key pattern *belongs* (*i.e.* $\in_\star$) to the ciphertext pattern and no longer that both patterns *match* (*i.e.* $=_\star$). Also, notice that our two instantiations are made for these restrictions specifically therefore they do not work without them. For the third instantiation, patterns for both keys and ciphertexts belong to $\{0,1,\star\}^L$.

Let $\boldsymbol{P} \in \{0,1\}^L \backslash \{\mathbf{0}^L\}$ be a pattern. We set $\mathcal{I} = \{i \in [L] | P_i = 1\}$ and $\mathcal{O} = \{i \in [L] | P_i = 0\}$; notice that $[L] = \mathcal{I} \cup \mathcal{O}$.

4.3.1 A WIBE Scheme With Constant Size Ciphertext

Intuition. Kim *et al.* [86] proposed an identity-based encryption with wildcards, in the symmetric prime order bilinear group setting. Their scheme has constant size ciphertext and satisfies selective IND-WIBE-CPA security under the ℓ-BDHE problem. We first adapt it to the keys and ciphertexts patterns we are interested in, i.e. respectively $\{0,1\}^L \backslash \{\mathbf{0}^L\}$, and $\{0,\star\}^L \backslash \{\mathbf{0}^L\}$ and moved it to asymmetric prime order bilinear group setting. This results in a new scheme, with shorter secret keys and ciphertexts. Then we modified our scheme in order to obtain adaptive security. To do so, we followed [86]'s idea to use composite order groups and the dual system encryption framework. Finally, we moved our scheme from the composite order bilinear group setting to the prime order bilinear group setting, for efficiency and security reasons. We were able to do this change thanks to the use of dual pairing vectors spaces and following the works of Lewko [90] and Chen *et al.* [51].
An important point to notice here is in the Setup algorithm: in [86]'s schemes, random groups elements h_i (for $i = 1, \cdots, L$) are generated by the scheme authority and put in the public key. In our scheme, the authority will pick random elements a_i (for $i = 1, \cdots, L$) in $\mathbb{Z}_p$, sets $h_i = g_1^{a_i}$ and gives in the public key $h_i^{\boldsymbol{d}_2}$, for $\boldsymbol{d}_2 \in \mathbb{D}$ for $\mathbb{D}$ one base of the DPVS. This change was necessary for the security proof as the adversary against the security assumption, that simulates the scheme's authority receives $\boldsymbol{d}_2$ in the exponent of g_1 which means she cannot compute $h_i^{\boldsymbol{d}_2}$ if h_i is not of the form $g_1^{a_i}$ for a_i known by the adversary.

This gives us our first WIBE scheme, presented in Figure 4.9. With the restrictions on key and ciphertext pattern spaces and the notations we present above, we can rewrite the decryption condition as follows: for patterns $\boldsymbol{P}, \boldsymbol{P}^*$ in respectively $\{0,1\}^L \backslash \{\mathbf{0}^L\}$

- Setup(λ, L): generate an asymmetric bilinear pairing group $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ for prime order p. Sample random dual orthonormal bases $(\mathbb{D}, \mathbb{D}^*) \leftarrow$ Dual$(\mathbb{Z}_p^4)$. Let $\boldsymbol{d}_1, \cdots, \boldsymbol{d}_4$ denote the elements of $\mathbb{D}$ and $\boldsymbol{d}_1^*, \cdots, \boldsymbol{d}_4^*$ denote the elements of $\mathbb{D}^*$. Pick $\alpha, a_1, \cdots, a_L \leftarrow \mathbb{Z}_p$. The public key is computed as: pk $= (\Gamma, e(g_1, g_2)^{\alpha \boldsymbol{d}_1 . \boldsymbol{d}_1^*}, g_1^{\boldsymbol{d}_1}, \boldsymbol{h}_1 = g_1^{a_1 \cdot \boldsymbol{d}_2}, \cdots, \boldsymbol{h}_L = g_1^{a_L \cdot \boldsymbol{d}_2})$ and the master secret key is msk $= (\alpha, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_2^*}, a_1, \cdots, a_L)$.
- KeyGen(msk, $\boldsymbol{P}'$): pick $r \leftarrow \mathbb{Z}_p$. Compute $\boldsymbol{\beta} = g_2^{\alpha \boldsymbol{d}_1^* + r \cdot \sum\limits_{i \in \mathcal{I}} a_i \cdot \boldsymbol{d}_1^* - r \cdot \boldsymbol{d}_2^*}$ and $\boldsymbol{v}_i = g_2^{r \cdot a_i \cdot \boldsymbol{d}_1^*}$ for $i \in \mathcal{O}$. The secret key is $\mathsf{sk}_{\boldsymbol{P}'} = (\boldsymbol{\beta}, \{\boldsymbol{v}_i\}_{i \in \mathcal{O}})$.
- Encrypt(pk, $\boldsymbol{P}$, m $\in \mathbb{G}_T$): choose $s \leftarrow \mathbb{Z}_p$ and compute ct $= (c_1, \boldsymbol{c}_2)$ where $c_1 = \mathsf{m} \cdot (e(g_1, g_2)^{\alpha \boldsymbol{d}_1^* \cdot \boldsymbol{d}_1})^s$, $\boldsymbol{c}_2 = g_1^{s \boldsymbol{d}_1} \cdot \prod\limits_{i \in W(\boldsymbol{P})} \boldsymbol{h}_i^s = g_1^{s \boldsymbol{d}_1 + s \boldsymbol{d}_2 \sum\limits_{i \in W(\boldsymbol{P})} a_i}$.
- Decrypt($\mathsf{sk}_{\boldsymbol{P}'}$, ct, $\boldsymbol{P}$): compute $\boldsymbol{\beta}' = \boldsymbol{\beta} \prod\limits_{i \in W(\boldsymbol{P}) \cap \mathcal{O}} \boldsymbol{v}_i$ and finally $c_1 \cdot \frac{1}{e(\boldsymbol{c}_2, \boldsymbol{\beta}')}$.

Figure 4.9: Our identity-based encryption with wildcards scheme in prime order group, with constant size ciphertext and adaptive security.

and $\{0, \star\}^L \backslash \{\mathbf{0}^L\}$, we have that $\boldsymbol{P} \in_\star \boldsymbol{P}^* \implies \forall i \in [L]$, if $P = 1$ then $P_i^* = \star$ and thus $\mathcal{I} \subseteq W(\boldsymbol{P}^*)$.

Our Scheme. We now present in Figure 4.9 our new identity-based encryption with wildcards scheme.

Theorem 4.3.1 *Our first WIBE scheme is correct.*

Proof 4.3.1 *We have that*

$$\begin{aligned} e(\boldsymbol{c}_2, \boldsymbol{\beta}') &= e\left(g_1^{s \boldsymbol{d}_1} . \prod_{i \in W(\boldsymbol{P})} \boldsymbol{h}_i^s, g_2^{\alpha \boldsymbol{d}_1^* + r . \sum\limits_{i \in \mathcal{I}} a_i . \boldsymbol{d}_1^* - r . \boldsymbol{d}_2^*} \cdot \prod_{i \in W(\boldsymbol{P}) \cap \mathcal{O}} g_2^{r \boldsymbol{d}_1^* a_i} \right) \\ &= e\left(g_1^{s \boldsymbol{d}_1}, g_2^{\alpha \boldsymbol{d}_1^*} \right) . e\left(g_1^{s \boldsymbol{d}_1}, g_2^{r . \boldsymbol{d}_1^* (\sum\limits_{i \in \mathcal{I}} a_i + \sum\limits_{i \in W(\boldsymbol{P}) \cap \mathcal{O}} a_i)} \right) . e\left(g_1^{s \boldsymbol{d}_2 \sum\limits_{i \in W(\boldsymbol{P})} a_i}, g_2^{-r . \boldsymbol{d}_2^*} \right) \end{aligned}$$

as thanks to dual vector spaces properties: $e\left(g_1^{s \boldsymbol{d}_1}, g_2^{-r \boldsymbol{d}_2^*}\right) = e(g_1, g_2)^0$ *and*

$$e\left(g_1^{s \boldsymbol{d}_2 \sum\limits_{i \in W(\boldsymbol{P})} a_i}, g_2^{\alpha \boldsymbol{d}_1^* + r . \sum\limits_{i \in \mathcal{I}} a_i . \boldsymbol{d}_1^* + \sum\limits_{i \in W(P) \cap \mathcal{O}} r . a_i . \boldsymbol{d}_1^*} \right) = e(g_1, g_2)^0 = 1.$$

The first pairing is equal to $(e(g_1, g_2)^{\alpha \boldsymbol{d}_1 . \boldsymbol{d}_1^*})^s$ *which will canceled with the element of* c_1*.*

The second pairing is equal to $e(g_1, g_2)^{sr\psi(\sum_{i\in\mathcal{I}} a_i + \sum_{i\in W(\boldsymbol{P})\cap\mathcal{O}} a_i)}$ *and the third pairing is equal to* $e(g_1, g_2)^{-rs\psi \sum_{i\in W(\boldsymbol{P})} a_i}$.
As user is allowed to decrypt then $\mathcal{I} \subseteq W(\boldsymbol{P})$*, thus we can rewrite* $\mathcal{I}$ *as* $\mathcal{I} \cap W(\boldsymbol{P})$ *and we have that* $\sum_{i\in\mathcal{I}} a_i + \sum_{i\in W(\boldsymbol{P})\cap\mathcal{O}} a_i = \sum_{i\in W(\boldsymbol{P})\cap(\mathcal{I}\cup\mathcal{O})} a_i = \sum_{i\in W(\boldsymbol{P})} a_i$*. Therefore multiplying the two last pairings gives* 1*, and user can decrypt.* □

Security. The following theorem establishes the security of our new WIBE scheme.

Theorem 4.3.2 *If SXDH holds then our scheme satisfies adaptive* IND-WIBE-CPA*.*

Our proof is based on the ones of [90] (Section 4.6) and [51] (Section 4) and is done with the dual system encryption framework. [90]'s proof is done in the symmetric pairing setting but moving from symmetric pairings to asymmetric pairings is not an issue if elements are taken in the correct group ($\mathbb{G}_1$ for ciphertext and public key elements, and $\mathbb{G}_2$ for secret keys elements).

Let us define semi-functional keys and semi-functional ciphertexts that we will use in the proof. Let $\mathsf{sk} = (\boldsymbol{\beta}, \{\boldsymbol{v}_i\}_{i\in\mathcal{O}})$ be a normal key, and $t_3, t_4, \{t_{b,i}\}_{i\in\mathcal{O}}$ be random elements of $\mathbb{Z}_p$. We define a semi-functional key as $\mathsf{sk}' = (\boldsymbol{\beta}', \{\boldsymbol{v}_i'\}_{i\in\mathcal{O}})$ where $\boldsymbol{\beta}' = \boldsymbol{\beta} \cdot g_2^{t_3\cdot\boldsymbol{d}_3^* + t_4\cdot\boldsymbol{d}_4^*}$ and $\boldsymbol{v}_i' = \boldsymbol{v}_i \cdot g_2^{t_{b,i}\cdot\boldsymbol{d}_3^*}$ for $i \in \mathcal{O}$.
Let $\mathsf{ct} = (c_1, \boldsymbol{c}_2)$ be a normal ciphertext, and $z_3, z_4 \leftarrow \mathbb{Z}_p$ be random elements. We define a semi-functional ciphertext as $\mathsf{ct}' = (c_1', \boldsymbol{c}_2')$ where $c_1' = c_1$ and $\boldsymbol{c}_2' = \boldsymbol{c}_2 \cdot g_1^{z_3\cdot\boldsymbol{d}_3 + z_4\cdot\boldsymbol{d}_4}$.

We are going to prove Theorem 4.3.2 with a sequence of $Q+3$ hybrids games, where $Q \in \mathbb{N}$ is the number of secret keys that an adversary can query.

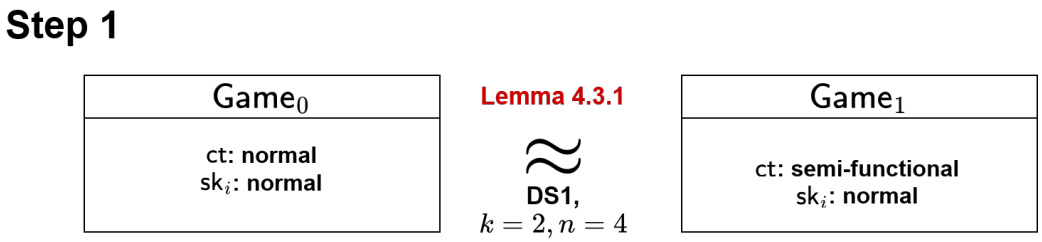
- Game_0: is the real IND-WIBE security game, presented in Figure 4.1.
- Game_1: is as Game_0 except that the challenge ciphertext is semi-functional.
- Game_{2-j}: for j from 1 to Q, Game_{2-j} is the same as Game_1 except that the first j keys are semi-functional and the remaining keys are normal.
- Game_3: is the same as Game_{2-Q}, except that the challenge ciphertext is a semi-functional encryption of a random message in $\mathbb{G}_T$.

We prove indistinguishability between those security games by proving the three following lemmas. The proofs are using assumptions DS1 and DS2, presented in Section 2.3.2. Informally, we prove indistinguishability between the $Q+3$ games above as explained in

Figure 4.10.

- **Step 1**: we prove that if an adversary can distinguish Game_0 from Game_1 then we can build an adversary with non-negligible advantage against DS1 with $k = 2$ and $n = 4$. This is formalized by Lemma 4.3.1.
- **Step 2**: we show that if an adversary can distinguish $\text{Game}_{2-(j-1)}$ from Game_{2-j} we can build an adversary with non-negligible advantage against DS2 with $k = 2$ and $n = 4$. This is resumed in Lemma 4.3.2.
- **Step 3**: we prove that if an adversary can distinguish Game_{2-Q} from Game_3 then we can build an adversary with non-negligible advantage against DS1 with $k = 1$ and $n = 4$. This is traduced by Lemma 4.3.3. We actually prove this in two steps, by randomizing each appearance of s in the $\boldsymbol{c}_2$ term of the ciphertext, thereby severing its link with the blinding factor. The end result is a semi-functional encryption of a random message. As a first step, we consider an intermediary game, called $\text{Game}_{2-Q'}$, that is exactly like Game_{2-Q}, except that in the $\boldsymbol{c}_2$ term of the challenge ciphertext the coefficient of $\boldsymbol{d}_2$ is changed from being $s \sum_{i \in W(P)} a_i$ to a fresh random value in $\mathbb{Z}_p$. Then we prove that
  - **Step 3.1**: if an adversary can distinguish Game_{2-Q} from $\text{Game}_{2-Q'}$ then we can build an adversary with non-negligible advantage against DS1 with $k = 1$ and $n = 4$. This formalized by Lemma 4.3.4.
  - **Step 3.2**: if an adversary can distinguish $\text{Game}_{2-Q'}$ from Game_3 then we can build an adversary with non-negligible advantage against DS2 with $k = 1$ and $n = 4$, as stated in Lemma 4.3.5.

Figure 4.10: Informal security proof for our constant size ciphertext identity-based encryption with wildcards scheme.



Lemma 4.3.1 *If there exists a PPT algorithm $\mathcal{A}$ such that $\text{Adv}^0_{\mathcal{A}} - \text{Adv}^1_{\mathcal{A}}$ is non-negligible, then there exists a PPT algorithm $\mathcal{B}$ with non-negligible advantage against DS1 with $k = 2$ and $n = 4$.*

Proof 4.3.2 *INIT: $\mathcal{B}$ is given $\Delta = (\Gamma, g_2^{\boldsymbol{b}_1^*}, g_2^{\boldsymbol{b}_2^*}, g_1^{\boldsymbol{b}_1}, g_1^{\boldsymbol{b}_2}, g_1^{\boldsymbol{b}_3}, g_1^{\boldsymbol{b}_4}, \boldsymbol{u}_1, \boldsymbol{u}_2, \mu_2)$ along with $\boldsymbol{t}_1, \boldsymbol{t}_2$, distributed either as $g_1^{\tau_1 \boldsymbol{b}_1}$, $g_1^{\tau_1 \boldsymbol{b}_2}$ or $g_1^{\tau_1 \boldsymbol{b}_1 + \tau_2 \boldsymbol{b}_3}$, $g_1^{\tau_1 \boldsymbol{b}_2 + \tau_2 \boldsymbol{b}_4}$.*

SETUP: $\mathcal{B}$ first chooses a random invertible matrix $\boldsymbol{A} \in \mathbb{Z}_p^{2\times 2}$. She implicitly sets dual orthonormal bases $\mathbb{D}, \mathbb{D}^*$ to: $\boldsymbol{d}_1 = \boldsymbol{b}_1$,$\boldsymbol{d}_2 = \boldsymbol{b}_2$, $(\boldsymbol{d}_3, \boldsymbol{d}_4) = (\boldsymbol{b}_3, \boldsymbol{b}_4) \cdot \boldsymbol{A}$, $\boldsymbol{d}_1^* = \boldsymbol{b}_1^*$, $\boldsymbol{d}_2^* = \boldsymbol{b}_2^*$, $(\boldsymbol{d}_3^*, \boldsymbol{d}_4^*) = (\boldsymbol{b}_3^*, \boldsymbol{b}_4^*) \cdot (\boldsymbol{A}^{-1})^\top$. We note that $\mathbb{D}, \mathbb{D}^*$ are properly distributed and reveal no information about $\boldsymbol{A}$. Notice also that $\mathcal{B}$ cannot produce $g_2^{\boldsymbol{d}_3^*}, g_2^{\boldsymbol{d}_4^*}$, but these will not be needed to create normal keys. $\mathcal{B}$ chooses random values $\alpha, a_1, \cdots, a_L \in \mathbb{Z}_p$. $\mathcal{A}$ is given the public key

$$\mathit{pk} = (\Gamma, e(g_1, g_2)^{\alpha \boldsymbol{d}_1 \cdot \boldsymbol{d}_1^*}, g_1^{\boldsymbol{d}_1}, \boldsymbol{h}_1 = g_1^{a_1 \boldsymbol{d}_2}, \cdots, \boldsymbol{h}_L = g_1^{a_L \boldsymbol{d}_2}).$$

The master secret key is $\mathit{msk} = (\alpha, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_2^*}, a_1, \cdots, a_L)$.

KEY QUERY: msk is known to $\mathcal{B}$, which allows $\mathcal{B}$ to respond to all of $\mathcal{A}$'s key queries by calling the normal key generation algorithm.

CHALLENGE: $\mathcal{A}$ sends $\mathcal{B}$ a challenge pattern $\boldsymbol{P}$ and two messages (m_0, m_1). $\mathcal{B}$ chooses a random bit $b \in \{0, 1\}$ and encrypts m_b under $\boldsymbol{P}$ as follows:

$$c_1 = m_b \cdot (e(\boldsymbol{t}_1, g_2^{\boldsymbol{b}_1^*}))^\alpha = m_b \cdot (e(\boldsymbol{t}_1, g_2^{\boldsymbol{d}_1^*}))^\alpha \quad , \quad \boldsymbol{c}_2 = \boldsymbol{t}_1 \cdot \boldsymbol{t}_2^{\sum\limits_{i \in W(\boldsymbol{P})} a_i}.$$

She gives the ciphertext $\mathit{ct}^* = (c_1, \boldsymbol{c}_2)$ to $\mathcal{A}$.

- *If $(\boldsymbol{t}_1, \boldsymbol{t}_2) = (g_1^{\tau_1 \boldsymbol{b}_1}, g_1^{\tau_1 \boldsymbol{b}_2})$, we have a normal ciphertext with randomness τ_1: $c_1 = (m_b (e(g_1, g_2)^{\boldsymbol{d}_1 \cdot \boldsymbol{d}_1^* \alpha})^{\tau_1}$, and $\boldsymbol{c}_2 = g_1^{\tau_1 \boldsymbol{d}_1 + \tau_1 \boldsymbol{d}_2 \sum\limits_{i \in W(\boldsymbol{P})} a_i}$. Thus $\mathcal{B}$ has properly simulated Game_0.*

- *If $(\boldsymbol{t}_1, \boldsymbol{t}_2) = (g_1^{\tau_1 \boldsymbol{b}_1 + \tau_2 \boldsymbol{b}_3}, g_1^{\tau_1 \boldsymbol{b}_2 + \tau_2 \boldsymbol{b}_4})$, $c_1 = m_b \cdot (e(g_1, g_2)^{\boldsymbol{b}_1 \cdot \boldsymbol{b}_1^* \alpha})^{\tau_1} \cdot e(g_1, g_2)^{\tau_2 \boldsymbol{b}_3 \boldsymbol{b}_1^* \alpha} = m_b \cdot (e(g_1, g_2)^{\boldsymbol{d}_1 \cdot \boldsymbol{d}_1^* \alpha})^{\tau_1}$ and $\boldsymbol{c}_2 = g_1^{\tau_1 \boldsymbol{d}_1 + \tau_1 \boldsymbol{d}_2 \sum\limits_{i \in W(\boldsymbol{P})} a_i + \tau_2 \boldsymbol{b}_3 + \tau_2 \boldsymbol{b}_4 \sum\limits_{i \in W(\boldsymbol{P})} a_i}$.*

This ciphertext has an additional term with coefficients in basis $\boldsymbol{b}_3, \boldsymbol{b}_4$, which form the vector $\tau_2(1, \sum\limits_{i \in W(\boldsymbol{P})} a_i)$. To compute coefficients in the basis $(\boldsymbol{d}_3, \boldsymbol{d}_4)$ we multiply the matrix $\boldsymbol{A}^{-1}$ by the transpose of this vector. Since $\boldsymbol{A}$ is random, these new coefficients are uniformly random. Thus in this case the ciphertext is SF (with coefficients in the base $\mathbb{D}$) and $\mathcal{B}$ has properly simulated Game_1. This allows $\mathcal{B}$ to leverage $\mathcal{A}$'s non-negligible difference in advantage between Game_0 and Game_1 to achieve a non-negligible advantage against DS1. □

Step 2

$j = 1, \cdots, Q$:

$\mathsf{Game}_{2-(j-1)}$
ct: **semi-functional** $\mathsf{sk}_1, \cdots, \mathsf{sk}_{j-1}$: **semi-functional** $\mathsf{sk}_j, \cdots, \mathsf{sk}_Q$: **normal**

Lemma 4.3.2
$\approx$
DS2, $k = 2, n = 4$

$\mathsf{Game}_{2-(j-1)}$
ct: **semi-functional** $\mathsf{sk}_1, \cdots, \mathsf{sk}_j$: **semi-functional** $\mathsf{sk}_{j+1}, \cdots, \mathsf{sk}_Q$: **normal**

Lemma 4.3.2 *If there exists a PPT algorithm $\mathcal{A}$ such that $\mathsf{Adv}_{\mathcal{A}}^{2-(j-1)} - \mathsf{Adv}_{\mathcal{A}}^{2-j}$ is non-negligible, then there exists a PPT algorithm $\mathcal{B}$ with non-negligible advantage against DS2 with $k = 2$ and $n = 4$.*

Proof 4.3.3 *INIT: $\mathcal{B}$ is given $\Delta = (\Gamma, g_1^{\boldsymbol{b}_1}, g_1^{\boldsymbol{b}_2}, g_2^{\boldsymbol{b}_1^*}, g_2^{\boldsymbol{b}_2^*}, g_2^{\boldsymbol{b}_3^*}, g_2^{\boldsymbol{b}_4^*}, \boldsymbol{u}_1, \boldsymbol{u}_2, \mu_2)$ along with $\boldsymbol{t}_1, \boldsymbol{t}_2$, distributed either as $g_2^{\tau_1 \boldsymbol{b}_1^*}, g_2^{\tau_1 \boldsymbol{b}_2^*}$ or $g_2^{\tau_1 \boldsymbol{b}_1^* + \tau_2 \boldsymbol{b}_3^*}, g_2^{\tau_1 \boldsymbol{b}_2^* + \tau_2 \boldsymbol{b}_4^*}$.*

SETUP: $\mathcal{B}$ chooses a random invertible matrix $\boldsymbol{A} \in \mathbb{Z}_q^{2\times 2}$. Then she implicitly sets dual orthonormal bases $\mathbb{D}, \mathbb{D}^*$ to: $\boldsymbol{d}_1 = \boldsymbol{b}_1$, $\boldsymbol{d}_2 = \boldsymbol{b}_2$, $(\boldsymbol{d}_3, \boldsymbol{d}_4) = (\boldsymbol{b}_3, \boldsymbol{b}_4) \cdot \boldsymbol{A}$, $\boldsymbol{d}_1^* = \boldsymbol{b}_1^*$, $\boldsymbol{d}_2^* = \boldsymbol{b}_2^*$,$(\boldsymbol{d}_3^*, \boldsymbol{d}_4^*) = (\boldsymbol{b}_3^*, \boldsymbol{b}_4^*) \cdot (\boldsymbol{A}^{-1})^\top$. We note that $\mathbb{D}, \mathbb{D}^*$ are properly distributed and reveal no information about $\boldsymbol{A}$. $\mathcal{B}$ chooses random values $\alpha, a_1, \cdots, a_L \in \mathbb{Z}_p$. $\mathcal{A}$ is given the public key

$$\mathsf{pk} = (\Gamma, e(g_1, g_2)^{\alpha \boldsymbol{d}_1 . \boldsymbol{d}_1^*}, g_1^{\boldsymbol{d}_1}, \boldsymbol{h}_1 = g_1^{a_1 \boldsymbol{d}_2}, \cdots, \boldsymbol{h}_L = g_1^{a_L \boldsymbol{d}_2}).$$

The master secret key is $\mathsf{msk} = (\alpha, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_2^*}, a_1, \cdots, a_L)$.

KEY QUERY: $\mathcal{B}$ knows msk and $g_2^{\boldsymbol{d}_3^*}, g_2^{\boldsymbol{d}_4^*}$, thus can easily call the key generation algorithm or produce semi-functional keys. It allows $\mathcal{B}$ to answer to all $\mathcal{A}$'s key queries.

- *To answer the first j-1 key queries that $\mathcal{A}$ makes, $\mathcal{B}$ runs the semi-functional key generation algorithm to produce semi-functional keys.*
- *To answer to the j-th key query for $\boldsymbol{P}^j$, $\mathcal{B}$ responds with:*

$$\boldsymbol{\beta} = (g_2^{\boldsymbol{b}_1^*})^{\alpha} \cdot \boldsymbol{t}_1^{\sum_{i \in \mathcal{I}} a_i} \cdot \boldsymbol{t}_2^{-1}, \qquad \boldsymbol{v}_i = \boldsymbol{t}_1^{a_i} \quad \text{for } i \in \mathcal{O}.$$

  - *If $\boldsymbol{t}_1, \boldsymbol{t}_2 = g_2^{\tau_1 \boldsymbol{b}_1^*}, g_2^{\tau_1 \boldsymbol{b}_2^*}$, then $\mathsf{sk}_{\boldsymbol{P}^j}$ is a normal key with randomness τ_1: $\boldsymbol{\beta} = g_2^{\boldsymbol{d}_1 \alpha + \tau_1 \boldsymbol{d}_1^* \sum_{i \in \mathcal{I}} a_i - \tau_1 \boldsymbol{d}_2^*}$ and $\boldsymbol{v}_i = g_2^{\tau_1 \boldsymbol{d}_1^* a_i}$, for $i \in \mathcal{O}$.*
  - *If $\boldsymbol{t}_1, \boldsymbol{t}_2 = g_2^{\tau_1 \boldsymbol{b}_1^* + \tau_2 \boldsymbol{b}_3^*}, g_2^{\tau_1 \boldsymbol{b}_2^* + \tau_2 \boldsymbol{b}_4^*}$, $\boldsymbol{\beta} = g_2^{\boldsymbol{d}_1 \alpha + \tau_1 \boldsymbol{d}_1^* \sum_{i \in \mathcal{I}} a_i - \tau_1 \boldsymbol{d}_2^* + \tau_2 \sum_{i \in \mathcal{I}} a_i \boldsymbol{b}_3^* - \tau_2 \boldsymbol{b}_4^*}$ and $\boldsymbol{v}_i = g_2^{\tau_1 \boldsymbol{d}_1^* a_i + \tau_2 \boldsymbol{b}_3^*}$, for $i \in \mathcal{O}$.*

- *For the remaining key queries, $\mathcal{B}$ runs the normal key generation algorithm.*

CHALLENGE: At some point, $\mathcal{A}$ sends to $\mathcal{B}$ two messages m_0, m_1 and a challenge pattern $\boldsymbol{P}$. $\mathcal{B}$ chooses a random bit $b \in \{0,1\}$ and encrypts m_b under $\boldsymbol{P}$ as follows:

$$c_1 = m_b \cdot (e(\boldsymbol{u}_1, g_2^{\boldsymbol{b}_1^*}))^{\alpha} = m_b \cdot (e(g_1, g_2))^{\alpha \boldsymbol{d}_1 \cdot \boldsymbol{d}_1^* \mu_1}$$
$$\text{and } \boldsymbol{c}_2 = \boldsymbol{u}_1 \cdot \boldsymbol{u}_2^{\sum_{i \in W(\boldsymbol{P})} a_i} = g_1^{\mu_1 \boldsymbol{d}_1 + \mu_1 \boldsymbol{d}_2 \sum_{i \in W(\boldsymbol{P})} a_i + \mu_2 \boldsymbol{b}_3 + \mu_2 \boldsymbol{b}_4 \sum_{i \in W(\boldsymbol{P})} a_i}.$$

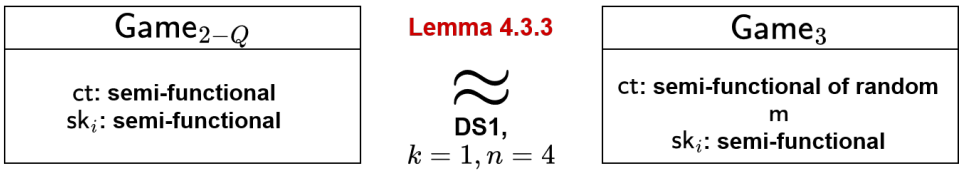
Suppose that $\mathcal{B}$ decides not to be honest, and find the nature of the j-th key by itself. To do so, she creates a ciphertext for a pattern $\boldsymbol{P}^*$ such that $\boldsymbol{P}^j \in_\star \boldsymbol{P}^*$. She tries to decrypt it with $sk_{\boldsymbol{P}^j}$ to learn if $sk_{\boldsymbol{P}^j}$ is a normal or a semi-functional key (a normal key will decrypt correctly while a SF key will with high probability fail to decrypt). Let us see that by construction even if $sk_{\boldsymbol{P}^j}$ is SF it will decrypt correctly.
Suppose that $\boldsymbol{t}_1, \boldsymbol{t}_2 = (g_2^{\tau_1 \boldsymbol{b}_1^* + \tau_2 \boldsymbol{b}_3^*}, g_2^{\tau_1 \boldsymbol{b}_2^* + \tau_2 \boldsymbol{b}_4^*})$. During decryption, $\mathcal{B}$ obtains the term

$$e\left(g_1^{\mu_2 \boldsymbol{b}_3 + \mu_2 \sum_{i \in W(\boldsymbol{P}^*)} a_i \boldsymbol{b}_4}, g_2^{\tau_2 \boldsymbol{b}_3^* \sum_{i \in \mathcal{I}} a_i - \tau_2 \boldsymbol{b}_4^*} \cdot g_2^{\tau_2 \boldsymbol{b}_3^* \sum_{i \in W(\boldsymbol{P}^*) \cap \mathcal{O}} a_i}\right).$$

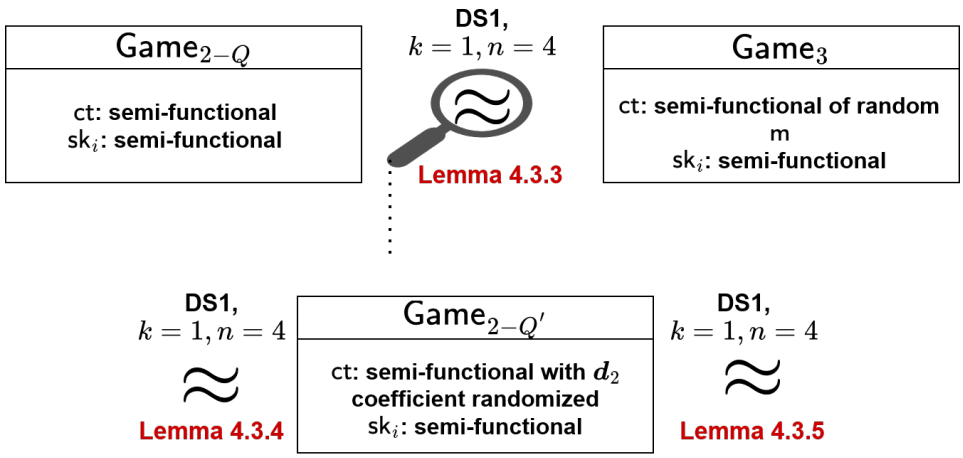
In the exponent we have $\mu_2(\boldsymbol{b}_3 + \boldsymbol{b}_4 \sum_{i \in W(\boldsymbol{P}^*)} a_i \boldsymbol{b}_4) \cdot \tau_2(\boldsymbol{b}_3^* \sum_{i \in W(\boldsymbol{P}^*)} a_i - \boldsymbol{b}_4^*)$ because $\boldsymbol{P}^j \in_\star \boldsymbol{P}^*$ implies $\mathcal{I} \cap (W(\boldsymbol{P}^*) \cup \mathcal{O}) = W(\boldsymbol{P}^*)$. The term in the exponent is: $\mu_2 \tau_2 \psi \sum_{i \in W(\boldsymbol{P}^*)} a_i - \mu_2 \tau_2 \psi \sum_{i \in W(\boldsymbol{P}^*)} a_i = 0$. Thus it will decrypt, and $\mathcal{B}$ will have no information about the j-th key 's nature.

In the authorized case, $\boldsymbol{P}^j \notin_\star \boldsymbol{P}$. Let's see that the extra coefficients in basis $(\boldsymbol{b}_3, \boldsymbol{b}_4)$ of the ciphertext and the extra coefficients in basis $(\boldsymbol{b}_3^*, \boldsymbol{b}_4^*)$ of the key are distributed as random vectors in the spans of $(\boldsymbol{d}_3, \boldsymbol{d}_4)$ and $(\boldsymbol{d}_3^*, \boldsymbol{d}_4^*)$ respectively. To express them in basis $(\boldsymbol{d}_3, \boldsymbol{d}_4)$ and $(\boldsymbol{d}_3^*, \boldsymbol{d}_4^*)$ respectively, we multiply them by $\boldsymbol{A}^{-1}$ and $\boldsymbol{A}^\top$ respectively. Since the distribution of everything given to $\mathcal{A}$ except for the $j-$th key and the challenge ciphertext is independent of the random matrix $\boldsymbol{A}$ and $\boldsymbol{P}^j \notin_\star \boldsymbol{P}$, we can conclude that these coefficients are uniformly random. Thus $\mathcal{B}$ has properly simulated $Game_{2-j}$ in this case.
If $\boldsymbol{t}_1, \boldsymbol{t}_2 = g_2^{\tau_1 \boldsymbol{b}_1^*}, g_2^{\tau_1 \boldsymbol{b}_2^*}$ then the coefficients of the semi functional part of the ciphertext are uniformly random. Thus $\mathcal{B}$ has properly simulated $Game_{2-(j-1)}$ in this case. Therefore $\mathcal{B}$ can leverage $\mathcal{A}$'s non-negligible difference in advantage between these games to obtain a non-negligible advantage against DS2. □

Step 3

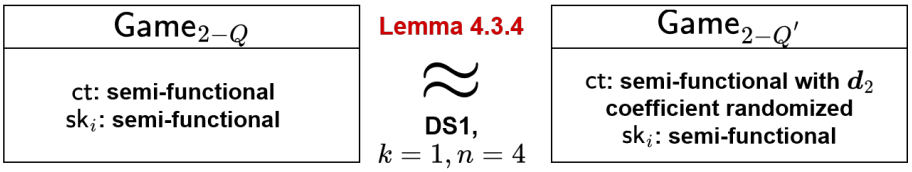


Lemma 4.3.3 *If there exists a PPT algorithm $\mathcal{A}$ such that* $\mathsf{Adv}^{2-Q}_{\mathcal{A}} - \mathsf{Adv}^{3}_{\mathcal{A}}$ *is non-negligible, then there exists a PPT algorithm $\mathcal{B}$ with non-negligible advantage against DS1 with $k = 1$ and $n = 4$.*



We prove this lemma in two steps, by randomizing each appearance of s in the c_2 term of the ciphertext, thereby severing its link with the blinding factor. The end result is a semi-functional encryption of a random message. As a first step, we consider an intermediary game, called $\mathsf{Game}_{2-Q'}$, that is exactly like Game_{2-Q}, except that in the c_2 term of the challenge ciphertext the coefficient of d_2 is changed from being $s \sum_{i \in W(\boldsymbol{P})} a_i$ to a fresh random value in $\mathbb{Z}_p$. We denote the advantage of an algorithm $\mathcal{A}$ in this game by $\mathsf{Adv}^{Q'}_{\mathcal{A}}$. We first prove the following lemma.

Step 3.1



Lemma 4.3.4 *If there exists a PPT algorithm $\mathcal{A}$ such that $\mathsf{Adv}_{\mathcal{A}}^{2-Q} - \mathsf{Adv}_{\mathcal{A}}^{2-Q'}$ is non-negligible, then there exists a PPT algorithm $\mathcal{B}$ with non-negligible advantage against DS1 with $k = 1$ and $n = 4$.*

Proof 4.3.4 *INIT: $\mathcal{B}$ is given $\Delta = (\Gamma, g_2^{\boldsymbol{b}_1^*}, g_2^{\boldsymbol{b}_3^*}, g_2^{\boldsymbol{b}_4^*}, g_1^{\boldsymbol{b}_1}, g_1^{\boldsymbol{b}_2}, g_1^{\boldsymbol{b}_3}, g_1^{\boldsymbol{b}_4}, \boldsymbol{u}_1, \mu_2)$, along with $\boldsymbol{t}_1$ either equal to $g_1^{\tau_1 \boldsymbol{b}_1}$ or $g_1^{\tau_1 \boldsymbol{b}_1 + \tau_2 \boldsymbol{b}_2}$.*

SETUP: $\mathcal{B}$ implicitly sets $\boldsymbol{d}_1 = \boldsymbol{b}_3, \boldsymbol{d}_2 = \boldsymbol{b}_2, \boldsymbol{d}_3 = \boldsymbol{b}_1, \boldsymbol{d}_4 = \boldsymbol{b}_4$, and $\boldsymbol{d}_1^* = \boldsymbol{b}_3^*, \boldsymbol{d}_2^* = \boldsymbol{b}_2^*, \boldsymbol{d}_3^* = \boldsymbol{b}_1^*, \boldsymbol{d}_4^* = \boldsymbol{b}_4^*$.

This enables $\mathcal{B}$ to produce $g_1^{\boldsymbol{d}_1}, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_3}, g_1^{\boldsymbol{d}_4}$. We note also that $\mathbb{D}, \mathbb{D}^*$ are properly distributed dual orthonormal bases, and that $\mathcal{B}$ can produce $g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_3^*}$ and $g_2^{\boldsymbol{d}_4^*}$ but does not know $g_2^{\boldsymbol{d}_2^*}$. $\mathcal{B}$ chooses random values $\alpha, a_1, \cdots, a_L \in \mathbb{Z}_p$. She gives $\mathcal{A}$ the public key

$$\mathit{pk} = (\Gamma, e(g_1, g_2)^{\alpha \boldsymbol{d}_1 \cdot \boldsymbol{d}_1^*}, g_1^{\boldsymbol{d}_1}, \boldsymbol{h}_1 = \boldsymbol{g}_1^{a_1 \boldsymbol{d}_2}, \cdots, \boldsymbol{h}_L = g_1^{a_L \boldsymbol{d}_2}).$$

KEY QUERY: we note that $\mathcal{B}$ does not know the full master secret key, but she knows $\boldsymbol{u}_1 = g_2^{\mu_1 \boldsymbol{b}_1^* + \mu_2 \boldsymbol{b}_2^*}$, μ_2 and $a_1, \cdots, a_L$. This allows her to produce semi-functional keys as follows: when $\mathcal{A}$ requests a key for some pattern $\boldsymbol{P}'$, $\mathcal{B}$ chooses random values $r', t_4 \in \mathbb{Z}_p$. She sets $r = \mu_2 r'$ and forms the secret key as: $\boldsymbol{\beta} = (\boldsymbol{u}_1)^{-r'} \cdot g_2^{\alpha \boldsymbol{d}_1^* + \mu_2 r' \sum_{i \in \mathcal{I}} a_i \boldsymbol{d}_1^* + t_4 \boldsymbol{d}_4^*}$, $\boldsymbol{v}_i = g_2^{\mu_2 r' a_i \boldsymbol{d}_1^* + t_{b,i} \boldsymbol{d}_3^*}$.
We obtain that $\boldsymbol{\beta} = g_2^{\alpha \boldsymbol{d}_1^* + r \boldsymbol{d}_1^* \sum_{i \in \mathcal{I}} a_i - r \boldsymbol{d}_2^* + (-r' \mu_1) \boldsymbol{d}_3^* + t_4 \boldsymbol{d}_4^*}$. The coefficients of $\boldsymbol{d}_3^*, \boldsymbol{d}_4^*$ are uniformly random thus it is a SF key.

CHALLENGE: $\mathcal{A}$ submits two messages m_0, m_1 and a challenge pattern $\boldsymbol{P}$. $\mathcal{B}$ chooses $b \in \{0, 1\}$ and forms the challenge ciphertext as follows:

$$c_1 = m_b \cdot (e(g_1, g_2)^{\alpha \boldsymbol{d}_1 \cdot \boldsymbol{d}_1^*})^s, \qquad \boldsymbol{c}_2 = g_1^{s \boldsymbol{d}_1 + s \boldsymbol{d}_2 \sum_{i \in W(\boldsymbol{P})} a_i} \cdot \boldsymbol{t}_1 \cdot g_1^{z \boldsymbol{d}_4}$$

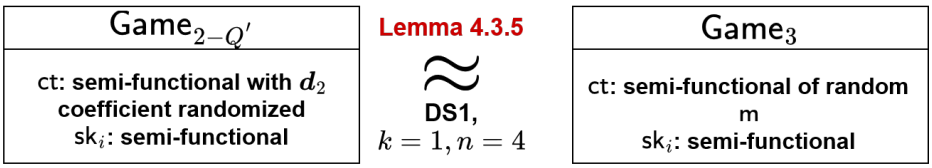
where $s, z \leftarrow \mathbb{Z}_p$.

- *If $\boldsymbol{t}_1$ is equal to $g_1^{\tau_1 \boldsymbol{b}_1}$ then $\boldsymbol{c}_2 = g_1^{s \boldsymbol{d}_1 + s \boldsymbol{d}_2 \sum_{i \in W(\boldsymbol{P})} a_i + \tau_1 \boldsymbol{d}_3 + z \boldsymbol{d}_4}$ which is a semi functional ciphertext and $\mathcal{B}$ simulates Game$_{2-Q}$.*
- *If $\boldsymbol{t}_1 = g_1^{\tau_1 \boldsymbol{b}_1 + \tau_2 \boldsymbol{b}_2}$ then $\boldsymbol{c}_2 = g_1^{s \boldsymbol{d}_1 + (s \sum_{i \in W(\boldsymbol{P})} a_i + \tau_2) \boldsymbol{d}_2 + \tau_1 \boldsymbol{d}_3 + z \boldsymbol{d}_4}$ is a semi functional ciphertext with randomized coefficients for $\boldsymbol{d}_2$, thus $\mathcal{B}$ simulates Game$_{2-Q'}$.*

Therefore, $\mathcal{B}$ can leverage $\mathcal{A}$'s non-negligible difference of advantage between these two games to achieve a non-negligible advantage against DS1. □

Note 4.3.1 *All queried keys shared μ_1, μ_2 in their randomness. However, as it is in exponent and "randomized" by other random elements, then for an adversary it is indistinguishable from a truly random element.*

Step 3.2



Lemma 4.3.5 *If there exists a PPT algorithm $\mathcal{A}$ such that $\mathsf{Adv}_{\mathcal{A}}^{2-Q'} - \mathsf{Adv}_{\mathcal{A}}^{3}$ is non-negligible, then there exists a PPT algorithm $\mathcal{B}$ with non-negligible advantage against DS1 with $k = 1$ and $n = 4$.*

Proof 4.3.5 *INIT: $\mathcal{B}$ is given $\Delta = (\Gamma, g_2^{\boldsymbol{b}_1^*}, g_2^{\boldsymbol{b}_3^*}, g_2^{\boldsymbol{b}_4^*}, g_1^{\boldsymbol{b}_1}, g_1^{\boldsymbol{b}_2}, g_1^{\boldsymbol{b}_3}, g_1^{\boldsymbol{b}_4}, \boldsymbol{u}_1, \mu_2)$, along with $\boldsymbol{t}_1$ either equal to $g_1^{\tau_1 \boldsymbol{b}_1}$ or $g_1^{\tau_1 \boldsymbol{b}_1 + \tau_2 \boldsymbol{b}_2}$.*

SETUP: $\mathcal{B}$ implicitly sets $\boldsymbol{d}_1 = \boldsymbol{b}_2, \boldsymbol{d}_2 = \boldsymbol{b}_3, \boldsymbol{d}_3 = \boldsymbol{b}_1, \boldsymbol{d}_4 = \boldsymbol{b}_4$, and $\boldsymbol{d}_1^* = \boldsymbol{b}_2^*, \boldsymbol{d}_2^* = \boldsymbol{b}_3^*, \boldsymbol{d}_3^* = \boldsymbol{b}_1^*, \boldsymbol{d}_4^* = \boldsymbol{b}_4^*$.

This enables $\mathcal{B}$ to produce $g_1^{\boldsymbol{d}_1}, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_3}, g_1^{\boldsymbol{d}_4}$, but not $\boldsymbol{d}_2$. We note also that $\mathbb{D}, \mathbb{D}^*$ are properly distributed dual orthonormal bases, and that $\mathcal{B}$ can produce $g_2^{\boldsymbol{d}_2^*}, g_2^{\boldsymbol{d}_3^*}$ and $g_2^{\boldsymbol{d}_4^*}$ but does not know $g_2^{\boldsymbol{d}_1^*}$. $\mathcal{B}$ chooses random values $\alpha', a_1, \cdots, a_L \in \mathbb{Z}_p$. She computes $e(g_1^{\boldsymbol{b}_3}, g_2^{\boldsymbol{b}_3^*})^{\alpha} = e(g_1, g_2)^{\alpha \boldsymbol{d}_2 \cdot \boldsymbol{d}_2^*} = e(g_1, g_2)^{\alpha \psi} = e(g_1, g_2)^{\alpha \boldsymbol{d}_1 \cdot \boldsymbol{d}_1^*}$. She gives $\mathcal{A}$ the public key

$$pk = (\Gamma, e(g_1, g_2)^{\alpha \boldsymbol{d}_1 \cdot \boldsymbol{d}_1^*}, g_1^{\boldsymbol{d}_1}, \boldsymbol{h}_1 = g_1^{a_1 \boldsymbol{d}_2}, \cdots, \boldsymbol{h}_L = g_1^{a_L \boldsymbol{d}_2}).$$

KEY QUERY: We note that $\mathcal{B}$ does not know the full master secret key, but she knows $\boldsymbol{u}_1 = g_2^{\mu_1 \boldsymbol{b}_1^* + \mu_2 \boldsymbol{b}_2^*}$, μ_2 and $a_1, \cdots, a_L$. This allows her to produce SF keys as follows: when $\mathcal{A}$ requests a key for some pattern $\boldsymbol{P}'$, $\mathcal{B}$ chooses random values $r', t_4 \in \mathbb{Z}_p$. She sets $r = \mu_2 r'$ and forms the secret key as: $\boldsymbol{\beta} = (\boldsymbol{u}_1)^{(\alpha' + r' \sum_{i \in \mathcal{I}} a_i)} \cdot g_2^{-\mu_2 r' \boldsymbol{d}_2^* + t_4 \boldsymbol{d}_4^*}$, $\boldsymbol{v}_i = \boldsymbol{u}_1^{r' a_i}$.

We obtain that $\boldsymbol{\beta} = g_2^{\alpha \boldsymbol{d}_1^* + r\boldsymbol{d}_1^* \sum_{i\in\mathcal{I}} a_i - r\boldsymbol{d}_2^* + (\alpha' \mu_1 + r' \mu_1 \sum_{i\in\mathcal{I}} a_i)\boldsymbol{d}_3^* + t_4 \boldsymbol{d}_4^*}$ *and* $\boldsymbol{v}_i = g_2^{r\boldsymbol{d}_1^* a_i + r' \mu_1 a_i \boldsymbol{d}_3^*}$*. The coefficients of* $\boldsymbol{d}_3^*, \boldsymbol{d}_4^*$ *are uniformly random thus it is a SK key.*

CHALLENGE: $\mathcal{A}$ *submits messages* m_0, m_1 *and challenge pattern* $\boldsymbol{P}$*,* $\mathcal{B}$ *chooses* $b \in \{0,1\}$ *and forms the challenge ciphertext as follows:* $s, w, z \leftarrow \mathbb{Z}_p$*,*

$$c_1 = m_b \cdot (e(g_1, g_2)^{\alpha \boldsymbol{d}_1 \cdot \boldsymbol{d}_1^*})^s, \qquad \boldsymbol{c}_2 = g_1^{s\boldsymbol{d}_1 + w\boldsymbol{d}_2} \cdot \boldsymbol{t}_1 \cdot g_1^{z\boldsymbol{d}_4}$$

- *If* $\boldsymbol{t}_1$ *is equal to* $g_1^{\tau_1 \boldsymbol{b}_1}$ *then* $\boldsymbol{c}_2 = g_1^{s\boldsymbol{d}_1 + w\boldsymbol{d}_2 + \tau_1 \boldsymbol{d}_3 + z\boldsymbol{d}_4}$ *is a semi functional ciphertext with the second appearance of* s *randomized. In this case* $\mathcal{B}$ *simulates* $\text{Game}_{2-Q'}$*.*
- *If* $\boldsymbol{t}_1$ *is equal to* $g_1^{\tau_1 \boldsymbol{b}_1 + \tau_2 \boldsymbol{b}_2}$ *then* $\boldsymbol{c}_2 = g_1^{(s+\tau_2)\boldsymbol{d}_1 + w\boldsymbol{d}_2 + \tau_1 \boldsymbol{d}_3 + z\boldsymbol{d}_4}$ *which is a semi functional ciphertext with randomized coefficients for* $\boldsymbol{d}_1$ *and* $\boldsymbol{d}_2$*. Thus in this case* $\mathcal{B}$ *simulates* Game_3*.*

Therefore, $\mathcal{B}$ *can leverage* $\mathcal{A}$*'s non-negligible difference of advantage between these two games to achieve a non-negligible advantage against DS1.* □

Combining lemmas 4.3.4 and 4.3.5 we obtain lemma 6.2.3. Along with lemmas 2.3.2, 4.3.1 and 4.3.2, this completes the proof of theorem 4.3.2.

Note 4.3.2 *Our first WIBE does not satisfy* pattern-hiding *security. We can easily see that as in order to decrypt, one must know which parts of her secret key she has to take according to the ciphertext pattern. That means that this pattern must be given to make decryption working, and thus the scheme is not pattern-hiding.*

4.3.2 Our Pattern-Hiding WIBE Scheme

Intuition. Additionally we tried to add *pattern-hiding* security to our first WIBE scheme. We based on work on the works of Lewko *et al.* [91] and Okamoto *et al.* [115, 114]: they propose Inner Product Encryption (IPE) schemes with different levels of security. We easily see a similarity between strong (resp. weak) attribute-hiding security of IPE, Definition 4.2.3, and pattern-hiding, Definition 4.2.3 (resp. anonymity, Definition 4.1.5) security of WIBE. It is that similarity that we are going to exploit.

First let us briefly present the ideas of the above quoted works. Let $L \in \mathbb{N}$ be the vectors length. In these works each position of the vector is associated to a different vector of the DPVS, thus they need a DPVS with dimension L. Actually they use DPVS with

dimension $L+1$ to have one vector that "carries" the scheme secret key α (as we did in our first scheme).
To bring security (indistinguishability and weak attribute-hiding) [91] increases the DPVS dimension to $2L+3$ to add 1-dimensional randomness space to the secret keys, 1-dimensional randomness space to the ciphertexts and L-dimensional hidden subspace to realize the semi-functional forms of the secret keys and ciphertexts in the security proof. [115] improved [91]'s scheme security: in the latter, the security assumption is non-standard and non-static, while in the former the security assumption is the DLin assumption, which is a standard, static assumption that brings higher security. This is done at the cost of less efficiency as they increase by $n-1$ the dimension of the secret keys randomness space, thus the DPVS dimension is now equal to $3L+2$.
Finally, to bring fully attribute-hiding security, [114] increased the hidden subspace from L to $2L$, which results in a scheme with DPVS dimension equals to $4L+2$. This enlargement in needed as Okamato *et al.* extended the Dual System Encryption framework: secret keys can now have three forms (normal, *temporal 1* and *temporal 2*) while ciphertexts can have five forms (normal, *temporal 0*, *temporal 1*, *temporal 2* and *unbiased*).

Based on that, we transform [114]'s IPE scheme in a WIBE scheme with keys patterns space equals to $\{0,1\}^L \backslash \{\mathbf{0}^L\}$ and ciphertexts patterns space equals to $\{0,\star\}^L \backslash \{\star^L\}$. Our idea to deal with wildcards is simply to only keep the pattern positions equals to 1 in the secret key and the pattern positions equals to 0 in the ciphertext. This gives us our second WIBE scheme, presented in Figure 4.11.

It is important here to notice that the decryption condition in our second WIBE is expressed differently from the one of our first WIBE. Indeed, for patterns $\boldsymbol{P}, \boldsymbol{P}^*$ in respectively $\{0,1\}^L \backslash \{\mathbf{0}^L\}$ and $\{0,\star\}^L \backslash \{\star^L\}$, we have that $\boldsymbol{P} \in_\star \boldsymbol{P}^* \implies \forall i \in [L]$, if $P_i^* \neq \star$ then $P_i = P_i^\star = 0$ and thus $\mathcal{I} \cap \bar{W}(\boldsymbol{P}^*) = \varnothing$.

Our pattern-hiding scheme. In Figure 4.11 we present our pattern-hiding WIBE scheme.

Theorem 4.3.3 *Our WIBE scheme is correct.*

Proof 4.3.6

$$e(\boldsymbol{c}_2, \boldsymbol{sk}_{\boldsymbol{P}'}) = e\left(g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1}} \cdot \prod_{i \in \bar{W}(\boldsymbol{P})} \boldsymbol{h}_i^{s_3}, g_2^{\alpha \boldsymbol{d}_0^* + \sum_{j \in \mathcal{I}} r_j \boldsymbol{d}_j^* + \sum_{l=1}^{L} \eta_l \cdot \boldsymbol{d}^* 3L+l} \right)$$

- Setup(λ, L): generate an asymmetric bilinear pairing group $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ for prime order p. Sample random dual orthonormal bases $(\mathbb{D}, \mathbb{D}^*) \leftarrow \text{Dual}(\mathbb{Z}_p^{4L+2})$. Let $\boldsymbol{d}_0, \cdots, \boldsymbol{d}_{4L+1}$ denote the elements of $\mathbb{D}$ and $\boldsymbol{d}_0^*, \cdots, \boldsymbol{d}_{4L+1}^*$ denote the elements of $\mathbb{D}^*$. Pick $\alpha \leftarrow \mathbb{Z}_p$. The public key is computed as: $\text{pk} = (\Gamma, e(g_1, g_2)^{\alpha \boldsymbol{d}_0 \cdot \boldsymbol{d}_0^*}, g_1^{\boldsymbol{d}_0}, g_1^{\boldsymbol{d}_{4L+1}}, \boldsymbol{h}_1 = g_1^{\boldsymbol{d}_1}, \cdots, \boldsymbol{h}_L = g_1^{\boldsymbol{d}_L})$ and the master secret key is $\text{msk} = (\alpha, g_2^{\boldsymbol{d}_0^*}, g_2^{\boldsymbol{d}_1^*}, \cdots g_2^{\boldsymbol{d}_L^*}, g_2^{\boldsymbol{d}_{3L+1}^*} \cdots, g_2^{\boldsymbol{d}_{4L}^*})$.
- KeyGen$(\text{msk}, \boldsymbol{P}')$: pick $\boldsymbol{r}, \boldsymbol{\eta} \in \mathbb{Z}_p^L$. The secret key is $\text{sk}_{\boldsymbol{P}'} = g_2^{\alpha \boldsymbol{d}_0^* + \sum_{j \in \mathcal{I}} r_j \boldsymbol{d}_j^* + \sum_{l=1}^{L} \eta_l \cdot \boldsymbol{d}_{3L+l}^*}$.
- Encrypt$(\text{pk}, \boldsymbol{P}, \text{m} \in \mathbb{G}_T)$: choose $s_1, s_2, s_3 \leftarrow \mathbb{Z}_p$ and compute $\text{ct} = (c_1, \boldsymbol{c}_2)$ where

$$c_1 = \text{m} \cdot (e(g_1, g_2)^{\alpha b_0^* \cdot b_0})^{s_1}, \qquad \boldsymbol{c}_2 = g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1}} \cdot \prod_{i \in \bar{W}(\boldsymbol{P})} \boldsymbol{h}_i^{s_3}.$$

- Decrypt$(\text{sk}_{\boldsymbol{P}'}, \text{ct})$: compute $c_1 \cdot \frac{1}{e(\boldsymbol{c}_2, \text{sk}_{\boldsymbol{P}'})}$.

Figure 4.11: Our identity-based encryption with wildcards scheme in prime order group, with adaptive indistinguishability and adaptive pattern-hiding security.

$$= e\left(g_1^{s_1 \boldsymbol{d}_0}, g_2^{\alpha \boldsymbol{d}_0^*}\right) \cdot e\left(g_1^{\sum_{i \in \bar{W}(\boldsymbol{P})} \boldsymbol{d}_i \cdot s_3}, g_2^{\sum_{j \in \mathcal{I}} r_j \boldsymbol{d}_j^*}\right)$$

The last row is obtained thanks to dual vector spaces properties. The first pairing cancels itself with the pairing in c_1. Now, let's see the value of $\sum_{i \in \bar{W}(\boldsymbol{P})} \boldsymbol{d}_i \cdot \sum_{j \in \mathcal{I}} \boldsymbol{d}_j^*$. As user with pattern $\boldsymbol{P}'$ is allowed to decrypt, $\mathcal{I} \cap \bar{W}(\boldsymbol{P}) = \varnothing$, and thanks to dual vector spaces properties, the above product is equal to 0 and decryptor obtains m. □

Security. The following theorem establishes the security of our new WIBE scheme.

Theorem 4.3.4 *If XDLin1, XDLin2 hold, then our scheme satisfies adaptive indistinguishability and is pattern-hiding, in the standard model.*

We are going to prove this theorem in two steps: first by proving the pattern-hiding security, then the adaptive indistinguishability.

Pattern-hiding security. Let us start with the proof that our scheme is pattern-hiding, as stated by the following lemma.

Lemma 4.3.6 *If XDLin*$1, XDLin2$ *hold, then our scheme satisfies pattern-hiding security.*

The proof is done as in [114] (Section 4.3.3), except that it is in the asymmetric setting. The different forms of ciphertext are defined according to challenge patterns $\boldsymbol{P}^0, \boldsymbol{P}^1$. c_1 is the same in all forms, just $\boldsymbol{c}_2$ is different. We use a sequence of $4Q+3$ games:

- Game$_0$: is the original pattern-hiding security game, presented in Figure 4.7.
- Game$_1$: is as Game$_0$ except that the ciphertext is changed to *temporal 0* form: let $b \in \{0,1\}, t \in \mathbb{Z}_p$ and suppose that $P_1^b = 0$. Define $\boldsymbol{c}_2$ as

$$g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}+s_3 \sum\limits_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i+t\boldsymbol{d}_{L+1}} \tag{4.1}$$

  This game is also called Game$_{2-0-4}$.
- For $1 \leq h \leq Q$ (the number of keys queried), we define the following 4 games:
  - Game$_{2-h-1}$: in this game, the challenge ciphertext is changed to *temporal 1* form: let $b \in \{0,1\}, t, u, \tilde{u} \in \mathbb{Z}_p$. Define $\boldsymbol{c}_2$ as

$$g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}+s_3 \sum\limits_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i+t \sum\limits_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_{L+i}+u \sum\limits_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i}+\tilde{u} \sum\limits_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{2L+i}} \tag{4.2}$$

    and the first $h-1$ keys are *temporal 2* forms: let $\boldsymbol{x} \in \mathbb{Z}_p^L$ be a random vector. Define the key as

$$g_2^{\alpha\boldsymbol{d}_0^*+\sum\limits_{j\in\mathcal{I}} r_j\boldsymbol{d}_j^*+\sum\limits_{j\in\mathcal{I}} x_j\boldsymbol{d}_{2L+j}^*+\sum\limits_{l=1}^{L} \eta_l\cdot\boldsymbol{d}_{3L+l}^*} \tag{4.3}$$

    while the remaining keys are normal.
  - Game$_{2-h-2}$: in this game the h-th key is changed to *temporal 1* form: let $\boldsymbol{z} \in \mathbb{Z}_p^L$ be a random vector. Define the key as

$$g_2^{\alpha\boldsymbol{d}_0^*+\sum\limits_{j\in\mathcal{I}} r_j\boldsymbol{d}_j^*+\sum\limits_{j\in\mathcal{I}} z_j\boldsymbol{d}_{L+j}^*+\sum\limits_{l=1}^{L} \eta_l\cdot\boldsymbol{d}_{3L+l}^*} \tag{4.4}$$

    while the remaining keys and the challenge ciphertext are the same as in Game$_{2-h-1}$.
  - Game$_{2-h-3}$: in this game, challenge ciphertext is changed to *temporal 2* form: let $b \in \{0,1\}, t, \tilde{t}, u, \tilde{u} \in \mathbb{Z}_p$. Define $\boldsymbol{c}_2$ as

$$g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}+s_3 \sum\limits_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i+t \sum\limits_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{L+i}+\tilde{t} \sum\limits_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{L+i}+u \sum\limits_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i}+\tilde{u} \sum\limits_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{2L+i}} \tag{4.5}$$

while all the queried keys are the same as in Game_{2-h-2}.

– Game_{2-h-4}: in this game, the h-th key is changed to *temporal 2* form (eq. 4.3) while the remaining keys and the challenge ciphertext are as in Game_{2-h-3}.

- Game_3: the challenge ciphertext is changed to *unbiased form*: let $b \in \{0,1\}, w, \tilde{w}, t, \tilde{t}, u, \tilde{u} \in \mathbb{Z}_p$. Define $\boldsymbol{c}_2$ as

$$g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}+w\sum\limits_{i\in\overline{W}(\boldsymbol{P}^0)}\boldsymbol{d}_i+\tilde{w}\sum\limits_{i\in\overline{W}(\boldsymbol{P}^1)}\boldsymbol{d}_i+t\sum\limits_{i\in\overline{W}(\boldsymbol{P}^0)}\boldsymbol{d}_{L+i}+\tilde{t}\sum\limits_{i\in\overline{W}(\boldsymbol{P}^1)}\boldsymbol{d}_{L+i}+u\sum\limits_{i\in\overline{W}(\boldsymbol{P}^0)}\boldsymbol{d}_{2L+i}+\tilde{u}\sum\limits_{i\in\overline{W}(\boldsymbol{P}^1)}\boldsymbol{d}_{2L+i}} \tag{4.6}$$

while all the queried keys are *temporal 2* form (eq. 4.3). In this game, the advantage of adversary is 0.

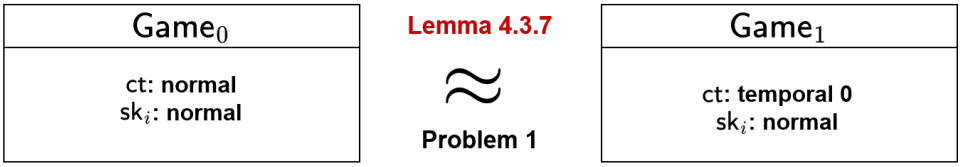
Informally, indistinguishability between those games is proven as in the original proof, using problems 1,2 and 3 (Definitions 2.3.5, 2.3.6 and 2.3.7). We present the informal security proof in Figure 4.12.

- **Step 1**: we prove that if there exists an adversary that can distinguish Game_0 from Game_1 then there exists an adversary that breaks Problem 1. This is formalized by Lemma 4.3.7.
- **Step 2**: we show that $\text{Game}_{2-(h-1)-4}$ can conceptually be changed into Game_{2-h-1}. The advantage of an adversary in distinguishing theses games is equal to $4/p$ when $h = 1$, otherwise it is equal to $3/p$. This is summarized by Lemma 4.3.8.
- **Step 3**: we prove that if there exists an adversary that can distinguish Game_{2-h-1} from Game_{2-h-2} then there exists an adversary that breaks Problem 2, as stated in Lemma 4.3.9.
- **Step 4**: we show that Game_{2-h-2} can conceptually be changed into Game_{2-h-3}. The advantage of an adversary in distinguishing theses games is equal to $\frac{4}{p^{|\mathcal{I}|}} + 5/p$. Here we had to modify [114]'s change of bases as it was not working with our scheme. This is summarized in Lemma 4.3.10.
- **Step 5**: we prove that if there exists an adversary that can distinguish Game_{2-h-3} from Game_{2-h-4} then there exists an adversary that breaks Problem 3 as formalized by Lemma 4.3.11.
- **Step 6**: we show that Game_{2-Q-4} can conceptually be changed into Game_3. The advantage of an adversary in distinguishing theses games is equal to $3/p$, as stated in Lemma 4.3.12.

Figure 4.12: Informal security proof for our pattern-hiding identity-based encryption with wildcards scheme.

We now start the formal proof.

Step 1



Lemma 4.3.7 *For any adversary $\mathcal{A}$, there exists a probabilistic adversary $\mathcal{B}$ against Problem 1, whose running time is essentially the same as that of $\mathcal{A}$, such that for any security parameter λ, $\left|\mathsf{Adv}^{(0)}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{(1)}_{\mathcal{A}}(\lambda)\right| \leq \mathsf{Adv}^{P1}_{\mathcal{B}}(\lambda)$.*

Proof 4.3.7 *To prove lemma 4.3.7, we construct a probabilistic machine $\mathcal{B}$ against Problem 1 using an adversary $\mathcal{A}$ in a security game (Game$_0$ or Game$_1$) as a black box as follows.*

INIT: $\mathcal{B}$ is given a Problem 1 instance $(\Gamma, \mathbb{D}, \hat{\mathbb{D}}^*, \boldsymbol{e}_{\beta,1}, \{\boldsymbol{e}_i\}_{i\in[\![2,n]\!]})$.

SETUP: $\mathcal{B}$ plays a role of the challenger in the security game against adversary $\mathcal{A}$. She provides $\mathcal{A}$ a public key pk $= (\Gamma, e(g_1,g_2)^{\alpha \boldsymbol{d}_0.\boldsymbol{d}_0^*}, g_1^{\boldsymbol{d}_0}, g_1^{\boldsymbol{d}_{4L+1}}, \boldsymbol{h}_1 = g_1^{\boldsymbol{d}_1}, \cdots, \boldsymbol{h}_L = g_1^{\boldsymbol{d}_L})$ of Game$_{0'}$ (and Game$_1$).

KEY QUERY: When a key query is issued for a pattern $\boldsymbol{P}$, $\mathcal{B}$ answers normal key sk$_{\boldsymbol{P}}$, that is computed using $\hat{\mathbb{B}}^*$ of the Problem 1 instance.

CHALLENGE: When $\mathcal{B}$ receives an encryption query with challenge plaintext m and patterns $\boldsymbol{P}^0, \boldsymbol{P}^1$ from $\mathcal{A}$, $\mathcal{B}$ computes the challenge ciphertext $(c_1, \boldsymbol{c}_2)$ s.t.,

$$c_1 = m \cdot e(g_1,g_2)^{s_1 \alpha \boldsymbol{d}_0 \boldsymbol{d}_0^*} \qquad \boldsymbol{c}_2 = g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1}} \cdot \boldsymbol{e}_{\beta,1} \cdot \prod\nolimits_{i \in \bar{W}(\boldsymbol{P}^b)\setminus\{1\}} \boldsymbol{e}_i,$$

where $s_1, s_2 \leftarrow \mathbb{Z}_p$, $b \leftarrow \{0,1\}$ and $\{\boldsymbol{d}_i\}_{i=0,4L+1}, \boldsymbol{e}_{\beta,1}, \{\boldsymbol{e}_i\}_{i=2,\cdots,L}$ is part of the Problem 1 instance.

GUESS: $\mathcal{A}$ finally outputs bit b'. If $b = b'$, $\mathcal{B}$ outputs $\beta' = 1$. Otherwise, $\mathcal{B}$ outputs $\beta' = 0$.

Let us see that the distribution of $\mathcal{A}$'s view in the above-mentioned game, simulated by $\mathcal{B}$ given a Problem 1 instance with $\beta \in \{0,1\}$, is the same as that in Game$_0$ (resp.

Game$_1$) if $\beta = 0$ (resp. $\beta = 1$).

We will consider the distribution of $\boldsymbol{c}_2$. When $\beta = 0$, challenge ciphertext element $\boldsymbol{c}_2$ is

$$\begin{aligned}\boldsymbol{c}_2 &= g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}} \cdot \boldsymbol{e}_{\beta,1} \cdot \prod_{i\in\bar{W}(\boldsymbol{P}^b)\setminus\{1\}} \boldsymbol{e}_i \\ &= g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}+\omega\boldsymbol{d}_1+\gamma\boldsymbol{d}_{4L+1}+\omega \sum\limits_{i\in\bar{W}(\boldsymbol{P}^b)\setminus\{1\}} \boldsymbol{d}_i} \\ &= g_1^{s_1\boldsymbol{d}_0+s_3 \sum\limits_{i\in\bar{W}(\boldsymbol{P}^b)} +s_2'\boldsymbol{d}_{4L+1}}\end{aligned}$$

where $s_3 = \omega, s_2' = s_2 + \gamma, s_1 \in \mathbb{Z}_p$ are uniformly and independently distributed. When $\beta = 1$, challenge ciphertext element $\boldsymbol{c}_2$ is

$$\begin{aligned}\boldsymbol{c}_2 = \; & g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}} \cdot \boldsymbol{e}_{\beta,1} \cdot \prod_{i\in\bar{W}(\boldsymbol{P}^b)\setminus\{1\}} \boldsymbol{e}_i \\ &= g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}+\omega\boldsymbol{d}_1+z\boldsymbol{d}_{L+1}+\gamma\boldsymbol{d}_{4L+1}+\omega \sum\limits_{i\in\bar{W}(\boldsymbol{P}^b)\setminus\{1\}} \boldsymbol{d}_i} \\ &= g_1^{s_1\boldsymbol{d}_0+s_3 \sum\limits_{i\in\bar{W}(\boldsymbol{P}^b)} +t\boldsymbol{d}_{L+1}+s_2'\boldsymbol{d}_{4L+1}}\end{aligned}$$

where $t = z, s_3 = \omega, s_2' = s_2 + \gamma, s_1 \in \mathbb{Z}_p$ are uniformly and independently distributed. Therefore, the above $c_1, \boldsymbol{c}_2$ give a challenge ciphertext in Game$_0$ when $\beta = 0$ and that in Game$_1$ when $\beta = 1$. Thus,

$$\begin{aligned}&\left|\mathsf{Adv}_{\mathcal{A}}^{(0')}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(1)}(\lambda)\right| \\ = \; &\left|\Pr\left[\mathcal{B}_1(\lambda,\varrho) \to 1 | \varrho \leftarrow \mathcal{G}_0^{P1}(\lambda, L)\right] - \Pr\left[\mathcal{B}_1(\lambda,\varrho) \to 1 | \varrho \leftarrow \mathcal{G}_1^{P1}(\lambda, L)\right]\right| \\ &\leq \mathsf{Adv}_{\mathcal{B}_1}^{P1}(\lambda).\end{aligned}$$

This complete the proof of lemma 4.3.7. □

Step 2

$h = 1, \cdots, Q$:

$\mathsf{Game}_{2-(h-1)-4}$	Lemma 4.3.8	Game_{2-h-1}
ct: **temporal 2** $\mathsf{sk}_1, \cdots, \mathsf{sk}_h$: **temporal 2** $\mathsf{sk}_{h+1}, \cdots, \mathsf{sk}_Q$: **normal**	= **Conceptually change**	ct: **temporal 1** $\mathsf{sk}_1, \cdots, \mathsf{sk}_{h-1}$: **temporal 2** $\mathsf{sk}_h, \cdots, \mathsf{sk}_Q$: **normal**

Lemma 4.3.8 *For any adversary $\mathcal{A}$, $\left|\mathsf{Adv}_{\mathcal{A}}^{(2-(h-1)-4)}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(2-h-1)}(\lambda)\right| \leq \epsilon$, for $\epsilon = 4/p$ when $h = 1$ and $\epsilon = 3/p$ when $h \geq 2$.*

Proof 4.3.8 *We start with the case $h = 1$, i.e. the proof for $\left|\mathsf{Adv}_{\mathcal{A}}^{(1)}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(2-1-1)}(\lambda)\right| \leq 4/p$.*

Game_{2-0-4} / Game_1		Game_{2-1-1}
ct: **temporal 0** $\mathsf{sk}_1, \cdots, \mathsf{sk}_Q$: **normal**	**Conceptually change**	ct: **temporal 1** $\mathsf{sk}_1, \cdots, \mathsf{sk}_Q$: **normal**

We define an intermediate game, $\mathsf{Game}_{1'}$, and will show the equivalence of the distribution of the views of $\mathcal{A}$ in Game_1 and that in $\mathsf{Game}_{1'}$ and those in Game_{2-1-1} and in $\mathsf{Game}_{1'}$.

$\mathsf{Game}_{1'}$: $\mathsf{Game}_{1'}$ is the same as Game_1 except that the $\boldsymbol{c}_2$ of the challenge ciphertext for (challenge plaintext m and) patterns $\boldsymbol{P}^0, \boldsymbol{P}^1$ is:

$$\boldsymbol{c}_2 = g_1^{s_1\boldsymbol{d}_0+s_3 \sum\limits_{i\in\overline{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i + \sum\limits_{i=1}^{2L} r_i\boldsymbol{d}_{L+i}+s_2\boldsymbol{d}_{4L+1}}$$

where $r_i \leftarrow \mathbb{Z}_p$ for $i \in [2L]$, $\boldsymbol{r} = (r_1, \cdots, r_{2L}) \neq \boldsymbol{0}^{2L}$, and all the other variables are generated as in Game_1.

Let us see that the distribution of $(pk, \left\{sk^{(j)}\right\}_{j\in[Q]}, c_1, \boldsymbol{c}_2)$ in Game_1 and that in $\mathsf{Game}_{1'}$ are equivalent except with negligible probability.

We will consider the distribution in Game_1. We define new dual orthonormal bases $(\mathbb{B}, \mathbb{B}^*)$ below. Pick $\boldsymbol{F} \leftarrow GL(2L, \mathbb{Z}_p)$, and set

$$\begin{pmatrix} \boldsymbol{b}_{L+1} \\ \vdots \\ \boldsymbol{b}_{3L} \end{pmatrix} = \boldsymbol{F}^{-1} \cdot \begin{pmatrix} \boldsymbol{d}_{L+1} \\ \vdots \\ \boldsymbol{d}_{3L} \end{pmatrix}, \quad \begin{pmatrix} \boldsymbol{b}^*_{L+1} \\ \vdots \\ \boldsymbol{b}^*_{3L} \end{pmatrix} = \boldsymbol{F}^{\top} \cdot \begin{pmatrix} \boldsymbol{d}^*_{L+1} \\ \vdots \\ \boldsymbol{d}^*_{3L} \end{pmatrix},$$

and

$$\mathbb{B} = (\boldsymbol{d}_0, \cdots, \boldsymbol{d}_L, \boldsymbol{b}_{L+1}, \cdots, \boldsymbol{b}_{3L}, \boldsymbol{d}_{3L+1}, \cdots, \boldsymbol{d}_{4L+1})$$
$$\mathbb{B}^* = (\boldsymbol{d}^*_0, \cdots, \boldsymbol{d}^*_L, \boldsymbol{b}^*_{L+1}, \cdots, \boldsymbol{b}^*_{3L}, \boldsymbol{d}^*_{3L+1}, \cdots, \boldsymbol{d}^*_{4L+1}).$$

Then, $\mathbb{B}, \mathbb{B}^*$ *are dual orthonormal bases. Notice that then* $\boldsymbol{d}_{L+1}$ *is equal to* $\boldsymbol{F} \cdot \begin{pmatrix} \boldsymbol{b}_{L+1} \\ \vdots \\ \boldsymbol{b}_{3L} \end{pmatrix}$*, thus can be written as* $\boldsymbol{d}_{L+1} = f_{1,1}\boldsymbol{b}_{L+1} + f_{1,2}\boldsymbol{b}_{L+2} + \cdots + f_{1,2L}\boldsymbol{b}_{3L}$*, with*

$$\boldsymbol{F} = \begin{pmatrix} f_{1,1} & f_{1,2}, & \cdots & f_{1,2L} \\ f_{2,1} & f_{2,2}, & \cdots & f_{2,2L} \\ \vdots & & & \\ f_{2L,1} & f_{2L,2}, & \cdots & f_{2L,2L} \end{pmatrix}.$$

Challenge ciphertext $\boldsymbol{c}_2$ *is expressed as*

$$g_1^{s_1\boldsymbol{d}_0+s_3 \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i+t\boldsymbol{b}_{L+1}+s_2\boldsymbol{d}_{4L+1}} = g_1^{s_1\boldsymbol{b}_0+s_3 \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{b}_i+t(f_{1,1}\boldsymbol{b}_{L+1}+f_{1,2}\boldsymbol{b}_{L+2}+\cdots+f_{1,2L}\boldsymbol{b}_{3L})+s_2\boldsymbol{b}_{4L+1}}$$
$$= g_1^{s_1\boldsymbol{b}_0+\boldsymbol{s}_3 \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{b}_i+\sum_{i=1}^{2L} r_i\boldsymbol{b}_{L+i}+s_2\boldsymbol{b}_{4L+1}}$$

where $s_1, s_2, s_3 \leftarrow \mathbb{Z}_p$ *and* $\boldsymbol{r} = (r_i = tf_{1,i})_{i\in[2L]}$*. Vector* $\boldsymbol{r}$ *is uniformly distributed in* $\mathbb{Z}_p^{2L} \backslash \{\boldsymbol{0}^{2L}\}$ *except for probability* $1/p$ *and independent of all the other variables.*

In Game$_1$*,* $sk_{\boldsymbol{P}}$ *is* $g_2^{\alpha\boldsymbol{d}_0^*+\sum_{j\in\mathcal{I}} r_j\boldsymbol{d}_j^*+\sum_{l=1}^{L}\eta_l\cdot\boldsymbol{d}_{3L+l}^*} = g_2^{\alpha\boldsymbol{b}_0^*+\sum_{j\in\mathcal{I}} r_j\boldsymbol{b}_j^*+\sum_{l=1}^{L}\eta_l\cdot\boldsymbol{b}_{3L+l}^*}$*, where* $r, \{\eta_l\}_{l\in[L]} \leftarrow \mathbb{Z}_p$*, for every queried key.*

In the light of the adversary's view, $(\mathbb{B}, \mathbb{B}^*)$ *is consistent with public key* $(\Gamma, e(g_1, g_2)^{\alpha\boldsymbol{d}_0\cdot\boldsymbol{d}_0^*}, g_1^{\boldsymbol{d}_0}, g_1^{\boldsymbol{d}_{4L+1}}, \boldsymbol{h}_1 = g_1^{\boldsymbol{d}_1}, \cdots, \boldsymbol{h}_L = g_1^{\boldsymbol{d}_L})$*. Moreover, the challenge ciphertext in Game*$_1$ *can be conceptually changed to that in Game*$_{1'}$ *except with probability* $1/p$*.*

Let's see that the distribution of $(pk, \left\{sk^{(j)}\right\}_{j\in[Q]}, c_1, \boldsymbol{c}_2)$ *in Game*$_{2-1-1}$ *and that in Game*$_{1'}$ *are equivalent except with probability* $3/p$*.*

We will consider the distribution in Game$_{2-1-1}$*. We define new dual orthonormal bases* $(\mathbb{B}, \mathbb{B}^*)$ *as above. Challenge ciphertext* $\boldsymbol{c}_2$ *is expressed as*

$$g_1^{s_1\boldsymbol{d}_0+s_3 \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i+t \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_{L+i}+u \sum_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i}+\tilde{u} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{2L+i}+s_2\boldsymbol{d}_{4L+1}}$$
$$= g_1^{s_1\boldsymbol{b}_0+s_3 \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{b}_i+t \sum_{i\in\bar{W}(\boldsymbol{P}^b)} (\sum_{j=1}^{2N} f_{i,j}\boldsymbol{b}_{L+j})+u \sum_{i\in\bar{W}(\boldsymbol{P}^0)} (\sum_{j=1}^{2L} f_{i,j}\boldsymbol{b}_{L+j}) \quad \tilde{u} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} (\sum_{j=1}^{2L} f_{i,j}\boldsymbol{b}_{L+j})+s_2\boldsymbol{b}_{4L+1}} \cdot g_1$$

$$= g_1^{s_1 \boldsymbol{b}_0 + s_3 \sum\limits_{i \in \bar{W}(\boldsymbol{P}^b)} \boldsymbol{b}_i + \sum\limits_{i=1}^{2L} r_i \boldsymbol{b}_{L+i} + s_2 \boldsymbol{b}_{4L+1}}$$

where $s_1, s_2, s_3 \leftarrow \mathbb{Z}_p$ *and vector* $\boldsymbol{r}$ *such that for* $i \in [2L]$, $r_i = t \sum\limits_{j \in \bar{W}(\boldsymbol{P}^b)} f_{j,i} + u \sum\limits_{j \in \bar{W}(\boldsymbol{P}^0)} f_{j,i} + \tilde{u} \sum\limits_{j \in \bar{W}(\boldsymbol{P}^1)} f_{j,i}$.

Vector $\boldsymbol{r} \neq \boldsymbol{0}^{2L}$ *except with probability* $3/p$*, is uniformly distributed in* $\mathbb{Z}_p^{2L} \setminus \{\boldsymbol{0}^{2L}\}$*, and independent of all the other variables. For the queried keys, the same as above holds also in Game*$_{2-1-1}$.

In the light of the adversary's view, $(\mathbb{B}, \mathbb{B}^*)$ *is consistent with public key* $(\Gamma, e(g_1, g_2)^{\alpha \boldsymbol{d}_0 \cdot \boldsymbol{d}_0^*}, g_1^{\boldsymbol{d}_0}, g_1^{\boldsymbol{d}_{4L+1}}, \boldsymbol{h}_1 = g_1^{\boldsymbol{d}_1}, \cdots, \boldsymbol{h}_L = g_1^{\boldsymbol{d}_L})$*. Moreover, the challenge ciphertext in Game*$_{2-1-1}$ *can be conceptually changed to that in Game*$_{1'}$ *except with probability* $3/p$.

This completes the proof when $h = 1$.

Now $h \geq 2$*, i.e. proof for* $\left| \mathsf{Adv}_{\mathcal{A}}^{(2-(h-1)-4)}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(2-h-1)}(\lambda) \right| \leq 3/p$.

$h = 2, \cdots, Q$:

$\mathsf{Game}_{2-(h-1)-4}$		Game_{2-h-1}
ct: **temporal 2** $\mathsf{sk}_1, \cdots, \mathsf{sk}_h$: **temporal 2** $\mathsf{sk}_{h+1}, \cdots, \mathsf{sk}_Q$: **normal**	**Conceptually change**	ct: **temporal 1** $\mathsf{sk}_1, \cdots, \mathsf{sk}_{h-1}$: **temporal 2** $\mathsf{sk}_h, \cdots, \mathsf{sk}_Q$: **normal**

We define an intermediate game, Game$_{2-(h-1)-4'}$*, and will show the equivalence of the distribution of the views of* $\mathcal{A}$ *in Game*$_{2-(h-1)-4}$ *and that in Game*$_{2-(h-1)-4'}$ *and those in Game*$_{2-h-1}$ *and in Game*$_{2-(h-1)-4'}$.

Game$_{2-(h-1)-4'}$*: Game*$_{2-(h-1)-4'}$ *is the same as Game*$_{2-(h-1)-4}$ *except that the element* $\boldsymbol{c}_2$ *of the challenge ciphertext for (challenge plaintext m and) patterns* $\boldsymbol{P}^0, \boldsymbol{P}^1$ *is:*

$$\boldsymbol{c}_2 = g_1^{s_1 \boldsymbol{d}_0 + s_3 \sum\limits_{i \in \bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i + \sum\limits_{i=1}^{L} r_i \boldsymbol{d}_{L+i} + u \sum\limits_{i \in \bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i} + \tilde{u} \sum\limits_{i \in \bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{2L+i} + s_2 \boldsymbol{d}_{4L+1}}$$

where $r_i \leftarrow \mathbb{Z}_p$ *for* $i \in [L]$, $\boldsymbol{r} = (r_1, \cdots, r_L) \neq \boldsymbol{0}^L$*, and all the other variables are generated as in Game*$_{2-(h-1)-4}$.

Let's see that the distribution of $(pk, \left\{sk^{(j)}\right\}_{j\in[Q]}, c_1, \boldsymbol{c}_2)$ *in Game*$_{2-(h-1)-4}$ *and that in Game*$_{2-(h-1)-4'}$ *are equivalent except with probability* $2/p$.

We will consider the distribution in Game$_{2-(h-1)-4}$. *We define new dual orthonormal bases* $(\mathbb{B}, \mathbb{B}^*)$ *below. We generate* $\boldsymbol{F} \leftarrow GL(L, \mathbb{Z}_p)$, *and set*

$$\begin{pmatrix} \boldsymbol{b}_{L+1} \\ \vdots \\ \boldsymbol{b}_{2L} \end{pmatrix} = \boldsymbol{F}^{-1} \begin{pmatrix} \boldsymbol{d}_{L+1} \\ \vdots \\ \boldsymbol{d}_{2L} \end{pmatrix} \quad \begin{pmatrix} \boldsymbol{b}^*_{L+1} \\ \vdots \\ \boldsymbol{b}^*_{2L} \end{pmatrix} = \boldsymbol{F}^{\top} \begin{pmatrix} \boldsymbol{d}^*_{L+1} \\ \vdots \\ \boldsymbol{d}^*_{2L} \end{pmatrix}$$

and

$$\mathbb{B} = (\boldsymbol{d}_0, \cdots, \boldsymbol{d}_L, \boldsymbol{b}_{L+1}, \cdots, \boldsymbol{b}_{2L}, \boldsymbol{d}_{2L+1}, \cdots, \boldsymbol{d}_{4L+1})$$
$$\mathbb{B}^* = (\boldsymbol{d}^*_0, \cdots, \boldsymbol{d}^*_L, \boldsymbol{b}^*_{L+1}, \cdots, \boldsymbol{b}^*_{2L}, \boldsymbol{d}^*_{2L+1}, \cdots, \boldsymbol{d}^*_{4L+1}).$$

Then $\mathbb{B}$ *and* $\mathbb{B}^*$ *are dual orthonormal bases. Challenge ciphertext* $\boldsymbol{c}_2$ *is expressed as*

$$\begin{aligned} & g_1^{s_1\boldsymbol{d}_0+s_3\sum_{i\in\bar{W}(\boldsymbol{P}^b)}\boldsymbol{d}_i+t\sum_{i\in\bar{W}(\boldsymbol{P}^0)}\boldsymbol{d}_{L+i}+\tilde{t}\sum_{i\in\bar{W}(\boldsymbol{P}^1)}\boldsymbol{d}_{L+i}+u\sum_{i\in\bar{W}(\boldsymbol{P}^0)}\boldsymbol{d}_{2L+i}+\tilde{u}\sum_{i\in\bar{W}(\boldsymbol{P}^0)}\boldsymbol{d}_{2L+i}+s_2\boldsymbol{d}_{4L+1}} \\ = \; & g_1^{s_1\boldsymbol{b}_0+s_3\sum_{i\in\bar{W}(\boldsymbol{P}^b)}\boldsymbol{b}_i+t\sum_{i\in\bar{W}(\boldsymbol{P}^0)}(\sum_{j=1}^{L}f_{i,j}\boldsymbol{b}_{L+j})+\tilde{t}\sum_{i\in\bar{W}(\boldsymbol{P}^1)}(\sum_{j=1}^{L}f_{i,j}\boldsymbol{b}_{L+j})+u\sum_{i\in\bar{W}(\boldsymbol{P}^0)}\boldsymbol{b}_{2L+i}+\tilde{u}\sum_{i\in\bar{W}(\boldsymbol{P}^0)}\boldsymbol{b}_{2L+i}+s_2\boldsymbol{b}_{4L+1}} \\ = \; & g_1^{s_1\boldsymbol{b}_0+s_3\sum_{i\in\bar{W}(\boldsymbol{P}^b)}\boldsymbol{b}_i+\sum_{i=1}^{L}r_i\boldsymbol{b}_{L+i}+u\sum_{i\in\bar{W}(\boldsymbol{P}^0)}\boldsymbol{b}_{2L+i}+\tilde{u}\sum_{i\in\bar{W}(\boldsymbol{P}^0)}\boldsymbol{b}_{2L+i}+s_2\boldsymbol{b}_{4L+1}} \end{aligned}$$

where $s_1, s_2, s_3, u, \tilde{u} \leftarrow \mathbb{Z}_p$ *and* $\boldsymbol{r}$ *is defined such that for* $i \in [L]$, $r_i = t\sum_{j\in\bar{W}(\boldsymbol{P}^0)} f_{j,i} + \tilde{t}\sum_{j\in\bar{W}(\boldsymbol{P}^1)} f_{j,i}$. *Thus* $\boldsymbol{r} \neq \boldsymbol{0}^L$ *except with probability* $2/p$, *is uniformly distributed and independent of all the other variables.*

When $1 \leq j \leq h-1$, *the* j*-th queried key* $sk_{\boldsymbol{P}^{(j)}}$ *is*

$$\begin{aligned} & g_2^{\alpha\boldsymbol{d}^*_0+\sum_{j\in\mathcal{I}}r_j\boldsymbol{d}^*_j+\sum_{j\in\mathcal{I}}x_j\boldsymbol{d}_{2L+j}+\sum_{l=1}^{L}\eta_l\cdot\boldsymbol{d}^*_{3L+l}} \\ = \; & g_2^{\alpha\boldsymbol{b}^*_0+\sum_{j\in\mathcal{I}}r_j\boldsymbol{b}^*_j+\sum_{j\in\mathcal{I}}x_j\boldsymbol{b}_{2L+j}+\sum_{l=1}^{L}\eta_l\cdot\boldsymbol{b}^*_{3L+l}} \end{aligned}$$

where $\{x_j, r_j\}_{i,j\in[L]}, \{\eta_l\}_{l\in[L]} \leftarrow \mathbb{Z}_p$. *When* $h \leq j \leq Q$, *the* j*-th queried key* $sk_{\boldsymbol{P}^{(j)}}$ *is*

$$g_2^{\alpha\boldsymbol{d}^*_0+\sum_{j\in\mathcal{I}}r_j\boldsymbol{d}^*_j+\sum_{l=1}^{L}\eta_l\cdot\boldsymbol{d}^*_{3L+l}}$$

$$= g_2^{\alpha \boldsymbol{b}_0^* + \sum_{j \in \mathcal{I}} r_j \boldsymbol{b}_j^* + \sum_{l=1}^{L} \eta_l \cdot \boldsymbol{b}_{3L+l}^*}$$

where $\{r_j\}_{j \in [L]}, \{\eta_l\}_{l \in [L]} \leftarrow \mathbb{Z}_p$.

In the light of the adversary's view, $(\mathbb{B}, \mathbb{B}^*)$ *is consistent with public key* $(\Gamma, e(g_1, g_2)^{\alpha \boldsymbol{d}_0 \cdot \boldsymbol{d}_0^*}, g_1^{\boldsymbol{d}_0}, g_1^{\boldsymbol{d}_{4L+1}}, \boldsymbol{h}_1 = g_1^{\boldsymbol{d}_1}, \cdots, \boldsymbol{h}_L = g_1^{\boldsymbol{d}_L})$. *Moreover, the challenge ciphertext in Game*$_{2-(h-1)-4}$ *can be conceptually changed to that in Game*$_{2-(h-1)-4'}$ *except with probability* $2/p$.

Let us see that the distribution of $(pk, \left\{sk^{(j)}\right\}_{j \in [Q]}, c_1, \boldsymbol{c}_2)$ *in Game*$_{2-h-1}$ *and that in Game*$_{2-(h-1)-4'}$ *are equivalent except with probability* $1/p$.

We will consider the distribution in Game$_{2-h-1}$. *We define new dual orthonormal bases* $(\mathbb{B}, \mathbb{B}^*)$ *as above. Challenge ciphertext* $\boldsymbol{c}_2$ *is expressed as*

$$\begin{aligned}
& g_1^{s_1 \boldsymbol{d}_0 + s_3 \sum_{i \in \bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i + t \sum_{i \in \bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_{L+i} + u \sum_{i \in \bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i} + \tilde{u} \sum_{i \in \bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i} + s_2 \boldsymbol{d}_{4L+1}} \\
= \quad & g_1^{s_1 \boldsymbol{b}_0 + s_3 \sum_{i \in \bar{W}(\boldsymbol{P}^b)} \boldsymbol{b}_i + t \sum_{i \in \bar{W}(\boldsymbol{P}^b)} (\sum_{j=1}^{L} \boldsymbol{b}_{L+j}) + u \sum_{i \in \bar{W}(\boldsymbol{P}^0)} \boldsymbol{b}_{2L+i} + \tilde{u} \sum_{i \in \bar{W}(\boldsymbol{P}^0)} \boldsymbol{b}_{2L+i} + s_2 \boldsymbol{b}_{4L+1}} \\
= \quad & g_1^{s_1 \boldsymbol{b}_0 + s_3 \sum_{i \in \bar{W}(\boldsymbol{P}^b)} \boldsymbol{b}_i + \sum_{i=1}^{L} r_i \boldsymbol{d}_{L+i} + \sum_{i=1}^{L} \boldsymbol{b}_{L+i} + u \sum_{i \in \bar{W}(\boldsymbol{P}^0)} \boldsymbol{b}_{2L+i} + \tilde{u} \sum_{i \in \bar{W}(\boldsymbol{P}^0)} \boldsymbol{b}_{2L+i} + s_2 \boldsymbol{b}_{4L+1}}
\end{aligned}$$

where $s_1, s_2, s_3, u, \tilde{u} \leftarrow \mathbb{Z}_p$ *and vector* $\boldsymbol{r}$ *such that for* $i \in [L]$, $r_i = t \sum_{j \in \bar{W}(\boldsymbol{P}^b)} f_{j,i}$. *Vector* $\boldsymbol{r} \neq 0$ *except with probability* $1/p$, *then is uniformly distributed in* $\mathbb{Z}_p^L \setminus \{\mathbf{0}^L\}$, *and independent of all the other variables.*
For the queried keys, the same as above holds also in Game$_{2-h-1}$.

In the light of the adversary's view, $(\mathbb{B}, \mathbb{B}^*)$ *is consistent with public key* $(\Gamma, e(g_1, g_2)^{\alpha \boldsymbol{d}_0 \cdot \boldsymbol{d}_0^*}, g_1^{\boldsymbol{d}_0}, g_1^{\boldsymbol{d}_{4L+1}}, \boldsymbol{h}_1 = g_1^{\boldsymbol{d}_1}, \cdots, \boldsymbol{h}_L = g_1^{\boldsymbol{d}_L})$. *Moreover, the challenge ciphertext in Game*$_{2-h-1}$ *can be conceptually changed to that in Game*$_{2-(h-1)-4'}$ *except with probability* $1/p$.

This completes the proof when $h \geq 2$, *and thus also the proof of lemma 4.3.8.* □

Step 3

$h = 1, \cdots, Q$:

Game_{2-h-1}	Lemma 4.3.9 / $\approx$ / Problem 2	Game_{2-h-2}
ct: **temporal 1** $\mathsf{sk}_1, \cdots, \mathsf{sk}_{h-1}$: **temporal 2** $\mathsf{sk}_h, \cdots, \mathsf{sk}_Q$: **normal**		ct: **temporal 1** $\mathsf{sk}_1, \cdots, \mathsf{sk}_{h-1}$: **temporal 2** sk_h: **temporal 1** $\mathsf{sk}_{h+1}, \cdots, \mathsf{sk}_Q$: **normal**

Lemma 4.3.9 *For any adversary $\mathcal{A}$, there exists a probabilistic adversary $\mathcal{B}$, whose running time is essentially the same as that of $\mathcal{A}$, such that for any security parameter λ, $\left|\mathsf{Adv}_{\mathcal{A}}^{2-h-1}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{2-h-2}(\lambda)\right| \leq \mathsf{Adv}_{\mathcal{B}}^{P2}(\lambda)$.*

Proof 4.3.9 *We construct a probabilistic adversary $\mathcal{B}$ against Problem 2 using an adversary $\mathcal{A}$ in a security game (Game_{2-h-1} or Game_{2-h-2}) as a black box as follows.*

INIT: $\mathcal{B}$ is given an integer h and $(\Gamma, \hat{\mathbb{D}}, \mathbb{D}^*, \{\boldsymbol{h}^*_{\beta,i}, \boldsymbol{e}_i\}_{i\in[Q]})$.

SETUP: $\mathcal{B}$ plays a role of the challenger in the security game against adversary $\mathcal{A}$. She provides $\mathcal{A}$ elements $\Gamma, \hat{\mathbb{D}}'$ of $\mathsf{Game}_{2-(h-1)-4}$ (and Game_{2-h-1}) for public key, where $\hat{\mathbb{D}}' = (\boldsymbol{d}_0, \cdots, \boldsymbol{d}_L, \boldsymbol{d}_{4L+1})$ is obtained from the Problem 2 instance.

KEY QUERY: when the ι-th key query is issued for a pattern $\boldsymbol{P}$, $\mathcal{B}$ answers as follows:

- *When $1 \leq \iota \leq h-1$, $\mathcal{B}$ answers keys of temporal 2 form, that are computed using $\mathbb{B}^*$ of the Problem 2 instance.*
- *When $\iota = h$, $\mathcal{B}$ calculates $\mathsf{sk}_{\boldsymbol{P}}$ using $\{\boldsymbol{h}^*_{\beta,i}\}_{i\in[Q]}$, $\{\boldsymbol{d}^*_i\}_{i=0,3L+1,\cdots,4L}$ of the Problem 2 instance as follows: $\boldsymbol{\eta} = (\eta_1, \cdots, \eta_L) \leftarrow \mathbb{Z}_p^L, \xi_i \leftarrow \mathbb{Z}_p$ for $i \in [L]$*

$$\mathsf{sk}_{\boldsymbol{P}} = g_2^{\alpha \boldsymbol{d}^*_0} \cdot \prod_{i\in\mathcal{I}} \boldsymbol{h}^{*\xi_i}_{\beta,i} \cdot g_2^{\sum_{i\in[L]} \eta_i \boldsymbol{d}^*_{3L+i}}$$

- *When $\iota \geq h+1$, $\mathcal{B}$ answers normal keys using $\mathbb{B}^*$ of the Problem 2 instance.*

CHALLENGE: when $\mathcal{B}$ receives an encryption query with challenge plaintext m and patterns $\boldsymbol{P}^0, \boldsymbol{P}^1$ from $\mathcal{A}$, she computes challenge ciphertext $(c_1, \boldsymbol{c}_2)$ s.t.

$$c_1 = m \cdot e(g_1, g_2)^{s_1 \boldsymbol{d}_0 \cdot \boldsymbol{d}^*_0}$$

$$\boldsymbol{c}_2 = g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1}} \cdot \prod_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{e}_i \cdot g_1^{u \sum_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i} + \tilde{u} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{2L+i}}$$

where $s_1, s_2, u, \tilde{u} \leftarrow \mathbb{Z}_p, b \leftarrow \{0,1\}$ *and* $\{\boldsymbol{d}_i\}_{i=0,2L+1,\cdots,3L,4L+1}$, $\{\boldsymbol{e}_i\}_{i\in[L]}$ *is a part of the Problem 2 instance.*

GUESS*:* $\mathcal{A}$ *outputs bit* b'*. If* $b = b'$*,* $\mathcal{B}_{2-1}$ *outputs* $\beta' = 1$*. Otherwise,* $\mathcal{B}_{2-1}$ *outputs* $\beta' = 0$*.*

Let us see that if $\beta = 0$*, then the distribution of* $\mathcal{A}$*'s view in the above mentioned game, simulated by* $\mathcal{B}$*, is the same that in Game*$_{2-h-1}$*, and that if* $\beta = 1$ *it is the same that in Game*$_{2-h-2}$*. Ciphertext element* $\boldsymbol{c}_2$ *is*

$$\begin{aligned} & g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}} \cdot \prod_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{e}_i \cdot g_1^{u \sum\limits_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i} + \tilde{u} \sum\limits_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{2L+i}} \\ = \quad & g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}} \cdot \prod_{i\in\bar{W}(\boldsymbol{P}^b)} g_1^{\omega\boldsymbol{d}_i+\sigma\boldsymbol{d}_{L+i}} \cdot g_1^{u \sum\limits_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i} + \tilde{u} \sum\limits_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{2L+i}} \\ = \quad & g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}+\omega \sum\limits_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i+\sigma \sum\limits_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_{L+i}+u \sum\limits_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i}+\tilde{u} \sum\limits_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{2L+i}} \end{aligned}$$

where $s_1, s_2, \omega, \sigma, u, \tilde{u} \in \mathbb{Z}_p$ *are uniformly distributed.*

Now let us see the value of $\mathit{sk}_{\boldsymbol{P}}$*. When* $\beta = 0$*,* $\mathit{sk}_{\boldsymbol{P}}$ *in case* $\iota = h$

$$\begin{aligned} g_2^{\alpha\boldsymbol{d}_0^*} \cdot \prod_{i\in\mathcal{I}} \boldsymbol{h}_{\beta,i}^{*\xi_i} \cdot g_2^{\sum\limits_{i\in[L]} \eta_i\boldsymbol{d}_{3L+i}^*} &= g_2^{\alpha\boldsymbol{d}_0^*} \cdot \prod_{i\in\mathcal{I}} g_2^{\delta\xi_i\boldsymbol{d}_i^*+\xi_i\delta_0\boldsymbol{d}_{3L+i}^*} \cdot g_2^{\sum\limits_{i\in[L]} \eta_i\boldsymbol{d}_{3L+i}^*} \\ &= g_2^{\alpha\boldsymbol{d}_0^*+\sum\limits_{j\in\mathcal{I}} \delta\xi_i\boldsymbol{d}_j^*+\sum\limits_{i\in[L]} \phi_i\boldsymbol{d}_{3L+i}^*} \end{aligned}$$

where α, δ *are uniformly and independently distributed and* $\phi_i = \xi_i\delta_0 + \eta_i$ *if* $i \in \mathcal{I}$ *and* $\phi_i = \eta_i$ *otherwise. Therefore, generated* $\boldsymbol{c}_2, \mathit{sk}_{\boldsymbol{P}}$ *have the same joint distribution as in Game*$_{2-h-1}$*. When* $\beta = 1$*,* $\mathit{sk}_{\boldsymbol{P}}$ *in case* $\iota = h$

$$\begin{aligned} & g_2^{\alpha\boldsymbol{d}_0^*} \cdot \prod_{i\in\mathcal{I}} \boldsymbol{h}_{\beta,i}^{*\xi_i} \cdot g_2^{\sum\limits_{i\in[L]} \eta_i\boldsymbol{d}_{3L+i}^*} \\ = \quad & g_2^{\alpha\boldsymbol{d}_0^*} \cdot \prod_{i\in\mathcal{I}} g_2^{\xi_i\delta\boldsymbol{d}_i^*+\xi_i\tau\boldsymbol{d}_{L+i}+\xi_i\delta_0\boldsymbol{d}_{3L+i}^*} \cdot g_2^{\sum\limits_{i\in[L]} \eta_i\boldsymbol{d}_{3L+i}^*} \\ = \quad & g_2^{\alpha\boldsymbol{d}_0^*+\sum\limits_{i\in\mathcal{I}} \delta\xi_i\boldsymbol{d}_j^*+\sum\limits_{i\in\mathcal{I}} \tau\xi_i\boldsymbol{d}_{L+i}^*+\sum\limits_{i\in[L]} \phi_i\boldsymbol{d}_{3L+i}^*} \end{aligned}$$

where α, δ, τ *are uniformly and independently distributed and* $\phi_i = \xi_i\delta_0 + \eta_i$ *if* $i \in \mathcal{I}$ *and* $x_i = \eta_i$ *otherwise. Therefore, generated* $\boldsymbol{c}_2$ *and* $\mathit{sk}_{\boldsymbol{P}}$ *have the same joint distribution as*

in Game$_{2-h-2}$. *Thus* $\left|\mathsf{Adv}_{\mathcal{A}}^{2-h-1}(\lambda) - \mathcal{A}^{2-h-2}(\lambda)\right| \leq \mathsf{Adv}_{\mathcal{B}}^{P2}(\lambda)$. □

Step 4

$h = 1, \cdots, Q$:

Game_{2-h-2}	Lemma 4.3.10	Game_{2-h-3}
ct: **temporal 1** $\mathsf{sk}_1, \cdots, \mathsf{sk}_{h-1}$: **temporal 2** sk_h: **temporal 1** $\mathsf{sk}_{h+1}, \cdots, \mathsf{sk}_Q$: **normal**	= **Conceptually change**	ct: **temporal 2** $\mathsf{sk}_1, \cdots, \mathsf{sk}_{h-1}$: **temporal 2** sk_h: **temporal 1** $\mathsf{sk}_{h+1}, \cdots, \mathsf{sk}_Q$: **normal**

Lemma 4.3.10 *For any adversary* $\mathcal{A}$, $\left|\mathsf{Adv}_{\mathcal{A}}^{(2-h-2)}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(2-h-3)}\right| \leq \frac{4}{p^{|\mathcal{I}|}} + 5/p$.

This is the only part of the proof that cannot be done as in the original proof. Indeed, as already said, [114] proved that Game_{2-h-2} can be conceptually changed to Game_{2-h-3} with a change of bases and an intermediate game. However, with their change of bases $\mathbb{D}, \mathbb{D}^*$ to $\mathbb{B}, \mathbb{B}^*$, the h-th key of our scheme can no longer decrypt the ciphertext. Thus, the adversary can distinguish the different games as in one case the h-th key decrypts the challenge ciphertext but not in the other case. That is because, with the definition of $\mathbb{B}, \mathbb{B}^*$, some elements of $\mathbb{D}$ (resp. $\mathbb{D}^*$) are now linear combination of elements of $\mathbb{B}$ (resp. $\mathbb{B}^*$). Thus, the set $\bar{W}(\boldsymbol{P}^b) \cap \mathcal{I}$ is no longer equal to $\varnothing$ (the decryption condition) but is equal to $\bar{W}(\boldsymbol{P}^b)$. In our proof, we change the way the new dual orthonormal bases are computed. We define new dual orthonormal bases $(\mathbb{B}, \mathbb{B}^*)$, following the idea of the last lemma in the original proof. This solves the issue raised by our scheme's construction and allows us to prove the indistinguishability between the two games.

Proof 4.3.10 *We will show that distribution* $(pk, \left\{sk^{(j)}\right\}_{j\in[Q]}, c_1, \boldsymbol{c}_2)$ *in* Game_{2-h-2} *and that in* Game_{2-h-3} *are equivalent. For that purpose, we define an intermediate game:* $\mathsf{Game}_{2-h-2'}$*, that is the same as* Game_{2-h-2} *except that element* $\boldsymbol{c}_2$ *of the challenge ciphertext for challenge plaintext m and patterns* $\boldsymbol{P}^0, \boldsymbol{P}^1$ *is:*

$$\boldsymbol{c}_2 = g_1^{s_1\boldsymbol{d}_0 + s_2\boldsymbol{d}_{4L+1} + \sum\limits_{i\in\bar{W}(\boldsymbol{P}^b)} \tilde{s}_i\boldsymbol{d}_i + \sum\limits_{i=1}^{l} \nu_i\boldsymbol{d}_{L+i}} \cdot g_1^{u \sum\limits_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i} + \tilde{u} \sum\limits_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{2L+i}}$$

where $\{\tilde{s}_i \in \mathbb{Z}_p\}_{i\in[L]}, \boldsymbol{\nu} \leftarrow \mathbb{Z}_p^L$ *and all the other variables are generated as in* Game_{2-h-2}*. Notice that* $\boldsymbol{\nu}$ *is equal to zero at position* i *such that* $i \notin \bar{W}(\boldsymbol{P}^0) \wedge i \notin \bar{W}(\boldsymbol{P}^1)$.

Let us see that the distribution $(pk, \left\{sk^{(j)}\right\}_{j\in[Q]}, c_1, \boldsymbol{c}_2)$ *in* Game_{2-h-2} *and that in* $\mathsf{Game}_{2-h-2'}$ *are equivalent except with negligible probability.*

We will consider the distribution in Game_{2-h-2}*. We define new dual orthonormal bases* $(\mathbb{B}, \mathbb{B}^*)$*, following the idea of the last lemma in the original proof. For* $i \in [L]$ *let* $\theta_i, \tau_i \leftarrow \mathbb{Z}_p$ *and set*

$$\boldsymbol{b}_i = \tau_i^{-1}\boldsymbol{d}_i + \theta_i \boldsymbol{d}_{L+i}, \quad \boldsymbol{b}_i^* = \tau_i \boldsymbol{d}_i^* \quad \boldsymbol{b}_{L+i} = \tau_i \boldsymbol{d}_{L+i} \quad \boldsymbol{b}_{L+i}^* = -\theta_i \boldsymbol{d}_i^* + \tau_i^{-1}\boldsymbol{d}_{L+i}^* \ ,$$

and

$$\mathbb{B} = (\boldsymbol{d}_0, \boldsymbol{b}_1 \cdots, \boldsymbol{b}_L, \boldsymbol{b}_{L+1}, \cdots, \boldsymbol{b}_{2L}, \boldsymbol{d}_{2L+1} \cdots \boldsymbol{d}_{4L+1}),$$
$$\mathbb{B}^* = (\boldsymbol{d}_0^*, \boldsymbol{b}_1^*, \cdots, \boldsymbol{b}_L^*, \boldsymbol{b}_{L+1}^*, \cdots, \boldsymbol{b}_{2L}^*, \boldsymbol{d}_{2L+1}^*, \cdots, \boldsymbol{d}_{4L+1}^*).$$

We then easily verify that $\mathbb{B}$ *and* $\mathbb{B}^*$ *are dual orthonormal. The* h*-th queried key and challenge ciphertext* $(sk^{(h)}, c_1, \boldsymbol{c}_2)$ *in* Game_{2-h-2} *are expressed over bases* $(\mathbb{D}, \mathbb{D}^*)$ *and* $(\mathbb{B}, \mathbb{B}^*)$ *as*

$$\begin{aligned}
sk^{(h)} &= g_2^{\alpha \boldsymbol{d}_0^* + \sum_{j\in\mathcal{I}} r_j \boldsymbol{d}_j^* + \sum_{j\in\mathcal{I}} z_j \boldsymbol{d}_{L+j}^* + \sum_{l=1}^{L} \eta_l \boldsymbol{d}_{3L+l}^*} \\
&= g_2^{\alpha \boldsymbol{b}_0^* + \sum_{j\in\mathcal{I}} r_j\tau_j \boldsymbol{b}_j^* + \sum_{j\in\mathcal{I}} z_j(\tau_j \boldsymbol{b}_{L+j}^* + \theta_j \boldsymbol{b}_j^*) + \sum_{l=1}^{L} \eta_l \boldsymbol{b}_{3L+l}^*} \\
&= g_2^{\alpha \boldsymbol{b}_0^* + \sum_{j\in\mathcal{I}} (r_j\tau_j + z_j\theta_j) \boldsymbol{b}_j^* + \sum_{j\in\mathcal{I}} z_j\tau_j \boldsymbol{b}_{L+j}^* + \sum_{l=1}^{L} \eta_l \boldsymbol{b}_{3L+l}^*}
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{c}_2 &= g_1^{s_1\boldsymbol{d}_0 + s_2\boldsymbol{d}_{4L+1} + s_3 \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i + t \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_{L+i} + u \sum_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i} + \tilde{u} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{2L+i}} \\
&= g_1^{s_1\boldsymbol{b}_0 + s_2\boldsymbol{b}_{4L+1} + s_3 \sum_{i\in\bar{W}(\boldsymbol{P}^b)} (\tau_i\boldsymbol{b}_i - \theta_i\boldsymbol{b}_{L+i}) + t \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{b}_{L+i}^* + u \sum_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{b}_{2L+i} \quad \tilde{u} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{b}_{2L+i}} \\
&\qquad \cdot g_1^{s_1\boldsymbol{b}_0 + s_2\boldsymbol{b}_{4L+1} + s_3 \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \tau_i\boldsymbol{b}_i + \sum_{i\in\bar{W}(\boldsymbol{P}^b)} (t - s_3\theta_i)\boldsymbol{b}_{L+i}^* + u \sum_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{b}_{2L+i} \quad \tilde{u} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{b}_{2L+i}}
\end{aligned}$$

$$c_1 = m.e(g_1, g_2)^{s\boldsymbol{d}_0^*\boldsymbol{d}_0} = m.e(g_1, g_2)^{s\boldsymbol{b}_0^*\boldsymbol{b}_0}$$

where $\tilde{r}_j = r_j\tau_j + z_j\theta_j$*,* $\tilde{z}_j = z_j\tau_j$ *for* $j \in \mathcal{I}$ *and* $\tilde{r}_j = w_j = 0$ *otherwise, and* $\tilde{s}_i = s_3\tau_i$*,* $\nu_i = t - s_3\theta_i$ *for* $i \in \bar{W}(\boldsymbol{P}^b)$ *and* $\tilde{s}_i = \nu_i = 0$ *otherwise.* $\tilde{s}_i, \tilde{r}_i, \boldsymbol{\nu}, \boldsymbol{w}$ *are uniformly distributed for the position different of* 0 *and independent of all the other variables except with probability* $\frac{2}{p^{|\mathcal{I}|}} + 2/p$*.*

In the light of the adversary view, both $(\mathbb{D}, \mathbb{D}^*)$ *and* $(\mathbb{B}, \mathbb{B}^*)$ *are consistent with public key pk and the answered keys* $\left\{sk^{(j)}\right\}_{j\neq h}$*. Therefore, by using the above result for the distribution of* $(sk^{(h)}, c_1, \boldsymbol{c}_2)$*,* $\left\{sk^{(j)}\right\}_{j\in[Q]}$ *and* $\boldsymbol{c}_2$ *can be expressed as keys and ciphertext in two ways, in Game*$_{2-h-2}$ *over bases* $(\mathbb{D}, \mathbb{D}^*)$ *and in Game*$_{2-h-2'}$ *over bases* $(\mathbb{B}, \mathbb{B}^*)$*. Thus, Game*$_{2-h-2}$ *can be conceptually changed to Game*$_{2-h-2'}$*, except with probability* $\frac{2}{p^{|\mathcal{I}|}} + 2/p$*.*

Now let us see that the distribution $(pk, \left\{sk^{(j)}\right\}_{j\in[Q]}, c_1, \boldsymbol{c}_2)$ *in Game*$_{2-h-3}$ *and that in Game*$_{2-h-2'}$ *are equivalent except with negligible probability. As above, we set new bases* $(\mathbb{B}, \mathbb{B}^*)$*. The* h*-th queried key* $sk^{(h)}$ *in Game*$_{2-h-3}$ *is expressed as above in bases* $\mathbb{D}^*$ *and* $\mathbb{B}^*$*, and the part of the ciphertext* c_1 *in Game*$_{2-h-3}$ *is given as above. Element* $\boldsymbol{c}_2$ *in Game*$_{2-h-3}$ *is expressed over bases* $\mathbb{D}$ *and* $\mathbb{B}$ *as*

$$g_1^{s_1\boldsymbol{d}_0+s_2\boldsymbol{d}_{4L+1}+s_3\sum_{i\in\bar{W}(\boldsymbol{P}^b)}\boldsymbol{d}_i+t\sum_{i\in\bar{W}(\boldsymbol{P}^0)}\boldsymbol{d}_{L+i}+\tilde{t}\sum_{i\in\bar{W}(\boldsymbol{P}^1)}\boldsymbol{d}_{L+i}+u\sum_{i\in\bar{W}(\boldsymbol{P}^0)}\boldsymbol{d}_{2L+i}+\tilde{u}\sum_{i\in\bar{W}(\boldsymbol{P}^1)}\boldsymbol{d}_{2L+i}}$$

$$= g_1^{s_1\boldsymbol{b}_0+s_2\boldsymbol{b}_{4L+1}+s_3\sum_{i\in\bar{W}(\boldsymbol{P}^b)}(\tau_i\boldsymbol{b}_i-\theta_i\boldsymbol{b}_{L+i})+t\sum_{i\in\bar{W}(\boldsymbol{P}^0)}\tau_i^{-1}\boldsymbol{b}_{L+i}+\tilde{t}\sum_{i\in\bar{W}(\boldsymbol{P}^1)}\tau_i^{-1}\boldsymbol{b}_{L+i}+u\sum_{i\in\bar{W}(\boldsymbol{P}^0)}\boldsymbol{b}_{2L+i}} \cdot g_1^{\tilde{u}\sum_{i\in\bar{W}(\boldsymbol{P}^1)}\boldsymbol{b}_{2L+i}}$$

We can define $\boldsymbol{\nu}$ *the coefficient vector of* $(\boldsymbol{b}_{L+1}, \cdots, \boldsymbol{d}_{2L})$ *as for* $i \in [L]$*:*

$$\nu_i = \begin{cases} \tau_i^{-1}t & \text{if } i \in \bar{W}(\boldsymbol{P}^0) \wedge i \notin \bar{W}(\boldsymbol{P}^1) \wedge b = 1 \\ \tau_i^{-1}\tilde{t} & \text{if } i \in \bar{W}(\boldsymbol{P}^1) \wedge i \notin \bar{W}(\boldsymbol{P}^0) \wedge b = 0 \\ -s_3\theta_i + \tau_i^{-1}t & \text{if } i \in \bar{W}(\boldsymbol{P}^0) \wedge i \notin \bar{W}(\boldsymbol{P}^1) \wedge b = 0 \\ -s_3\theta_i + \tau_i^{-1}\tilde{t} & \text{if } i \in \bar{W}(\boldsymbol{P}^1) \wedge i \notin \bar{W}(\boldsymbol{P}^0) \wedge b = 1 \\ -s_3\theta_i + \tau_i^{-1}t + \tau_i^{-1}\tilde{t} & \text{if } i \in \bar{W}(\boldsymbol{P}^0) \wedge i \in \bar{W}(\boldsymbol{P}^1) \\ 0 & \text{otherwise} \end{cases}$$

and $\tilde{s}_i = s_3\tau_i$ *for* $i \in \bar{W}(\boldsymbol{P}^b)$*.* $\boldsymbol{\nu}, \{\tilde{s}_i\}_{i=1,\cdots,L}$ *are uniformly distributed for the position different of* 0 *and independent of the other variables except with probability* $3/p$ *.*

Similar as above, we see that $\left\{sk^{(j)}\right\}_{j\in[Q]}$ *and* $\boldsymbol{c}_2$ *can be expressed as keys and ciphertext in two ways, in Game*$_{2-h-3}$ *over bases* $(\mathbb{D}, \mathbb{D}^*)$ *and in Game*$_{2-h-2'}$ *over bases* $(\mathbb{B}, \mathbb{B}^*)$*. Thus Game*$_{2-h-3}$ *can be conceptually changed to Game*$_{2-h-2'}$ *except with probability* $\frac{2}{p^{|\mathcal{I}|}} + 3/p$*. Combining both, we obtain lemma 4.3.10.* □

Step 5

$h = 1, \cdots, Q$:

Game_{2-h-3}	Lemma 4.3.11	Game_{2-h-4}
ct: temporal 2 $\mathsf{sk}_1, \cdots, \mathsf{sk}_{h-1}$: temporal 2 sk_h: temporal 1 $\mathsf{sk}_{h+1}, \cdots, \mathsf{sk}_Q$: normal	≈ Problem 3	ct: temporal 2 $\mathsf{sk}_1, \cdots, \mathsf{sk}_h$: temporal 2 $\mathsf{sk}_{h+1}, \cdots, \mathsf{sk}_Q$: normal

Lemma 4.3.11 *For any adversary* $\mathcal{A}$*, there exists a probabilistic adversary* $\mathcal{B}$*, whose running time is essentially the same as that of* $\mathcal{A}$*, such that for any security parameter* λ*,*

$$\left|\mathsf{Adv}_{\mathcal{A}}^{2-h-3}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{2-h-4}(\lambda)\right| \leq \mathsf{Adv}_{\mathcal{B}}^{P3}(\lambda).$$

Proof 4.3.11 *We construct a probabilistic adversary* $\mathcal{B}$ *against Problem 3 using an adversary* $\mathcal{A}$ *in a security game (Game*$_{2-h-3}$ *or Game*$_{2-h-4}$*) as a black box as follows.*

`INIT`*:* $\mathcal{B}$ *is given an integer* h *and a Problem 3 instance,* $(\Gamma, \hat{\mathbb{D}}, \hat{\mathbb{D}}^*, \left\{\boldsymbol{h}^*_{\beta,i}, \boldsymbol{e}_i, \boldsymbol{f}_i\right\}_{i\in[n]})$*.*

`SETUP`*:* $\mathcal{B}$ *plays a role of the challenger in the security game against adversary* $\mathcal{A}$*. She provides* $\mathcal{A}$ *a public key* $pk = (\Gamma, e(g_1, g_2)^{\alpha \boldsymbol{d}_0 \boldsymbol{d}_0^*}, g_1^{\boldsymbol{d}_0}, g_1^{\boldsymbol{d}_{4L+1}}, g_1^{\boldsymbol{d}_1}, \cdots, g_1^{\boldsymbol{d}_L})$ *of Game*$_{2-h-3}$ *(and Game*$_{2-h-4}$*), obtained from the Problem 3 instance.*

`KEY QUERY`*: when the* ι*-th key query is issued for a pattern* $\boldsymbol{P}$*,* $\mathcal{B}$ *answers as follows:*

- *When* $1 \leq \iota \leq h-1$*,* $\mathcal{B}$ *answers keys of temporal 2 form, that are computed using* $\mathbb{D}^*$ *of the Problem 3 instance.*
- *When* $\iota = h$*,* $\mathcal{B}$ *calculates* $sk_{\boldsymbol{P}}$ *using* $(\left\{\boldsymbol{h}^*_{\beta,i}\right\}_{i\in[L]}, \left\{\boldsymbol{d}^*_i\right\}_{i=0,3L+1,\cdots,4L})$ *of the Problem 3 instance as follows:* $\{\sigma_i, \xi_i\}_{i\in[L]} \leftarrow \mathbb{Z}_p, \boldsymbol{\eta} = (\eta_1, \cdots, \eta_L) \leftarrow \mathbb{Z}_p^L$

$$sk_{\boldsymbol{P}} = g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i\in\mathcal{I}} \sigma_i \boldsymbol{d}_i^* + \sum_{i\in[L]} \eta_i \boldsymbol{d}_{3L+i}^*} \cdot \prod_{i\in\mathcal{I}} \boldsymbol{h}_{\beta,i}^{*\xi_i}$$

- *When* $\iota \geq h+1$*,* $\mathcal{B}$ *answers normal keys using* $\mathbb{D}^*$ *of the Problem 3 instance.*

CHALLENGE: when $\mathcal{B}$ receives an encryption query with the challenge plaintext m and patterns $\boldsymbol{P}^0, \boldsymbol{P}^1$ from $\mathcal{A}$, $\mathcal{B}$ computes the challenge ciphertext $(c_1, \boldsymbol{c}_2)$ such that

$$\begin{aligned} \boldsymbol{c}_2 &= g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1} + s_3 \sum\limits_{i \in \bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i} \cdot \prod_{i \in \bar{W}(\boldsymbol{P}^0)} \boldsymbol{e}_i \cdot \prod_{i \in \bar{W}(\boldsymbol{P}^1)} \boldsymbol{f}_i \\ c_1 &= m \cdot (e(g_1, g_2)^{\alpha \boldsymbol{d}_0 \cdot \boldsymbol{d}_0^*})^{s_1} \end{aligned}$$

where $s_1, s_2, s_3 \leftarrow \mathbb{Z}_p, b \leftarrow \{0,1\}$ and $(\{\boldsymbol{d}_i\}_{i=0,2L+1,\cdots,3L,4L+1}, \{\boldsymbol{e}_i\}_{i\in[L]}, \{\boldsymbol{e}_i, \boldsymbol{f}_i\}_{i\in[L]})$ is a part of the Problem 3 instance.

GUESS: $\mathcal{A}$ finally outputs bit b'. If $b = b'$, $\mathcal{B}$ outputs $\beta' = 1$. Otherwise, $\mathcal{B}_{2-2}$ outputs $\beta' = 0$.

Let us see that the distribution of the view of adversary $\mathcal{A}$ in the above-mentioned game, simulated by $\mathcal{B}$ given a Problem 3 instance with $\beta \in \{0,1\}$, is the same as that in Game$_{2-h-3}$ (resp. Game$_{2-h-4}$) if $\beta = 0$ (resp. $\beta = 1$).

We consider the joint distribution of $\boldsymbol{c}_2$ and $sk_{\boldsymbol{P}}$. Ciphertext element $\boldsymbol{c}_2$ is

$$g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1} + s_3 \sum\limits_{i \in \bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i} \cdot \prod\nolimits_{i \in \bar{W}(\boldsymbol{P}^0)} \boldsymbol{e}_i \cdot \prod\nolimits_{i \in \bar{W}(\boldsymbol{P}^1)} \boldsymbol{f}_i$$

$$= g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1} + s_3 \sum\limits_{i \in \bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i + \sum\limits_{i \in \bar{W}(\boldsymbol{P}^0)} (\omega' b_{L+i} + \omega'' b_{2L+i})} \cdot g_1^{\sum\limits_{i \in \bar{W}(\boldsymbol{P}^1)} (\kappa' b_{L+i} + \kappa'' b_{2L+i})}$$

$$= g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1} + s_3 \sum\limits_{i \in \bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i + \sum\limits_{i \in \bar{W}(\boldsymbol{P}^0)} \omega' b_{L+i} + \sum\limits_{i \in \bar{W}(\boldsymbol{P}^1)} \kappa' b_{L+i}} \cdot g_1^{\sum\limits_{i \in \bar{W}(\boldsymbol{P}^0)} \omega'' b_{2L+i} + \sum\limits_{i \in \bar{W}(\boldsymbol{P}^1)} \kappa'' b_{2L+i}}$$

where $s_1, s_2, s_3, \omega', \omega'', \kappa', \kappa'' \in \mathbb{Z}_p$ are uniformly distributed.

Now let us see the value of $sk_{\boldsymbol{P}}$. When $\beta = 0$, $sk_{\boldsymbol{P}}$ (in case $\iota = h$) is

$$g_2^{\alpha \boldsymbol{d}_0^* + \sum\limits_{i \in \mathcal{I}} \sigma_i \boldsymbol{d}_i^* + \sum\limits_{i \in [L]} \eta_i \boldsymbol{d}_{3L+i}^*} \cdot \prod\nolimits_{i \in \mathcal{I}} \boldsymbol{h}_{\beta,i}^{*\xi_i}$$

$$= g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i\in\mathcal{I}} \sigma_i \boldsymbol{d}_i^* + \sum_{i\in[L]} \eta_i \boldsymbol{d}_{3L+i}^* + \sum_{i\in\mathcal{I}} (\tau\xi_i \boldsymbol{d}_{L+i}^* + \xi_i\delta_0 \boldsymbol{d}_{3L+i}^*)}$$

$$= g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i\in\mathcal{I}} \sigma_i \boldsymbol{d}_i^* + \sum_{i\in\mathcal{I}} \tau\xi_i \boldsymbol{d}_{L+i}^* + \sum_{i\in\mathcal{I}} \delta_0\xi_i\eta_i \boldsymbol{d}_{3L+i}^* + \sum_{i\in\mathcal{O}} \eta_i \boldsymbol{d}_{3L+i}^*}$$

where $\alpha, \sigma, \tau, \delta_0, \{\eta_i\}_{i\in[L]}$ *are uniformly and independently distributed. Therefore, generated* $\boldsymbol{c}_2, \mathsf{sk}_{\boldsymbol{P}}$ *have the same joint distribution as in* Game_{2-h-3}.
When $\beta = 1$, $\mathsf{sk}_{\boldsymbol{P}}$ *(in case* $\iota = h$*) is*

$$g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i\in\mathcal{I}} \sigma_i \boldsymbol{d}_i^* + \sum_{i\in[L]} \eta_i \boldsymbol{d}_{3L+i}^*} \cdot \prod_{i\in\mathcal{I}} \boldsymbol{h}_{\beta,i}^{*\xi_i}$$

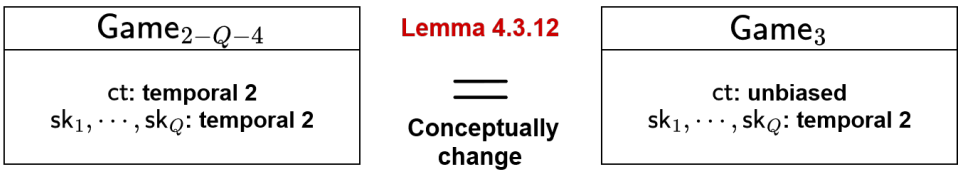
$$= g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i\in\mathcal{I}} \sigma_i \boldsymbol{d}_i^* + \sum_{i\in[L]} \eta_i \boldsymbol{d}_{3L+i}^* + \sum_{i\in\mathcal{I}} (\tau\xi_i \boldsymbol{d}_{2L+i}^* + \xi_i\delta_0 \boldsymbol{d}_{3L+i}^*)}$$

$$= g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i\in\mathcal{I}} \sigma_i \boldsymbol{d}_i^* + \sum_{i\in\mathcal{I}} \tau\xi_i \boldsymbol{d}_{2L+i}^* + \sum_{i\in\mathcal{I}} \xi_i\delta_0\eta_i \boldsymbol{d}_{3L+i}^* + \sum_{i\in\mathcal{O}} \eta_i \boldsymbol{d}_{3L+i}^*}$$

where $\alpha, \sigma, \tau, \delta_0, \{\eta_i\}_{i\in[L]}$ *are uniformly and independently distributed. Therefore, generated* $\boldsymbol{c}_2, \mathsf{sk}_{\boldsymbol{P}}$ *have the same joint distribution as in* Game_{2-h-4}.

Thus, $\left|\mathsf{Adv}_{\mathcal{A}}^{2-h-3}(\lambda) - \mathcal{A}^{2-h-4}(\lambda)\right| \leq \mathsf{Adv}_{\mathcal{B}}^{P3}(\lambda)$. □

Step 6



Lemma 4.3.12 *For any adversary* $\mathcal{A}$, $\left|\mathsf{Adv}_{\mathcal{A}}^{2-Q-4}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{3}(\lambda)\right| \leq \frac{2}{p^{|\mathcal{I}|}} + 3/p$.

Proof 4.3.12 *To prove this lemma, we will show that distribution* $(pk, \{\mathsf{sk}_{\boldsymbol{P}^j}\}_{j\in[Q]}, c_1, \boldsymbol{c}_2)$ *in* Game_{2-Q-4} *and that in* Game_3 *are equivalent. For that purpose, we define new dual orthonormal bases* $(\mathbb{B}, \mathbb{B}^*)$ *as follows:*

We generate $\theta_i \leftarrow \mathbb{Z}_p$ *for* $i \in [L]$ *and set for* $i \in [L]$, $\boldsymbol{b}_{2L+i} = \boldsymbol{d}_{2L+i} - \theta_i \boldsymbol{d}_i$, $\boldsymbol{b}_i^* = \boldsymbol{d}_i^* + \theta_i \boldsymbol{d}_{2L+i}^*$ *and*

$$\mathbb{B} = (\boldsymbol{d}_0, \cdots, \boldsymbol{d}_{2L}, \boldsymbol{b}_{2L+1}, \cdots, \boldsymbol{b}_{3L}, \boldsymbol{b}_{3L+1}, \cdots, \boldsymbol{b}_{4L+1}),$$
$$\mathbb{B}^* = (\boldsymbol{d}_0^*, \boldsymbol{b}_1^*, \cdots, \boldsymbol{b}_L^*, \boldsymbol{d}_{L+1}^*, \cdots, \boldsymbol{d}_{4L+1}^*)$$

We then easily verify that $\mathbb{B}$ *and* $\mathbb{B}^*$ *are dual orthonormal, and are distributed the same as the original bases* $(\mathbb{D}, \mathbb{D}^*)$*.*

Keys and challenge ciphertext $\{sk_{\boldsymbol{P}^j}\}_{j\in[Q]}, c_1, \boldsymbol{c}_2$ *in Game*$_{2-Q-4}$ *are expressed over bases* $(\mathbb{D}, \mathbb{D}^*)$ *and* $(\mathbb{B}, \mathbb{B}^*)$ *as*

$$\begin{aligned} sk_{\boldsymbol{P}^j} &= g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i\in\mathcal{I}} r_i^j \boldsymbol{d}_i^* + \sum_{i\in\mathcal{I}} x_i^j \boldsymbol{d}_{2L+i}^* + \sum_{l=1}^{L} \eta_l^j \cdot \boldsymbol{d}_{3L+l}^*} \\ &= g_2^{\alpha \boldsymbol{b}_0^* + \sum_{i\in\mathcal{I}} r_i^j (\boldsymbol{b}_i^* - \theta_i \boldsymbol{d}_{2L+i}^*) + \sum_{i\in\mathcal{I}} x_i^j \boldsymbol{b}_{2L+i}^* + \sum_{l=1}^{L} \eta_l^j \cdot \boldsymbol{b}_{3L+l}^*} \\ &= g_2^{\alpha \boldsymbol{b}_0^* + \sum_{i\in\mathcal{I}} r_i^j \boldsymbol{b}_i^* + \sum_{j\in\mathcal{I}} (x_i^j - r_i^j \theta_i) \boldsymbol{b}_{2L+i}^* + \sum_{l=1}^{L} \eta_l^j \cdot \boldsymbol{b}_{3L+l}^*} \end{aligned}$$

$$c_1 = \boldsymbol{m} \cdot e(g_1, g_2)^{\alpha \boldsymbol{d}_0 \boldsymbol{d}_0^* s} = \boldsymbol{m} \cdot e(g_1, g_2)^{\alpha \boldsymbol{b}_0 \boldsymbol{b}_0^* s}$$

$$\begin{aligned} \boldsymbol{c}_2 &= g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1} + s_3 \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{d}_i + t \sum_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{L+i} + \tilde{t} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{L+i} + u \sum_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{d}_{2L+i} + \tilde{u} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{d}_{2L+i}} \\ &= g_1^{s_1 \boldsymbol{b}_0 + s_2 \boldsymbol{b}_{4L+1} + s_3 \sum_{i\in\bar{W}(\boldsymbol{P}^b)} \boldsymbol{b}_i + t \sum_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{b}_{L+i} + \tilde{t} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{b}_{L+i}} \\ &\quad \cdot g_1^{u \sum_{i\in\bar{W}(\boldsymbol{P}^0)} (\boldsymbol{b}_{2L+i} + \theta_i \boldsymbol{b}_i) + \tilde{u} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} (\boldsymbol{b}_{2L+i} + \theta_i \boldsymbol{b}_i)} \\ &= g_1^{s_1 \boldsymbol{b}_0 + s_2 \boldsymbol{b}_{4L+1} + \sum_{i=1}^{L} \nu_i \boldsymbol{b}_i + t \sum_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{b}_{L+i} + \tilde{t} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{b}_{L+i} + u \sum_{i\in\bar{W}(\boldsymbol{P}^0)} \boldsymbol{b}_{2L+i} + \tilde{u} \sum_{i\in\bar{W}(\boldsymbol{P}^1)} \boldsymbol{b}_{2L+i}} \end{aligned}$$

where for $i \in [L]$*:*

$$\tilde{x}_i = \begin{cases} x_i^j - r_i^j \theta_i & \text{if } i \in \mathcal{I} \\ 0 & \text{otherwise} \end{cases}$$

and

$$\nu_i = \begin{cases} 0 & \text{if } i \notin \bar{W}(\boldsymbol{P}^0) \wedge i \notin \bar{W}(\boldsymbol{P}^1) \\ \theta_i u & \text{if } i \in \bar{W}(\boldsymbol{P}^0) \wedge i \notin \bar{W}(\boldsymbol{P}^1) \wedge b = 1 \\ \theta_i \tilde{u} & \text{if } i \in \bar{W}(\boldsymbol{P}^1) \wedge i \notin \bar{W}(\boldsymbol{P}^0) \wedge b = 0 \\ s_3 + u\theta_i & \text{if } i \in \bar{W}(\boldsymbol{P}^0) \wedge i \notin \bar{W}(\boldsymbol{P}^1) \wedge b = 0 \\ s_3 + \tilde{u}\theta_i & \text{if } i \in \bar{W}(\boldsymbol{P}^1) \wedge i \notin \bar{W}(\boldsymbol{P}^0) \wedge b = 1 \\ s_3 + (u+\tilde{u})\theta_i & \text{if } i \in \bar{W}(\boldsymbol{P}^1) \wedge i \in \bar{W}(\boldsymbol{P}^0) \end{cases}$$

are uniformly, independently (from other variables) distributed since $s_3, \theta, t^j \leftarrow \mathbb{Z}_p$*, except with probability* $\frac{2}{p^{|\mathcal{I}|}} + 3/p$*.*

In the light of the adversary's view, both $(\mathbb{D}, \mathbb{D}^*)$ *and* $(\mathbb{B}, \mathbb{B}^*)$ *are consistent with public key pk. Therefore,* $\{sk_{\boldsymbol{P}^j}\}_{j\in[Q]}, c_1, \boldsymbol{c}_2$ *can be expressed as keys and ciphertext in two ways, in Game*$_{2-Q-4}$ *over bases* $(\mathbb{D}, \mathbb{D}^*)$ *and in Game*$_3$ *over bases* $(\mathbb{B}, \mathbb{B}^*)$*.*

Thus, Game$_{2-Q-4}$ *can be conceptually changed to Game*$_3$*.* □

Combining all theses proofs, we obtain that any adversary has no advantage in winning the security game. Adding to these the fact that Problem 1, Problem 2 and Problem 3 hold if XDLin1, XDLin2 hold, we have proven Lemma 4.3.6.

IND-WIBE-CPA **security.** We finally prove that our WIBE scheme satisfies indistinguishability, as stated in the following lemma.

Lemma 4.3.13 *If XDLin*1*, XDLin*2 *hold, then our scheme satisfies adaptive indistinguishability.*

Our proof is similar to the one of [91] (Section 3.5.2). It exploits the dual system encryption through a sequence of $Q+3$ games, where $Q \in \mathbb{N}$ is the number of secret keys an adversary is allowed to query.

- Game$_0$ is the original game given in the WIBE security in Figure 4.1.
- Game$_1$: is the same as Game$_0$ except that the challenge ciphertext $(c_1, \boldsymbol{c}_2)$ for challenge plaintexts $(\mathsf{m}_0, \mathsf{m}_1)$ and challenge pattern $\boldsymbol{P}^*$ is changed into semi-functional form: $s_1, s_2, s_3, t_1, \cdots, t_L \leftarrow \mathbb{Z}_p, b \leftarrow \{0,1\}$, and requires that $P_1^* \neq \star$,

$$c_1 = \mathsf{m}_b \cdot e(g_1, g_2)^{\alpha \boldsymbol{d}_0 \cdot \boldsymbol{d}_0^* s_1}, \qquad \boldsymbol{c}_2 = g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1} + s_3 \sum\limits_{i \in \bar{W}(\boldsymbol{P}^*)} \boldsymbol{d}_i + \sum\limits_{l=1}^{L} t_l \boldsymbol{d}_{L+l}} \tag{4.7}$$

- Game_{2-k} ($k \in [Q]$): is the same as $\text{Game}_{2-(k-1)}$ (for $k = 1$, $\text{Game}_{2-(k-1)}$ is Game_1) except that the reply to the k-th key queried for $\boldsymbol{P}$ is changed into semi-functional form: $\alpha, \{r_j\}_{j\in\mathcal{I}}, \{\eta_i, x_i\}_{i\in[L]} \leftarrow \mathbb{Z}_p$,

$$\boldsymbol{sk}_{\boldsymbol{P}} = g_2^{\alpha \boldsymbol{d}_0^* + \sum_{j\in\mathcal{I}} r_j \boldsymbol{d}_j^* + \sum_{l=1}^{L} x_l \boldsymbol{d}_{L+l}^* + \sum_{l=1}^{L} \eta_l \boldsymbol{d}_{3L+l}^*} \tag{4.8}$$

- Game_3: is the same as Game_{2-Q} except that the semi-functional challenge ciphertext $(c_1, \boldsymbol{c}_2)$ for challenge plaintexts $(\mathsf{m}_0, \mathsf{m}_1)$ and challenge pattern $\boldsymbol{P}^*$ is changed into a randomized form: $s_1', \{\tilde{s}_i\}_{i\in[L]} \leftarrow \mathbb{Z}_p$

$$c_1 = \mathsf{m}_b \cdot e(g_1, g_2)^{\alpha \boldsymbol{d}_0 \cdot \boldsymbol{d}_0^* s_1}, \qquad \boldsymbol{c}_2 = g_1^{s_1' \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1} + \sum_{i=1}^{L} \tilde{s}_i \boldsymbol{d}_i + \sum_{l=1}^{L} t_l \boldsymbol{d}_{L+l}} \tag{4.9}$$

  and all the other variables are generated as in Game_{2-Q}.

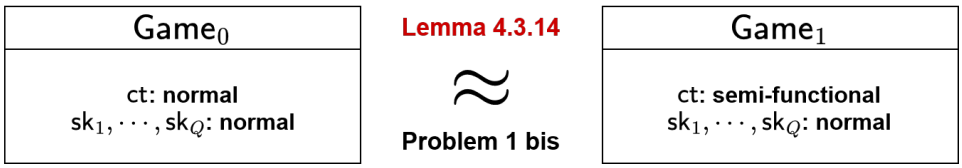
Informally, the proof works as explained in Figure 4.13.

- **Step 1**: we prove that if an adversary can distinguish Game_0 from Game_1 then an adversary against Problem 1 bis (Definition 2.3.8) can be created. This is formalized by Lemma 4.3.14.
- **Step 2**: we build an adversary against Problem 2 bis (Definition 2.3.9) using an adversary that distinguishes $\text{Game}_{2-(k-1)}$ from Game_{2-k}, as stated in Lemma 4.3.16.
- **Step 3**: we prove that the advantage of an adversary in winning Game_{2-Q} is the same than the one of an adversary winning Game_3; and the latter is equal to 0. This is formalized by Lemma 4.3.17 and Lemma 4.3.18.

Figure 4.13: Informal indistinguishability security proof for our pattern-hiding identity-based encryption with wildcards scheme.

The original proofs are made in the symmetric pairing settings but they can easily be made in the asymmetric setting by taking elements in the correct group.

Step 1



Lemma 4.3.14 *For any adversary $\mathcal{A}$, there exists a probabilistic adversary $\mathcal{B}$, whose running time is essentially the same as that of $\mathcal{A}$, such that for any security parameter λ, $\left|\mathsf{Adv}_{\mathcal{A}}^{(0)}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(1)}(\lambda)\right| = \mathsf{Adv}_{\mathcal{B}}^{P1^b}(\lambda)$.*

Proof 4.3.13 *In order to prove lemma 4.3.14, we construct a probabilistic adversary $\mathcal{B}$ against Problem 1 bis by using any adversary $\mathcal{A}$ in a security game (Game$_0$ or Game$_1$) as a black box as follows.*

INIT: $\mathcal{B}$ is given a Problem 1 bis instance $(\Gamma, \mathbb{D}, \hat{\mathbb{D}}^*, \boldsymbol{e}_{\beta,1}, \{\boldsymbol{e}_i\ \}_{i\in[n]})$.

SETUP: $\mathcal{B}$ plays a role of the challenger in the security game against adversary $\mathcal{A}$. She returns pk $= (\Gamma, e(g_1,g_2)^{\alpha \boldsymbol{d}_0.\boldsymbol{d}_0^*}, g_1^{\boldsymbol{d}_0}, g_1^{\boldsymbol{d}_{4L+1}}, \boldsymbol{h}_1 = g_1^{\boldsymbol{d}_1}, \cdots, \boldsymbol{h}_L = g_1^{\boldsymbol{d}_L})$ to $\mathcal{A}$.

KEY QUERY: when a key queried is issued, $\mathcal{B}$ answers a correct secret key computed by using $\hat{\mathbb{D}}^*$, i.e. a normal key.

CHALLENGE: when $\mathcal{B}$ gets challenge plaintexts m_0, m_1 and pattern $\boldsymbol{P}^*$ (with $\boldsymbol{P}_1^* \neq \star$) from $\mathcal{A}$, $\mathcal{B}$ returns $(c_1, \boldsymbol{c}_2)$ such that $c_1 = m_b \cdot e(g_1,g_2)^{\alpha \boldsymbol{d}_0 \cdot \boldsymbol{d}_0^* s_1}$ and $\boldsymbol{c}_2 = g_1^{s_1 \boldsymbol{d}_0} \cdot \boldsymbol{e}_{\beta,1} \cdot \prod_{i\in\bar{W}(\boldsymbol{P}^*), i\geq 2} \boldsymbol{e}_i$, where $\boldsymbol{e}_{\beta,1}$ and $\boldsymbol{e}_i$ are from the Problem 1 bis instance, $s_1 \leftarrow \mathbb{Z}_p$ and $b \in \{0,1\}$.

GUESS: $\mathcal{A}$ outputs a bit b'. If $b = b'$, $\mathcal{B}$ outputs $\beta' = 1$. Otherwise, $\mathcal{B}$ outputs $\beta' = 0$.

Let us see that if $\beta = 0$, then the distribution of $(c_1, \boldsymbol{c}_2)$ is the same as that in Game$_0$. If $\beta = 1$, the distribution of $(c_1, \boldsymbol{c}_2)$ is the same as that in Game$_1$. If $\beta = 0$,

$$\begin{aligned}
\boldsymbol{c}_2 &= g_1^{s_1 \boldsymbol{d}_0} \cdot \boldsymbol{e}_{\beta,1} \cdot \prod_{i\in\bar{W}(\boldsymbol{P}^*), i\geq 2} \boldsymbol{e}_i \\
&= g_1^{s_1 \boldsymbol{d}_0} \cdot g_1^{\omega \boldsymbol{d}_1 + \gamma \boldsymbol{d}_{4L+1}} \cdot \prod_{i\in\bar{W}(\boldsymbol{P}^*), i\geq 2} g_1^{\omega b_i} \\
&= g_1^{s_1 \boldsymbol{d}_0 + \omega \sum_{i\in\bar{W}(\boldsymbol{P}^*)} \boldsymbol{d}_i + \gamma \boldsymbol{d}_{4L+1}}.
\end{aligned}$$

This is the challenge ciphertext in Game$_0$. If $\beta = 1$,

$$\boldsymbol{c}_2 = g_1^{s_1 \boldsymbol{d}_0} \cdot \boldsymbol{e}_{\beta,1} \cdot \prod_{i\in\bar{W}(\boldsymbol{P}^*), i\geq 2} \boldsymbol{e}_i$$

$$= g_1^{s_1 \boldsymbol{d}_0} \cdot g_1^{\omega \boldsymbol{d}_1 + \sum_{l=1}^{L} z_l \boldsymbol{d}_{L+l} + \gamma \boldsymbol{d}_{4L+1}} \cdot \prod_{i \in \bar{W}(\boldsymbol{P}^*), i \geq 2} g_1^{\omega b_i}$$

$$= g_1^{s_1 \boldsymbol{d}_0 + \omega \sum_{i \in \bar{W}(\boldsymbol{P}^*)} \boldsymbol{d}_i + \sum_{l=1}^{L} z_l \boldsymbol{d}_{L+l} + \gamma \boldsymbol{d}_{4L+1}}.$$

Because $(z_1, \cdots, z_L) \leftarrow \mathbb{Z}_p^L \backslash \{\mathbf{0}^L\}$ *and* γ *are independently uniform, this is the challenge ciphertext in Game*$_1$.

When $\beta = 0$, *the advantage of* $\mathcal{A}$ *in the above game is equal to that in Game*$_0$, *i.e.,* $\mathsf{Adv}_{\mathcal{A}}^{(0)}(\lambda)$, *and is also equal to* $\Pr_0 = \Pr[\mathcal{B}(\Delta, \boldsymbol{t}) \to 0 | \beta = 0]$. *Similarly, when* $\beta = 1$, *we see that the advantage of* $\mathcal{A}$ *in the above game is equal to* $\mathsf{Adv}_{\mathcal{A}}^{(1)}(\lambda)$, *and is also equal to* $\Pr_1 = \Pr[\mathcal{B}(\Delta, \boldsymbol{t}) \to 1 | \beta = 1]$. *Therefore,* $\left|\mathsf{Adv}_{\mathcal{A}}^{(0)}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(1)}(\lambda)\right| = |\Pr_0 - \Pr_1| = \mathsf{Adv}_{\mathcal{B}}^{P1b}(\lambda)$. *This completes the proof.* □

To prove lemma 4.3.16, we need the following lemma from [91], that we admit.

Lemma 4.3.15 *[91] Let* $C = \{(\boldsymbol{x}, \boldsymbol{v}) | \boldsymbol{x} \cdot \boldsymbol{v} \neq 0\} \subset V \times V^*$, *where* V *is n-dimensional vector space* $\mathbb{Z}_p^n$, *and* V^* *its dual. For all* $(\boldsymbol{x}, \boldsymbol{v}) \in C$, *for all* $(\boldsymbol{r}, \boldsymbol{w}) \in C$,

$$\Pr[\boldsymbol{x}(\rho \boldsymbol{U}) = \boldsymbol{r} \wedge \boldsymbol{v}(\tau \boldsymbol{Z}) = \boldsymbol{w}] = \frac{1}{s},$$

where $\boldsymbol{Z} \leftarrow GL(n, \mathbb{Z}_p)$, $\rho, \tau \leftarrow \mathbb{Z}_p^*$, $\boldsymbol{U} = (\boldsymbol{Z}^{-1})^\top$ *and* $s = \#C (= (p^n - 1)(p^n - p^{n-1}))$.

Step 2

$k = 1, \cdots, Q$:

$\mathsf{Game}_{2-(k-1)}$	Lemma 4.3.16 ≈ Problem 2 bis	Game_{2-k}
ct: semi-functional $\mathsf{sk}_1, \cdots, \mathsf{sk}_{k-1}$: semi-functional $\mathsf{sk}_k, \cdots, \mathsf{sk}_Q$: normal		ct: semi-functional $\mathsf{sk}_1, \cdots, \mathsf{sk}_k$: semi-functional $\mathsf{sk}_{k+1}, \cdots, \mathsf{sk}_Q$: normal

Lemma 4.3.16 *For any adversary* $\mathcal{A}$, *there exists a probabilistic adversary* $\mathcal{B}$, *whose running time is essentially the same as that of* $\mathcal{A}$, *such that for any security parameter* λ, $\left|\mathsf{Adv}_{\mathcal{A}}^{(2-(k-1))}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(2-k)}(\lambda)\right| \leq \mathsf{Adv}_{\mathcal{B}}^{P2^b}(\lambda) + 1/p$.

Proof 4.3.14 *In order to prove lemma 4.3.16, we construct a probabilistic adversary* $\mathcal{B}$ *against Problem 2 bis by using any adversary* $\mathcal{A}$ *in a security game (Game*$_{2-(k-1)}$ *or*

Game$_{2-k}$) as a black box as follows.

INIT: $\mathcal{B}$ is given a Problem 2 bis instance $(\Gamma, \hat{\mathbb{D}}, \mathbb{D}^*, \{\boldsymbol{h}^*_{\beta,i}, \boldsymbol{e}_i\}_{i\in[n]})$.

SETUP: $\mathcal{B}$ plays a role of the challenger in the security game against adversary $\mathcal{A}$. She returns pk $= (\Gamma, e(g_1,g_2)^{\alpha \boldsymbol{d}_0 . \boldsymbol{d}^*_0}, g_1^{\boldsymbol{d}_0}, g_1^{\boldsymbol{d}_{4L+1}}, \boldsymbol{h}_1 = g_1^{\boldsymbol{d}_1}, \cdots, \boldsymbol{h}_L = g_1^{\boldsymbol{d}_L})$ to $\mathcal{A}$.

KEY QUERY: when the s-th key query is issued for predicate $\boldsymbol{P}$, $\mathcal{B}$ answers as follows:

- *When $1 \leq s \leq k-1$, $\mathcal{B}$ calculates and answers by using $\hat{\mathbb{D}}^*$*

$$\mathrm{sk}_{\boldsymbol{P}} = g_2^{\alpha \boldsymbol{d}^*_0 + \sum\limits_{j\in\mathcal{I}} r_j \boldsymbol{d}^*_j + \sum\limits_{l=1}^{L} x_l \boldsymbol{d}^*_{L+l} + \sum\limits_{l=1}^{L} \eta_l \boldsymbol{d}^*_{3L+l}}.$$

- *When $s = k$, $\mathcal{B}$ calculates and answers $\mathrm{sk}_{\boldsymbol{P}}$ as follows:$\{\xi_i \leftarrow \mathbb{Z}_p\}_{i\in\mathcal{I}}$,*

$$\mathrm{sk}_{\boldsymbol{P}} = g_2^{\alpha \boldsymbol{d}^*_0} \cdot \prod_{i\in\mathcal{I}} \boldsymbol{h}^{*\xi_i}_{\beta,i},$$

- *When $q \geq k+1$, $\mathcal{B}$ answers a correct secret key computed by using $\mathbb{D}^*$, i.e. normal key.*

CHALLENGE: when $\mathcal{B}$ gets challenge plaintexts m_0, m_1 and pattern $\boldsymbol{P}^*$ from $\mathcal{A}$, $\mathcal{B}$ calculates and returns $(c_1, \boldsymbol{c}_2)$ such that $c_1 = m_b \cdot e(g_1,g_2)^{\alpha \boldsymbol{d}_0 \cdot \boldsymbol{d}^*_0 s_1}$ and $\boldsymbol{c}_2 = g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1}} \cdot \prod\limits_{i\in\bar{W}(\boldsymbol{P}^*)} \boldsymbol{e}_i$, where $\boldsymbol{e}_i$ are from the Problem 2 bis instance, $s_1, s_2 \leftarrow \mathbb{Z}_p$ and $b \in \{0,1\}$.

GUESS: $\mathcal{A}$ outputs a bit b'. If $b = b'$, $\mathcal{B}$ outputs $\beta' = 1$. Otherwise, $\mathcal{B}$ outputs $\beta' = 0$.

Let us see that if $\beta = 0$, then the distribution of challenge $(c_1, \boldsymbol{c}_2)$ and $\mathrm{sk}_{\boldsymbol{P}}$ is the same as that in Game$_{2-(k-1)}$ except with probability $1/p$. If $\beta = 1$, the distribution of challenge $(c_1, \boldsymbol{c}_2)$ and $\boldsymbol{sk}_{\boldsymbol{P}}$ is the same as that in Game$_{2-k}$ except with probability $1/p$.

We consider the joint distribution of $\boldsymbol{c}_2$ and $\mathrm{sk}_{\boldsymbol{P}}$. Ciphertext element $\boldsymbol{c}_2$ is

$$\boldsymbol{c}_2 = g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1}} \cdot \prod_{i\in\bar{W}(\boldsymbol{P}^*)} \boldsymbol{e}_i = g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1}} \cdot \prod_{i\in\bar{W}(\boldsymbol{P}^*)} g_1^{\omega \boldsymbol{d}_i + \tau \sum\limits_{j=1}^{L} z_{i,j} \boldsymbol{d}_{L+j}}$$

$$= g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1} + \omega \sum\limits_{i\in\bar{W}(\boldsymbol{P}^*)} \boldsymbol{d}_i + \tau \sum\limits_{j=1}^{L} \sum\limits_{i\in\bar{W}(\boldsymbol{P}^*)} z_{i,j} \boldsymbol{d}_{L+j}}$$

$$= g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1} + \omega \sum_{i \in \bar{W}(\boldsymbol{P}^*)} \boldsymbol{d}_i + \sum_{j=1}^{L} \tilde{t}_j \boldsymbol{d}_{L+j}}$$

where $s_1, s_2, \omega \in \mathbb{Z}_p$, $\tilde{t}_j = \sum_{i \in \bar{W}(\boldsymbol{P}^*)} \tau z_{i,j}$ *and* $(\tilde{t}_1, \cdots, \tilde{t}_L) \leftarrow \mathbb{Z}_p^L \setminus \{\mathbf{0}\}$ *are independently uniform.*

If $\beta = 0$*, secret key (generated in case* $s = k$*) is*

$$\boldsymbol{sk}_{\boldsymbol{P}} = g_2^{\alpha \boldsymbol{d}_0^*} \cdot \prod_{i \in \mathcal{I}} \boldsymbol{h}_{\beta,i}^{*\xi_i} = g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i \in \mathcal{I}} \xi_i \delta \boldsymbol{d}_i^* + \sum_{i \in \mathcal{I}} \sum_{j=1}^{L} \xi_i \delta_{i,j} \boldsymbol{d}_{3L+j}^*}$$

This is a normal secret key, thus distribution of $(c_1, \boldsymbol{c}_2)$*,* $\boldsymbol{sk}_{\boldsymbol{P}}$ *are as in* $\mathsf{Game}_{2-(k-1)}$ *(i.e. temporal ciphertext and normal key). If* $\beta = 1$*,*

$$\begin{aligned} \boldsymbol{sk}_{\boldsymbol{P}} &= g_2^{\alpha \boldsymbol{d}_0^*} \cdot \prod_{i \in \mathcal{I}} \boldsymbol{h}_{\beta,i}^{*\xi_i} = g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i \in \mathcal{I}} \xi_i \delta \boldsymbol{d}_i^* + \sum_{i \in \mathcal{I}} \sum_{j=1}^{L} \xi_i u_{i,j} \boldsymbol{d}_{L+j}^* + \sum_{i \in \mathcal{I}} \sum_{j=1}^{L} \xi_i \delta_{i,j} \boldsymbol{d}_{3L+j}^*} \\ &= g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i \in \mathcal{I}} \xi_i \delta \boldsymbol{d}_i^* + \sum_{j=1}^{L} \tilde{x}_j \boldsymbol{d}_{L+j}^* + \sum_{i \in \mathcal{I}} \sum_{j=1}^{L} \xi_i \delta_{i,j} \boldsymbol{d}_{3L+j}^*} \end{aligned}$$

where $\tilde{x}_j = \sum_{i \in \mathcal{I}} \xi_i u_{i,j}$ *and* $(\tilde{x}_1, \cdots, \tilde{x}_L) \leftarrow \mathbb{Z}_p^L \setminus \{\mathbf{0}\}$*.*

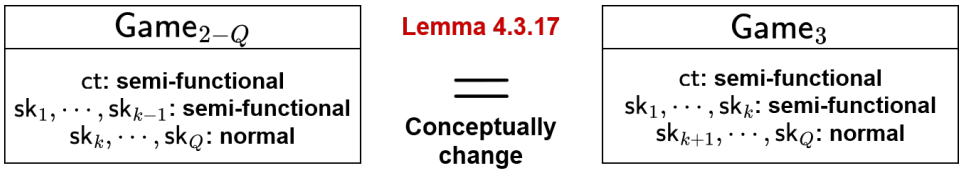
Since $\boldsymbol{Z} = (\boldsymbol{U}^{-1})^\top$ *where* $\boldsymbol{Z} = (z_{i,j})$ *and* $U = (u_{i,j})$*, we should verify the independence of coefficient vectors* $\tilde{\boldsymbol{t}} = (\tilde{t}_1, \cdots, \tilde{t}_l)$ *in* $\boldsymbol{c}_2$ *and* $\tilde{\boldsymbol{x}} = (\tilde{x}_1, \cdots, \tilde{x}_l)$ *in* $\boldsymbol{sk}_{\boldsymbol{P}}$*. Notice that we can rewrite* $\tilde{\boldsymbol{t}}$ *and* $\tilde{\boldsymbol{x}}$ *respectively as* $\overrightarrow{\boldsymbol{y}} \cdot \boldsymbol{U}$ *and* $\overrightarrow{\boldsymbol{x}} \cdot \boldsymbol{Z}$*, where* $\overrightarrow{\boldsymbol{y}}$ *and* $\overrightarrow{\boldsymbol{x}}$ *are vectors such that* $\overrightarrow{y}_i = \begin{cases} \xi_i & \text{if } i \in \mathcal{I} \\ 0 & \text{otherwise} \end{cases}$ *for* $i \in [L]$ *and* $\overrightarrow{x}_i = \begin{cases} \tau & \text{if } i \in \bar{W}(\boldsymbol{P}^*) \\ 0 & \text{otherwise} \end{cases}$ *for* $i \in [L]$*.*

Since $\mathcal{I} \cap \bar{W}(\boldsymbol{P}^*) \neq \varnothing$ *from condition on keys and challenge ciphertext, coefficients vectors* $\tilde{\boldsymbol{t}}$ *and* $\tilde{\boldsymbol{x}}$ *are (pairwise)-independently and uniformly distributed under the condition that* $\overrightarrow{\boldsymbol{y}} \cdot \overrightarrow{\boldsymbol{x}} \neq 0$ *(from lemma 4.3.15). Since* $(x_1, \cdots, x_l), (t_1, \cdots, t_l) \leftarrow \mathbb{Z}_p^L$ *in* Game_{2-k}*, the event that*$(x_, \cdots, x_l) \cdot (t_1, \cdots, t_l) = 0$ *occurs in the game with probability* $1/p$*.*
Thus this is a temporal 1 secret key, and the distribution of $(c_1, \boldsymbol{c}_2)$*,* $\boldsymbol{sk}_{\boldsymbol{P}}$ *are as in* Game_{2-k}*, except with probability* $1/p$*.*

When $\beta = 0$*, the advantage of* $\mathcal{A}$ *in the above game is equal to that in* $\mathsf{Game}_{2-(k-1)}$*, i.e.,* $\mathsf{Adv}_{\mathcal{A}}^{(2-(k-1))}(\lambda)$*, and is also equal to* $\mathrm{Pr}_0 = \Pr[\mathcal{B}(\Delta, \boldsymbol{t}) \to 0 | \beta = 0]$*. Similarly, when* $\beta = 1$*, we see that the advantage of* $\mathcal{A}$ *in the above game is equal to* $\mathsf{Adv}_{\mathcal{A}}^{(2-k)}(\lambda)$*, and is*

also equal to $\Pr_1 = \Pr\left[\mathcal{B}(\Delta, \boldsymbol{t}) \to 1 | \beta = 1\right]$. *Therefore,* $\left|\mathsf{Adv}_{\mathcal{A}}^{(2-(k-1))}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{(2-h)}(\lambda)\right| \leq |\Pr_0 - \Pr_1| + 1/p = \mathsf{Adv}_{\mathcal{B}}^{P2^b}(\lambda) + 1/p$. *This completes the proof.* □

Step 3



Lemma 4.3.17 *For any adversary* $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{(2-Q)}(\lambda) = \mathsf{Adv}_{\mathcal{A}}^{(3)}(\lambda)$.

Proof 4.3.15 *To prove lemma 4.3.17, we will show that distribution* $(pk, \left\{sk^{(j)}\right\}_{j\in[Q]}, c_1, \boldsymbol{c}_2)$ *in Game*$_{2-Q}$ *and that in Game*$_3$ *are equivalent. For that purpose, we define new bases* $\mathbb{B}, \mathbb{B}^*$ *as follows: we generate randoms* $\{\xi_{i,s}\}_{i,s\in[L]}$, $\{\theta_i\}_{i=1,\cdots L}$ *and set for* $i \in [L]$*:*

$$\boldsymbol{b}_{L+i} = \boldsymbol{d}_{L+i} - \sum_{s=1}^{L} \xi_{i,s}\boldsymbol{d}_s - \theta_i \boldsymbol{d}_0, \quad \boldsymbol{b}_i^* = \boldsymbol{d}_i^* + \sum_{s=1}^{L} \xi_{s,i}\boldsymbol{d}_{L+s}^*, \quad \boldsymbol{b}_0^* = \boldsymbol{d}_0^* + \sum_{s=1}^{L} \theta_s \boldsymbol{d}_{L+s}^*$$

and

$$\mathbb{B} = (\boldsymbol{d}_0, \boldsymbol{d}_1, \cdots, \boldsymbol{d}_L, \boldsymbol{b}_{L+1}, \cdots, \boldsymbol{b}_{2L}, \boldsymbol{d}_{2L+1}, \cdots, \boldsymbol{d}_{4L+1}),$$
$$\mathbb{B}^* = (\boldsymbol{b}_0^*, \boldsymbol{b}_1^*, \cdots, \boldsymbol{b}_L^*, \boldsymbol{d}_{L+1}^*, \cdots, \boldsymbol{d}_{2L}^*, \boldsymbol{d}_{2L+1}^*, \cdots, \boldsymbol{d}_{4L+1}^*).$$

We then easily verify that $\mathbb{B}$ *and* $\mathbb{B}^*$ *are dual orthonormal, and are distributed the same as the original bases,* $\mathbb{D}, \mathbb{D}^*$. *Keys and challenge ciphertext* $(\left\{sk^{(j)}\right\}_{j\in[Q]}, c_1, \boldsymbol{c}_2)$ *in Game*$_{2-Q}$ *are expressed over bases* $\mathbb{D}$ *and* $\mathbb{D}^*$ *as*

$$sk^{(j)} = g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i\in\mathcal{I}} r_i^{(j)} \boldsymbol{d}_i^* + \sum_{l=1}^{L} x_l^{(j)} \boldsymbol{d}_{L+l}^* + \sum_{l=1}^{L} \eta_l^{(j)} \boldsymbol{d}_{3L+l}^*},$$
$$c_1 = m_b \cdot e(g_1, g_2)^{\alpha \boldsymbol{d}_0 \cdot \boldsymbol{d}_0^* s_1},$$
$$\boldsymbol{c}_2 = g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1} + s_3 \sum_{i\in\overline{W}(\boldsymbol{P}^*)} \boldsymbol{d}_i + \sum_{l=1}^{L} t_l \boldsymbol{d}_{L+l}}$$

Then,

$$sk^{(j)} = g_2^{\alpha \boldsymbol{d}_0^* + \sum_{i\in\mathcal{I}} r_i^{(j)} \boldsymbol{d}_i^* + \sum_{l=1}^{L} x_l^{(j)} \boldsymbol{d}_{L+l}^* + \sum_{l=1}^{L} \eta_l^{(j)} \boldsymbol{d}_{3L+l}^*}$$

$$= g_2^{\alpha(\boldsymbol{b}_0^* - \sum_{s=1}^{L} \theta_s \boldsymbol{b}_{L+s}^*) + \sum_{i \in \mathcal{I}} r_i^{(j)} (\boldsymbol{b}_i^* - \sum_{s=1}^{L} \xi_{i,s} \boldsymbol{b}_{L+s}^*) + \sum_{l=1}^{L} x_l^{(j)} \boldsymbol{b}_{L+l}^* + \sum_{l=1}^{L} \eta_l^{(j)} \boldsymbol{b}_{3L+l}^*}$$

$$= g_2^{\alpha \boldsymbol{b}_0^* + \sum_{i \in \mathcal{I}} r_i^{(j)} \boldsymbol{b}_i^* + \sum_{l=1}^{L} \tilde{x}_l^{(j)} \boldsymbol{b}_{L+l}^* + \sum_{l=1}^{L} \eta_l^{(j)} \boldsymbol{b}_{3L+l}^*}$$

where $\tilde{x}_l^{(j)} = -\alpha\theta_l - \sum_{i \in \mathcal{I}} r_i^{(j)} \xi_{i,l} + x_l^{(j)}$ *for* $l \in [L]$*, which are uniformly, independently distributed since* $x_l^{(j)} \leftarrow \mathbb{Z}_p$*.*

$$\begin{aligned}
\boldsymbol{c}_2 &= g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4L+1} + s_3 \sum_{i \in \bar{W}(\boldsymbol{P}^*)} \boldsymbol{d}_i + \sum_{l=1}^{L} t_l \boldsymbol{d}_{L+l}} \\
&= g_1^{s_1 \boldsymbol{b}_0 + s_2 \boldsymbol{b}_{4L+1} + s_3 \sum_{i \in \bar{W}(\boldsymbol{P}^*)} \boldsymbol{b}_i + \sum_{l=1}^{L} t_l (\boldsymbol{b}_{L+l} + \sum_{s=1}^{L} \xi_{l,s} \boldsymbol{b}_s + \theta_l \boldsymbol{b}_0)} \\
&= g_1^{\boldsymbol{b}_0 + (s_1 + \sum_{l=1}^{L} t_l \theta_l) + s_2 \boldsymbol{b}_{4L+1} + s_3 \sum_{i \in \bar{W}(\boldsymbol{P}^*)} \boldsymbol{b}_i + \sum_{l=1}^{L} t_l \sum_{s=1}^{L} \xi_{l,s} \boldsymbol{b}_s + \sum_{l=1}^{L} t_l \boldsymbol{b}_{L+l}} \\
&= g_1^{s_1' \boldsymbol{b}_0 + s_2 \boldsymbol{b}_{4L+1} + \sum_{i=1}^{L} \tilde{s}_i \boldsymbol{b}_i + \sum_{l=1}^{L} t_l \boldsymbol{b}_{L+l}}
\end{aligned}$$

where $s_1' = s_1 + \sum_{l=1}^{L} t_l \theta_l$ *and* $\tilde{s}_i = \begin{cases} \sum_{l=1}^{L} t_l \xi_{l,i} & \text{if } i \notin \bar{W}(\boldsymbol{P}^*) \\ \sum_{l=1}^{L} t_l \xi_{l,i} + s_3 & \text{if } i \in \bar{W}(\boldsymbol{P}^*) \end{cases}$ *for* $k \in [L]$*.*

which are uniformly, independently distributed since $(t_1, \cdots, t_l) \leftarrow \mathbb{Z}_p^L \setminus \{\mathbf{0}\}$*,* $\{\xi_{t,i}\} \leftarrow \mathbb{Z}_p$*.*

In the light of the adversary's view, both $(\mathbb{D}, \mathbb{D}^*)$ *and* $(\mathbb{B}, \mathbb{B}^*)$ *are consistent with public key* $pk = (\Gamma, e(g_1, g_2)^{\alpha \boldsymbol{d}_0 . \boldsymbol{d}_0^*}, g_1^{\boldsymbol{d}_0}, g_1^{\boldsymbol{d}_{4L+1}}, \boldsymbol{h}_1 = g_1^{\boldsymbol{d}_1}, \cdots, \boldsymbol{h}_L = g_1^{\boldsymbol{d}_L})$*. Therefore,* $\left\{ sk^{(j)} \right\}_{j \in [Q]}$ *and* $\boldsymbol{c}_2$ *can be expressed as keys and ciphertext in two ways, in Game*$_{2-Q}$ *over bases* $(\mathbb{D}, \mathbb{D}^*)$ *and in Game*$_3$ *over bases* $(\mathbb{B}, \mathbb{B}^*)$*. Thus, Game*$_{2-Q}$ *can be conceptually changed to Game*$_3$*.* □

Lemma 4.3.18 *For any adversary* $\mathcal{A}$*,* $\mathsf{Adv}_{\mathcal{A}}^{(3)}(\lambda) = 0$*.*

Proof 4.3.16 *The value of* b *is independent from the adversary's view in Game*$_3$*. Hence,* $\mathsf{Adv}_{\mathcal{A}}^{(3)}(\lambda) = 0$*.* □

Combining all theses proofs, we obtain that any adversary has no advantage in winning the security game. Adding to these the fact that Problem 1 bis and Problem 2 bis hold if

XDLin1, XDLin2 hold, we have proven Lemma 4.3.13.

Combining Lemmas 4.3.6 and 4.3.13 we proved Theorem 4.3.4.

4.3.3 Our PPKG-WIBE Scheme

We now present a privacy-preserving key generation WIBE instantiation. Our starting point is the WIBE scheme obtained by the combination of [3] generic construction of anonymous WIBE from IPE schemes (see Section 4.1.2), and the IPE of Lewko *et al.* [91]. Our main contribution here is to find a way to protect the attributes of the user by introducing a privacy-preserving interactive key generation. This is done by using the properties of so-called Dual Pairing Vector Spaces. Our scheme is proven secure in the generic group model, under well-known assumptions, namely Decisional Diffie-Hellman and n-extended Decisional Diffie-Hellman introduced in [91]. We start with the presentation of the WIBE that we will use as building block.

The Basic WIBE. The basic WIBE we will use for our construction is the one by Abdalla *et al.* [3], derived from Lewko *et al.* IPE [91]. The latter, for vector of length n, is using dual pairing vector space of dimension $2n+3$, and as we saw in Section 4.1.2, the WIBE for patterns of length L is running the IPE with $n = 2L$. Therefore the construction presented in Figure 4.14 is using dual pairing vector spaces of length $4L+3$. Here keys and ciphertexts patterns belong to $\{0, 1, \star\}^L$.

- Setup(λ, L): generate an asymmetric bilinear group $\Gamma = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e, p)$ and run Dual$(\mathbb{Z}_p^{4L+3})$ to get two dual pairing vector spaces $(\mathbb{D}, \mathbb{D}^*)$. The master secret key is msk $= \mathbb{D}^*$ and the public key is pk $= (g_1^{\mathbf{d}_1}, \cdots, g_1^{\mathbf{d}_{2L}}, g_1^{\mathbf{d}_{4L+1}}, g_1^{\mathbf{d}_{4L+3}})$.
- KeyGen(msk, $\boldsymbol{P}'$): run ExtendingKeyPattern$(\boldsymbol{P}')$ to get $\boldsymbol{u}$, then compute $\mathsf{sk}_{\mathbf{P}'} = g_2^{\rho \sum_{j=1}^{2L} u_j \mathbf{d}_j^* + \mathbf{d}_{4L+1}^* + \eta \mathbf{d}_{4L+2}^*}$, where $\rho, \eta \leftarrow \mathbb{Z}_p$.
- Encrypt(msk, $\boldsymbol{P}$): run ExtendingCtPattern$(\boldsymbol{P})$ to get $\boldsymbol{v}$, then compute $\mathbf{c}_1 = g_1^{\delta_1 (\sum_{j=1}^{2L} v_j \mathbf{d}_j) + \xi \mathbf{d}_{4L+1} + \delta_2 \mathbf{d}_{4L+3}}$, and $c_2 = \mathsf{m} \cdot e(g_1, g_2)^{\xi}$, where $\xi, \delta_1, \delta_2 \in \mathbb{Z}_p$ are chosen randomly. The ciphertext is ct $= (\mathbf{c}_1, c_2)$.
- Decrypt$(\mathsf{sk}_{\boldsymbol{P}'}, \mathsf{ct}, \boldsymbol{P})$: the decryption consists in executing the IPE decryption, as $c_2 / e(\mathbf{c}_1, \mathsf{sk}_{\mathbf{P}'})$.

Figure 4.14: Abdalla *et al.* [3] 's anonymous WIBE scheme.

The following theorem is directly deduced from [3], which shows that the above WIBE is anonymous if the underlying IPE is payload-hiding and attribute hiding and [91], which proves that the used IPE is payload-hiding and attribute-hiding under the n-eDDH assumption, where $n = 2L$ in our case.

Theorem 4.3.5 *The obtained WIBE is anonymous under the n-eDDH assumption.*

For the proof of the above theorem, refer to [91].

Privacy-preserving key generation: intuition. To obtain a privacy-preserving key generation, one idea could be to send the randomized extended version of a pattern. To do so, we define the algorithm presented in Figure 4.15.

Algorithm 4.3 ExtendingKeyPatternRandomized

Input: key pattern $\boldsymbol{P}$ of length n
Output: randomized pattern $\boldsymbol{u}$ of length $2n$

$i \leftarrow 1, j \leftarrow 1$
while $i \leq n, j \leq 2n$ **do**
  if $P_i \neq \star$ **then**
    $u_j \leftarrow s_i$ and $u_{j+1} \leftarrow s_i \cdot P_i$ for $s_i \leftarrow \mathbb{Z}_p$
  else
    $u_j \leftarrow 0$ and $u_{j+1} \leftarrow 0$
  end if
  $j \leftarrow j + 2, i \leftarrow i + 1$
end while
return $\boldsymbol{u}$

Figure 4.15: Randomization of ExtendingKeyPattern algorithm.

However, it does not work since e.g., $u_j \neq 0 \wedge u_{j+1} = 0$ for j an odd number reveals that $P_j = 0$. With the same reasoning one can recover positions of $\boldsymbol{P}$ equals to 1 and the ones equal to $\star$, therefore recover $\boldsymbol{P}$.

To avoid such problem, we propose to exploit the fact that [91]'s scheme makes use of dual pairing vector spaces properties, of dimension $2n + 3$. More precisely, in [91]'s scheme, a secret key for a vector $\nu \in \mathbb{Z}_p^n$ is $\mathsf{sk}_\nu = g_2^{\rho(\sum_{i=1}^{n} \nu_i \mathbf{d}_i^*) + \mathbf{d}_{2n+1}^* + \eta \mathbf{d}_{2n+2}^*}$, where $\rho, \eta \leftarrow \mathbb{Z}_p$. Our idea is then to ask the Pattern Audit Center (PAC) to compute $\mathbf{x} = \prod_{i=1}^{2n} g_2^{\rho \cdot u_i \cdot \mathbf{d}_i^*} = g_2^{\rho \sum_{i \in [2n]} u_i \cdot \mathbf{d}_i^*}$ where $u \in \mathbb{Z}_p^{2n}$ is the result of ExtendingKeyPattern$(\mathbf{P})$.

PAC sends the result to the Key Generation Center (KGC).

But this is not enough since if KGC is not honest, it computes $\mathbf{x}^{\mathbf{d}_j}$ for $j = 1, \cdots, 2n$. If $u_j \neq 0$ then the result is g_2, otherwise it is equal to 1, where 1 is the identity element of $\mathbb{G}_2$. From that, KGC can learn $\boldsymbol{P}$ as for j an odd number, if $\mathbf{x}^{\mathbf{d}_j} \neq 1 \wedge \mathbf{x}^{\mathbf{d}_{j+1}} = 1$ it reveals that $u_j = 10 \wedge u_{j+1} = 0$ thus that $P'_j = 0$. With the same kind of reasoning KGC can find positions equal to 1 and those equal to $\star$, therefore she can recover $\boldsymbol{P}$.

Our second trick is to multiply $\mathbf{x}$ with a "security" component $\mathbf{t} = g_2^{\sum_{j=1}^{2n} \tau_j \mathbf{d}^*_j}$ (where $\tau_j \leftarrow \mathbb{Z}_p$ for $j = 1, \cdots, 2n$). Hence, KGC is no more able to get 1 when computing $(\mathbf{x} \cdot \mathbf{t})^{\mathbf{d}_l}$ for some $l \in [2n]$. But in order to recover its secret key, a user now needs to remove the security component $\mathbf{t}$. To do so she will use a token given by PAC: $\mathbf{t}^{-1} = g_2^{-\sum_{j=1}^{2n} \tau_j \mathbf{d}^*_j}$. By multiplying the value received from KGC with $\mathbf{t}^{-1}$, the user will get its private key.

However as is, PAC and user can recover from a key query the values $g_2^{\mathbf{d}^*_{2n+1} + \eta \mathbf{d}^*_{2n+2}}$ and forged their own secret key.

To prevent this, we slightly modify the protocol so that KGC chooses ρ. First, user choose a random $\theta \in \mathbb{Z}_p$ and gives PAC g_2^{θ}. The latter sends to KGC $(\prod_{i=1}^{2n} g_2^{u_i \cdot \mathbf{d}^*_i} \cdot \prod_{j=1}^{2n} (g_2^{\theta})^{\tau_j \mathbf{d}^*_j}, \mathbf{t}' = g_2^{\sum_{j=1}^{2n} \tau_j \mathbf{d}^*_j})$. By raising the first received value to the power ρ and multiplying it by $g_2^{\mathbf{d}^*_{4n+1} + \eta \mathbf{d}^*_{4n+2}}$, KGC obtain the blind secret key. To recover its secret key, user is using $(\mathbf{t}')^{\rho}$ given by KGC and his knowledge of θ.

Note 4.3.3 *In order to make the above work, PAC must know a part of the secret key,* i.e. *she must know* $\mathbf{d}^*_1, \cdots, \mathbf{d}^*_{2n}$.

Our privacy-preserving key generation WIBE scheme. We now detail our privacy-preserving key generation WIBE instantiation. From the above, the Setup, Encrypt and Decrypt phases are the same than in Figure 4.14, our full scheme only modifies the key generation process. Following Definition 4.2.1, we need to define the following procedures: UserTemKeyGen(pk), BlindTokenGen(pk, $\mathbf{P}$, tpk_{user}), BlindKeyGen(pk, msk, $\mathsf{bt}_{\mathbf{P}}$) and KeyExtract($\mathsf{bsk}_{\mathbf{P}}$, tsk_{user}). Our full scheme is given in Figure 4.16. As in Definition 4.2.1,the name of key generation protocol algorithms that are ran by KGC are written in blue, those ran by user in green and in orange those that are run by PAC.

- Setup(λ, L): generate an asymmetric bilinear group $\Gamma = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e, p)$ and run Dual$(\mathbb{Z}_p^{4L+3})$ to get two dual pairing vector spaces $(\mathbb{D}, \mathbb{D}^*)$. The master secret key is msk $= (g_2^{\mathbf{d}^*_{4L+1}}, g_2^{\mathbf{d}^*_{4L+2}})$ and the public key is pk $= (g_1^{\mathbf{d}_1}, \cdots, g_1^{\mathbf{d}_{2L}}, g_1^{\mathbf{d}_{4L+1}}, g_1^{\mathbf{d}_{4L+3}}, g_2^{\mathbf{d}^*_1}, \cdots, g_2^{\mathbf{d}^*_{2L}})$.
- UserTemKeyGen(pk): generate at random $\theta \leftarrow \mathbb{Z}_p$ and compute $\Theta = g_2^{\theta}$. Then define $\mathsf{tsk}_{user} = \theta$ and $\mathsf{tpk}_{user} = \Theta$. The latter is sent to PAC.
- BlindTokenGen(pk, $\boldsymbol{P}$, tpk_{user}): run ExtendingKeyPattern$(\boldsymbol{P})$ to get $\boldsymbol{u}$, then compute $\mathbf{x} = g_2^{\sum_{l=1}^{2L} \mathbf{u}_l \cdot \mathbf{d}^*_l}$, $\mathbf{t} = \Theta^{\sum_{l=1}^{2L} \tau_l \mathbf{d}^*_l}$ and $\mathbf{t}' = g_2^{\sum_{l=1}^{2L} \tau_l \mathbf{d}^*_l}$ where $\tau_l \leftarrow \mathbb{Z}_p$ for $= 1, \cdots, 2L$. Set the blind token $\mathsf{bt}_{\mathbf{P}} = (\mathbf{x} \cdot \mathbf{t}, \mathbf{t}')$ where $\mathbf{x} \cdot \mathbf{t} = g_2^{\sum_{l=1}^{2L} \mathbf{u}_l \cdot \mathbf{d}^*_l + \theta \sum_{l=1}^{2L} \tau_l \mathbf{d}^*_l}$ and send it to KGC.
- BlindKeyGen(pk, msk, $\mathsf{bt}_{\mathbf{P}}$): pick two randoms $\rho, \eta \leftarrow \mathbb{Z}_p$ and compute $\mathsf{bsk}_2 = (\mathbf{t}')^{\rho} = g_2^{\rho \sum_{l=1}^{2L} \tau_l \mathbf{d}^*_l}$ and $\mathsf{bsk}_1 = (\mathbf{x} \cdot \mathbf{t})^{\rho} \cdot g_2^{\mathbf{d}^*_{4L+1} + \eta \mathbf{d}^*_{4L+2}} = g_2^{\rho \sum_{l=1}^{2L} \mathbf{u}_l \cdot \mathbf{d}^*_l + \rho\theta \sum_{l=1}^{2L} \tau_l \mathbf{d}^*_l + \mathbf{d}^*_{4L+1} + \eta \mathbf{d}^*_{4L+2}}$. The blind secret key $\mathsf{bsk} = (\mathsf{bsk}_1, \mathsf{bsk}_2)$ is given to user.
- KeyExtract(bsk, tsk_{user}): output final secret key: $\mathsf{sk}_{\mathbf{P}} = \mathsf{bsk}_1 \cdot (\mathsf{bsk}_2)^{-\mathsf{tsk}_{user}} = g_2^{\sum_{l=1}^{2L} \mathbf{u}_l \cdot \mathbf{d}^*_l + \mathbf{d}^*_{4L+1} + \eta \mathbf{d}^*_{4L+2}}$.
- Encrypt(msk, $\boldsymbol{P}$): run ExtendingCtPattern$(\boldsymbol{P})$ to get $\boldsymbol{v}$, then compute $\mathbf{c}_1 = g_1^{\delta_1 (\sum_{j=1}^{2L} v_j \mathbf{d}_j) + \xi \mathbf{d}_{4L+1} + \delta_2 \mathbf{d}_{4L+3}}$, and $c_2 = \mathsf{m} \cdot e(g_1, g_2)^{\xi}$, where $\xi, \delta_1, \delta_2 \in \mathbb{Z}_p$ are chosen randomly. The ciphertext is $\mathsf{ct} = (\mathbf{c}_1, c_2)$.
- Decrypt($\mathsf{sk}_{\boldsymbol{P}'}$, ct, $\boldsymbol{P}$): the decryption consists in executing the IPE decryption, as $c_2 / e(\mathbf{c}_1, \mathsf{sk}_{\mathbf{P}'})$.

Figure 4.16: Our privacy-preserving key generation identity-based encryption with wildcards scheme.

The adaptive indistinguishability and the anonymity of our instantiation come directly from the security of Lewko *et al.* [91] and the security of Abdalla *et al.* [3]'s construction. Now let us prove it privacy-preserving key generation security.

Theorem 4.3.6 *Our WIBE satisfies privacy-preserving key generation under DDH in* $\mathbb{G}_2$.

We prove Theorem 4.3.6 with two games:

- Game_0 is the original WIBE PPKG security game, as in Definition 4.2.2
- Game_1 is as Game_0 with the restriction that the adversary queries only challenge patterns that are different at one position exactly.

The idea of the proof is the following: we first prove that if there exists an adversary that can win Game_1, then we can build an adversary against DDH in $\mathbb{G}_2$ by using the latter's challenge to build to challenge token. Then we prove that we can build an adversary against Game_1 by using an adversary against Game_0.

Note 4.3.4 *To be able to reduce the privacy-preserving key generation security of our scheme to DDH, we actually need to modify our scheme presented in Figure 4.16 by replacing* ExtendingKeyPattern *by* ExtendingKeyPatternRandomized*.*

Lemma 4.3.19 *If DDH holds in $\mathbb{G}_2$, then the advantage of any adversary to win Game_1 is negligible.*

Proof 4.3.17 *We prove the contrapositive. Let $\mathcal{B}$ be an adversary against Game_1, that wins with non negligible advantage. We construct below an adversary $\mathcal{A}$ against DDH in $\mathbb{G}_2$, that wins with non negligible advantage.*

- `SETUP`*: challenger $\mathcal{C}$ chooses $\Gamma = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, g_T, p, e)$, randoms $a, b, c \leftarrow \mathbb{Z}_p$ and $b^{'} \in \{0,1\}$ randomly. If $b^{'} = 0$, it sets $t = g_2^{ab}$, and $t = g_2^{ab+c}$ otherwise. It sends $(\Gamma, g_2^a, g_2^b, t)$ to $\mathcal{A}$. The latter gives Γ to $\mathcal{B}$, who chooses $(\mathbb{D}, \mathbb{D}^*) \leftarrow \mathsf{Dual}(\mathbb{Z}_p^{4L+3})$ and sends $\mathbf{d}_1^*, \cdots, \mathbf{d}_{2L}^*$ to $\mathcal{A}$.*

- `TOKEN or KEYQUERY`*: $\mathcal{B}$ can create key for patterns $\mathbf{P}^{'}$ it chooses or asks $\mathcal{A}$ for the associated token. The latter answer with $bt_{\mathbf{P}'} \leftarrow \mathsf{BlindTokenGen}(\mathsf{pk}, \mathbf{P}^{'})$.*

- `CHALLENGE`*: $\mathcal{B}$ chooses $\mathbf{P}^0, \mathbf{P}^1$ such that for all $i \in [L], i \neq j$, $P_i^0 = P_i^1$ and $P_j^0 \neq P_j^1$ where $j \in [L]$. We suppose w.l.o.g. that $P_j^0 = \star$ and $P_j^1 = 0$. $\mathcal{B}$ sends both patterns to $\mathcal{A}$. The latter creates the vector $\tilde{\mathbf{u}}$ which is equal to vector $\mathbf{u}$ output by* ExtendingKeyPatternRandomized*, except that positions j and $j+1$ are removed. Notice that by definition of $\mathbf{P}^0, \mathbf{P}^1$ and by construction $\tilde{\mathbf{u}}$ is the same for both patterns, and $\tilde{\mathbf{u}}$ has size equal to $2L-2$. $\mathcal{A}$ creates the blind token $bt_{\mathbf{P}^b}$ as follows: it chooses $\{\tau_l\}_{l=1,\cdots,2L,l\neq j}$ and sets $bt_{\mathbf{P}^b} = g_2^{\sum_{l=1}^{2L-2} \tilde{u}_l \cdot \mathbf{d}_l^*} \cdot (g_2^a)^{\sum_{l=1,l\neq j}^{2L} \tau_l \mathbf{d}_l^*} \cdot t^{\mathbf{d}_j^*}$, $g_2^{\sum_{l=1,l\neq j}^{2L} \tau_l \mathbf{d}_l^*} \cdot (g_2^b)^{\mathbf{d}_j^*}$. It gives $bt_{\mathbf{P}^b}$ to $\mathcal{B}$.*

- `TOKEN or KEYQUERY`*: is the same than the previous* `TOKEN or KEYQUERY` *step.*

- `GUESS`*: $\mathcal{B}$ outputs a bit $b^{'}$ to $\mathcal{A}$, who outputs it as its guess.*

Analysis: if $b^{'} = 0$*, then* $t = g_2^{ab}$ *and* $bt_{\mathbf{P}^b} = g_2^{\sum_{l=1}^{2L-2} \tilde{u}_l \cdot \mathbf{d}_l^* + a(\sum_{l=1, l\neq j}^{2L} \tau_l \mathbf{d}_l^* + b\mathbf{d}_j^*)}, g_2^{\sum_{l=1, l\neq j}^{2L} \tau_l \mathbf{d}_l^* + b\mathbf{d}_j^*}$*. By setting* $\theta = a$ *and* $\tau_j = b$*, we obtain a blind token for the pattern equals to* $\star$ *at position* j *and* $tsk_{user} = g_2^a$*.*

If $b^{'} = 1$*, then* $t = g_2^{ab+c}$ *and* $bt = g_2^{\sum_{l=1}^{2L-2} \tilde{u}_l \cdot \mathbf{d}_l^* + c\mathbf{d}_j^* + a(\sum_{l=1, l\neq j}^{2L} \tau_l \mathbf{d}_l^* + b\mathbf{d}_j^*)}, g_2^{\sum_{l=1, l\neq j}^{2L} \tau_l \mathbf{d}_l^* + b\mathbf{d}_j^*}$*. By setting* $\theta = a$*,* $\tau_j = b$ *and* $u_j = c$*, we obtain a blind token for the pattern equals to* 0 *at position* j *and* $tsk_{user} = g_2^a$*.* □

Lemma 4.3.20 *Game*$_1$ *implies Game*$_0$*.*

Proof 4.3.18 *We prove the contrapositive. Let* $\mathcal{B}$ *be an adversary that breaks PPKG game (Game*$_0$*) with non negligible advantage. We construct below an adversary* $\mathcal{A}$ *against Game*$_1$ *that wins with non negligible advantage.*

- `SETUP`*: challenger* $\mathcal{C}$ *runs* $\mathsf{Setup}(\lambda, 1^L)$ *to get* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, g_T, p, e)$ *and sends it to* $\mathcal{A}$ *who sends it to* $\mathcal{B}$*. The latter chooses* $(\mathbb{D}, \mathbb{D}^*) \leftarrow \mathsf{Dual}(\mathbb{Z}_p^{4L+3})$ *and sends* $\mathbf{d}_1^*, \cdots, \mathbf{d}_{2L}^*$ *to* $\mathcal{A}$*, who sends it to* $\mathcal{C}$*.*

- `TOKEN or KEYQUERY`*:* $\mathcal{B}$ *can create key for patterns* $\mathbf{P}^{'}$ *it chooses or ask* $\mathcal{A}$ *for a token associated to the chosen pattern.* $\mathcal{A}$ *asks* $\mathcal{C}$ *for the token. The latter responds with* $bt_{\mathbf{P}'} \leftarrow \mathsf{BlindTokenGen}(\mathsf{pk}, \mathbf{P}^{'})$*.*

- `CHALLENGE`*:* $\mathcal{A}$ *chooses* $\mathbf{P}^0, \mathbf{P}^1$ *such that for all* $i \in [2L], i \neq j$*,* $P_i^0 = P_i^1$ *and* $P_j^0 \neq P_j^1$ *where* $j \in [L]$*; it sends it to* $\mathcal{C}$ *who chooses* $b \in \{0, 1\}$ *and creates a token* $\theta_{\mathbf{P}^b} \leftarrow \mathsf{BlindTokenGen}(\mathsf{pk}, \mathbf{P}^b)$*. It sends* $\theta_{\mathbf{P}^b}$ *to* $\mathcal{A}$ *who sends* $\theta_{\mathbf{P}^b}$ *and* $\mathbf{P}^0, \mathbf{P}^1$ *to* $\mathcal{B}$*.*

- `TOKEN or KEYQUERY`*: is the same than the previous* `TOKEN or KEYQUERY` *step.*

- `GUESS`*:* $\mathcal{B}$ *outputs a bit* $b^{'}$ *to* $\mathcal{A}$ *who outputs it as its guess.* □

As $\mathcal{B}, \mathcal{A}$ *have no restriction on the challenge patterns, the simulation is perfect. Thus* $\mathcal{A}$ *wins with a non negligible advantage.*

Combining lemmas 4.3.19 and 4.3.20, we prove Theorem 4.3.6.

4.4 Conclusion of This Chapter

This chapter introduced identity-based encryption with wildcards (WIBE) schemes. Our contributions regarding that primitive is the definition of a stronger security property, call *pattern-hiding* along with the introduction of a new kind of WIBE: privacy-preserving key generation (ppkg) WIBE. In such scheme, the key generation is replaced by an interactive protocol between three entities such that the entity that creates the key does not learn the pattern associated to key, that is chosen by the entity requiring the key.
We also proposes three instantiations of WIBE: one scheme with constant size ciphertext, one pattern-hiding scheme and one privacy-preserving key generation scheme. It is interesting to notice that in our schemes, adding the pattern-hiding security property is possible only at the cost of an efficiency loose. Indeed while we were able to obtain a WIBE scheme with constant size ciphertext, we were only able to build a pattern-hiding WIBE that has *linear* ciphertext size. We leave as an open problem the proof that efficient pattern-hiding WIBE schemes are possible, and if applicable such a construction. Regarding our privacy-preserving key generation WIBE, we started from an existing WIBE scheme proposed by Abdalla *et al.* [3] and transformed it to obtain a privacy-preserving key generation scheme. It might be interesting to find a generic way to transform a WIBE scheme into a PPKG WIBE scheme.
In Chapter 6.1 we will see that our pattern-hiding security property is required in order to build a specific data sharing scheme, called *augmented broadcast encryption*, from a WIBE. Then, we will use our pattern-hiding WIBE instantiation to build such a scheme. We will also use our third instantiation for a specific use case, presented also in Chapter 6.3.

5

Second Cryptographic Tool: Cryptographic Accumulators

Contents

THE second primitive we used as a building block is called *cryptographic accumulators*. Briefly a cryptographic accumulator is a primitive that allows the representation of a set of values by a short object (the accumulator) and offers the possibility to prove that some input values are in the accumulator. Cryptographic accumulators were introduced in 1993 by Benaloh and De Mare [27], and since then a lot of new properties, definitions and notations were introduced. For example, originally any modification of the represented set required to recompute the associated accumulator. In 2002, Camenisch and Lysyanskaya [46] proposed cryptographic accumulators that support sets modifications, meaning that after a modification of the represented set the accumulator can be updated and does not have to be recomputed. They called this kind of accumulators *dynamic* and qualified original accumulators as *static*. We can also cite the property of *universal* cryptographic accumulators introduced by Li, Li and Xue [98] in 2007: such schemes provide as any cryptographic accumulator (called then *non-universal*) proof of membership, but also proof of non-membership for elements not in the represented set. We start, in Section 5.1.1, by giving a definition of static and universal cryptographic accumulators, along with an up-to-date overview of cryptographic accumulators and state of the art. In Section 5.2 we propose several discussions on accumulators, on the properties of *undeniability* [102] and *delegatable* [7] and on accumulators' applications. Then Section 5.3 presents two of our contributions regarding cryptographic accumulators: a new security property for cryptographic accumulator that protects against the forgery of accumulators computed using the scheme's secret key; and the introduction of a new feature for cryptographic accumulators, called *dually computable* accumulators. We end this chapter with Section 5.4 that presents our two accumulators schemes: the first scheme improves the state of the art by being the first scheme using dual pairing vector spaces and serves as a building block for our second instantiation, which is the first dually computable accumulator.

5.1 Cryptographic Accumulators

5.1.1 Definitions

Definition 5.1.1 ***Static universal accumulator*** *[27, 63, 57]. A static universal cryptographic accumulator scheme is a tuple of efficient algorithms defined as follows:*

- $\mathsf{Gen}(\lambda, \mathfrak{b})$*: the generation algorithm takes as input a security parameter* λ *and a bound* $\mathfrak{b} \in \mathbb{N} \cup \{\infty\}$ *such that if* $\mathfrak{b} \neq \infty$ *then the number of elements that can be accumulated is bounded by* $\mathfrak{b}$*. It returns a key pair* (sk_{acc}, pk_{acc})*, where* $sk_{acc} = \varnothing$ *if no trapdoor exists and* pk_{acc} *contains the parameter* $\mathfrak{b}$*.*
- $\mathsf{Eval}((\mathsf{sk}_{\mathsf{acc}},)\mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$*: the evaluation algorithm takes as input the accumulator (secret key* sk_{acc} *and) public key* pk_{acc} *and a set* $\mathcal{X}$ *to be accumulated. It returns an accumulator* $acc_{\mathcal{X}}$ *together with some auxiliary information aux.*
- $\mathsf{WitCreate}((\mathsf{sk}_{\mathsf{acc}},)\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}}, \mathsf{aux}, x, \mathsf{Type})$*: the witness creation algorithm takes as input the accumulator (secret key* sk_{acc} *and) public key* pk_{acc}*, an accumulator* $acc_{\mathcal{X}}$*, the associated set* $\mathcal{X}$*, auxiliary information aux, an element* x *and a boolean Type. If* $\mathsf{Type} = 0$ *and* $x \in \mathcal{X}$ *it outputs a membership witness* $mwit_x^{\mathcal{X}}$ *and if* $\mathsf{Type} = 1$ *and* $x \notin \mathcal{X}$ *it outputs a non-membership witness* $nmwit_x^{\mathcal{X}}$*.*
- $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{wit}_x^{\mathcal{X}}, x, \mathsf{Type})$*: the verification algorithm takes as input the accumulator public key* pk_{acc}*, an accumulator* $acc_{\mathcal{X}}$*, a witness* $wit_x^{\mathcal{X}}$*, an element* x *and a boolean Type. If* $wit_x^{\mathcal{X}} = mwit_x^{\mathcal{X}}$*,* $x \in \mathcal{X}$ *and* $\mathsf{Type} = 0$ *it returns* 1*, if* $wit_x^{\mathcal{X}} = nmwit_x^{\mathcal{X}}$*,* $x \notin \mathcal{X}$ *and* $\mathsf{Type} = 1$ *it returns* 1*, otherwise it returns* 0*.*

Note 5.1.1 *If* $\mathfrak{b} \neq \infty$ *then we say that the accumulator is* bounded *[15].*

Notation 5.1.1 *As one may have notice, membership witnesses are written "mwit" while non-membership witnesses are denoted by "nmwit". When considering general witnesses,* i.e. *without considering its type, we will write wit. Sometimes when no confusion is possible, we will drop the notation in the exponent of the associated set.*

Note 5.1.2 *Regarding the way witnesses are generated in* WitCreate*, the literature gives four possibilities: (i) only using the* public key *[101], (ii) using the* secret key *[82], (iii) using the public key or in a more efficient way using the secret key [14, 57], (iv) using a specially created private key, called the evaluation key [73]. The evaluation algorithm can take as input either both* sk_{acc} *and* pk_{acc} *or only just one. If* Eval *takes as input* sk_{acc} *(resp.* pk_{acc}*) solely, we say that the accumulator has* private evaluation *(resp.* public evaluation*). The same goes for* WitCreate*: depending on its input the accumulator either has* private witness generation *or* public witness generation*.*

Some accumulators are trapdoorless *meaning that* $sk_{acc} = \varnothing$.

Definition 5.1.2 ***Correctness.*** *A static, universal accumulator is said to be* correct *if for all security parameters* λ*, all integer* $\mathfrak{b} \in \mathbb{N} \cup \{\infty\}$*, all set of values* $\mathcal{X}$*, and all elements* x, y *such that* $x \in \mathcal{X}$ *and* $y \notin \mathcal{X}$*:*

$$\Pr\left[\begin{array}{c}(sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda, \mathfrak{b}), (\mathsf{acc}_{\mathcal{X}}, \mathsf{aux}) \leftarrow \mathsf{Eval}((\mathsf{sk}_{\mathsf{acc}},)\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}),\\ mwit_x^{\mathcal{X}} \leftarrow \mathsf{WitCreate}((\mathsf{sk}_{\mathsf{acc}},)\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathcal{X}, \mathsf{aux}, x, \mathsf{Type} = 0) \wedge\\ nmwit_y^{\mathcal{X}} \leftarrow \mathsf{WitCreate}((\mathsf{sk}_{\mathsf{acc}},)\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathcal{X}, \mathsf{aux}, x, \mathsf{Type} = 1) :\\ \mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{mwit}_x, x, \mathsf{Type} = 0) = 1\\ \wedge \mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{nmwit}_y, y, \mathsf{Type} = 1) = 1\end{array}\right] = 1$$

Regarding security of cryptographic accumulators, several notions were introduced such as *undeniability* [102], *indistinguishability* [57] or *zero-knowledge* [73] for example. We here only formally present the property of *collision resistance*, but in Section 5.1.2 we give an exhaustive list of all accumulator security properties and an informal definition for all of them. Informally a cryptographic accumulator is said to be *collision resistant* if it is hard for an adversary to forge a membership (resp. non-membership) witness for an element that is not (resp. that is) in the accumulated set.

Definition 5.1.3 ***Collision resistance*** *[18, 57]. A static universal accumulator scheme is said to satisfy* collision resistance *if all PPT adversaries* $\mathcal{A}$ *have at most negligible advantage in the game presented in Figure 5.1, where* $\mathcal{A}$*'s advantage is defined as*

$$\mathsf{Adv}_{\mathcal{A}}^{CR}(\lambda) := \Pr\left[\begin{array}{c}\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 0) = 1 \wedge x^* \notin \mathcal{X}^*\\ \vee \mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 1) = 1 \wedge x^* \in \mathcal{X}^* :\\ (sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda, \mathfrak{b}), (\mathcal{X}^*, \mathsf{wit}_{x^*}, x^*) \leftarrow \mathcal{A}(\mathsf{pk}_{\mathsf{acc}})\end{array}\right]$$

Let $\mathcal{C}$ *be a challenger.*

Note 5.1.3 *If the accumulator is* non-universal*, then in the above definitions remove the non-membership related parts.*

We now give the definition of two properties namely, *subset query* and *multiset setting*, that are satisfied by the accumulator schemes we will present in Section 5.4.

Definition 5.1.4 ***Subset query*** *[59, 73, 101]. A static (non-)universal accumulator is said to satisfy* subset query *if witnesses can be generated for a subset of the accumu-*

SETUP: on input λ, $\mathcal{C}$ runs Gen(λ) to get $(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}})$ and sends $\mathsf{pk}_{\mathsf{acc}}$ to $\mathcal{A}$.
ACCUMULATOR QUERY: $\mathcal{A}$ chooses a set $\mathcal{X}$ and sends it to $\mathcal{C}$. $\mathcal{C}$ returns to $\mathcal{A}$ $\mathsf{acc}_{\mathcal{X}}$, where $\mathsf{acc}_{\mathcal{X}} \leftarrow \mathsf{Eval}((\mathsf{sk}_{\mathsf{acc}},)\mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$.
WITNESSQUERY: $\mathcal{A}$ chooses a set $\mathcal{X}$ (or an accumulator $\mathsf{acc}_{\mathcal{X}}$ previously queried), an element x and a boolean Type and sends all to $\mathcal{C}$. The latter runs $\mathsf{WitCreate}((\mathsf{sk}_{\mathsf{acc}},)\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathcal{X}, x, \mathsf{Type})$ to get wit_x and returns it to $\mathcal{A}$.
GUESS: $\mathcal{A}$ returns a set $\mathcal{X}^*$, an element x^* and a witness wit_{x^*} and wins if $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 0) = 1 \wedge x^* \notin \mathcal{X}^*$ or $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 1) = 1 \wedge x^* \in \mathcal{X}^*$.
ACCUMULATOR QUERY: $\mathcal{A}$ can continue to query accumulator for sets $\mathcal{X}$.
WITNESSQUERY: $\mathcal{A}$ can continue to query witnesses for elements x.

Figure 5.1: Collision resistance security game.

lated set rather than individual elements.

Definition 5.1.5 ***Multiset setting*** *[62, 73, 35]. A static (non-)universal accumulator is said to satisfy* multiset setting *if sets that can be accumulated are multisets, meaning that an element can be present more than once in the set. In this case, each element is associated to a count (belonging to* $\mathbb{N}$*) that is equal to* 0 *when the element is not in accumulated.*

5.1.2 Overview And State of The Art

As stated previously, through the years accumulators have been used for multiple purposes. This results in new properties specific to individual needs and is how accumulators became *dynamic* [46], *universal* [98], *multisets* [62] or even the recent property of *zero-knowledge* [73], a privacy notion for accumulators. Unfortunately, all these new properties and functionalities were added separately, giving rise to several definitions of accumulators. That makes it complicated to have an overview of accumulators and their properties. That is why in 2015, Derler *et al.* [57] proposed a unified formal model, dealing with most of existing accumulators' properties. Their work became a reference when working with accumulators. However, since 2015 new relevant properties of accumulators have been introduced, such as the *zero-knowledge* [73] security property which extends *indistinguishability* [57], or the *asynchronous* [124] property which allows witnesses to be still correct when a fixed number of operations on the accumulated set have been carried out, without needing an update. Some functionalities, such as witnesses computed not for a single element but for a subset, or the multiset setting which allows an element to be accumulated more than once were not taken into account in the

work of Derler *et al.*. In this section we present an up-to-date overview of accumulators properties following Derler *et al.* definition for accumulators, and an updated state of the art.

All properties presented in this section will be presented for asymmetric accumulators but notice that many of them applied for symmetric accumulators as well. Our choice is motivated by asymmetric accumulator better efficiency (refer to Section 5.2.1 for more details about symmetric accumulators efficiency). We first list the features of accumulators. We do not present *correctness*, *subset query*, *multiset*, and *bounded* properties as they are formally given in Section 5.1.1.

- **Sizes** [45]: accumulator and witness sizes should be independent of the number of accumulated elements.
- **Dynamic** [46]: an accumulator that additionally provides efficient algorithms (Add, Delete, WitnesUpdate) that respectively adds/removes elements from the accumulated set and the accumulator, and updates the witness accordingly. Notice that Add and Delete also output updated information aux.
- **Publicly Updatable** [57]: updates performed without the secret key.
- **Universal** [98]: witnesses can be generated to prove membership or non-membership. When the accumulator is non-universal, then the witness creation and verification algorithms of Definition 5.1.1 have a new syntax: formally, witness creation algorithm is WitCreate$((\mathsf{sk}_{\mathsf{acc}},)\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}}, x)$ and verification algorithm is Verify$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, x, \mathsf{wit}_x)$.
- **Delegatable non-membership proofs** [7]: it is possible for a user to give to another entity the ability to prove non-membership of the former's element, without the latter knowing the concerned element. The main idea is to replace the witness by a *proof* of a proof system, satisfying some properties.
- **Trusted vs. Non-Trusted Setup** [57]: in the trusted setup model a trusted third party runs the setup algorithm Gen and discards $\mathsf{sk}_{\mathsf{acc}}$ afterwards, while in the non-trusted model such trusted third party does not exist. When considering the state of the art it seems most reasonable (regarding the efficiency of the schemes) to define a security model with respect to such trusted setup as [57] did and as we will do subsequently. We emphasize that this model is compatible with all existing constructions.
- **Low Update Frequency** [124]: the accumulator is dynamic, and witnesses do not have to be updated at each update of the accumulator (for witnesses associated to elements not added or removed of the accumulator).
- **Old Accumulator Compatibility** [124]: the accumulator is dynamic, and verification still holds with an updated witness and an old (not updated) accumulator, for an element already present (or not) in the old accumulator.

- **Asynchronous** [124]: the accumulator satisfies both *low update frequency* and *old accumulator compatibility*.
- **Determinantal** [103]: a accumulator with a structure consistent with CLPØ-style ([53]) set (non-)membership non interactive zero knowledge scheme.
- **Dually computable** [22]: an accumulator with two evaluation algorithms, one that takes as input only the secret key of the scheme while the other takes as input the public key solely. Outputs of both algorithms are distinguishable. This feature is one of our contributions regarding cryptographic accumulators and we present it more in details in Section 5.3.

We now give all security properties that could be found in the literature. The basic property of a secure cryptographic accumulator is the impossibility for an adversary to prove that a value is accumulated while this is not the case. This property, known as *collision resistance* is formally given in Definition 5.1.3 therefore we do not present it again here. Based on that, several definitions have been proposed in the literature and we discuss all of them in the case of dynamic and universal accumulators.

- **One-Wayness** [27]: it is hard for an adversary who is given a set $\mathcal{X} = (x_1, \cdots, x_N)$, their accumulation result $\text{acc}_{\mathcal{X}}$, and another value $x^{'} \notin \mathcal{X}$ (resp. $x^{'} \in \mathcal{X}$) to output a value $\text{wit}^{'}$ such that $\text{Verify}(\text{pk}_{\text{acc}}, \text{acc}_{\mathcal{X}}, x^{'}, \text{wit}^{'}, 0) = 1$ (resp. $\text{Verify}(\text{pk}_{\text{acc}}, \text{acc}_{\mathcal{X}}, x^{'}, \text{wit}^{'}, 1) = 1$).
- **Strong One-Wayness** [18]: given $\mathcal{X} = (x_1, \cdots, x_N)$ and acc, it is hard for an adversary to output $x^{'} \notin \mathcal{X}$ (resp. $x^{'} \in \mathcal{X}$) and $\text{wit}^{'}$ such that $\text{Verify}(\text{pk}_{\text{acc}}, \text{acc}_{\mathcal{X}}, x^{'}, \text{wit}^{'}, 0) = 1$ (resp. $\text{Verify}(\text{pk}_{\text{acc}}, \text{acc}_{\mathcal{X}}, x^{'}, \text{wit}^{'}, 1) = 1$).
- **Undeniability** [102]: it is hard for an adversary to output an accumulator acc, a value $x^{'}$ and two witnesses $\text{wit}^{'}$ and $\text{wit}^{''}$ such that both $\text{Verify}(\text{pk}_{\text{acc}}, \text{acc}_{\mathcal{X}}, x^{'}, \text{wit}^{'}, 0) = 1$ and $\text{Verify}(\text{pk}_{\text{acc}}, \text{acc}_{\mathcal{X}}, x^{'}, \text{wit}^{''}, 1) = 1$) hold.
- **One-Way-Domain** [59]: the accumulator is collision resistant, and the set of values that can be accumulated is the span of a one-way function. Hence, it is computationally intractable to find witnesses for random values in the accumulator's domain.
- **Indistinguishability** [57]: given the public key, the adversary chooses two sets $\mathcal{X}_0$ and $\mathcal{X}_1$ and obtain the evaluation of one of the two. It has to decide which one.
- **Zero-knowledge accumulator** [73]: accumulated value, and (non-)membership witnesses leak nothing about the accumulated set at any given point in the security game (even after insertions and deletions, if the accumulator is dynamic).
- **Element hiding** [16]: publicly available auxiliary information aux output by update algorithms (Add or Delete) and associated to an accumulator does not lead any information about the elements in the accumulated set.

- **Add-Del indistinguishability** [16]: no adversary given publicly available information aux output by update algorithms (Add or Delete) can learn if an operation is an addition or a deletion.
- **Obliviousness** [16]: when both element hiding and Add-Del indistinguishability hold.

Note 5.1.4 *Some works [46, 113, 14, 97] complete cryptographic accumulator with zero-knowledge proof-of-knowledge protocols: a client that knows his value x is (or is not) in $\mathcal{X}$, can efficiently prove to a third-party that his value is (resp. is not) in the set, without revealing x. This privacy notion is different from the one we will focus on, zero-knowledge notion of [73] in which the entire protocol execution (as observed by a curious client or an external attacker) leaks nothing.*

In Table 5.1, we present a comparison of existing constructions for accumulators. We present them according to the four categories we have previously seen, and compare them with respect to the different properties and functionalities that they provide.

Note 5.1.5 *When doing the state of the art we noticed something surprising: there is no accumulator scheme with private evaluation (meaning that* Eval *takes as input sk_{acc}) and public witness generation (meaning that* WitCreate *takes as input only pk_{acc}). Indeed, either both evaluation and witness creation are either public [101] or private [73], or witness generation is private while evaluation is public [82]. We summarize in Table 5.2 how evaluation and witness generation are done, depending on the type of accumulator scheme.*

In Section 5.4.1 we present a pairing based accumulator scheme that has private evaluation and public witness generation, filling the above gap in accumulators state of the art.

5.2 Discussions on Accumulators

In the accumulators literature, it is admitted without formal proof that symmetric accumulators cannot have a size less than linear in the number of accumulated elements.

[2] If we do not take into account the work of [28].

[3] Actually, [132] improves the dynamic property of [73]: the latter has efficient membership witness update but inefficient non-membership witness update. The former proposes a way to update non-membership witnesses efficiently.

[4] Secret key can be given for witness generation in order to improve efficiency. Creation is still possible without it.

Table 5.1: Comparison of existing asymmetric accumulators constructions. A $\surd$ indicates that the property has been proven by another paper. “Sec.”, “ST”,“ U”, “T” respectively means “section”, “semi trusted", “untrusted” and “trusted”. When a notion is not specified in a paper, but implied by another one which is satisfied by the paper’s definition, then we indicate this relation by a $\surd$. When a notion is informally defined we indicate it by a $\surd$, and when a slightly different notion is defined we indicate it by a $\approx$.

		Functionalities						Properties									Security Properties							
Type	Schemes	Evaluation key	Witness generation	Constant size witness	Constant size acc. value	Determinantal	Dually Computable	Dynamic	Publicly updatable	Universal	Delegatable proofs	Asynchronous	Setup	Bounded	Subset query	Multiset	One-way	Strong one-way	Collision resistant	Undeniable	One-way domain	Indistinguishable	Zero-knowledge	Obliviousness
Hash based	[44]		Public					$\surd$		$\surd$			U				$\surd$	$\surd$	$\surd$	$\approx$				
	[125]		Public					$\surd$				$\surd$	U				$\surd$	$\surd$	$\surd$					
Lattice based	[82]		Private	$\surd$	$\surd$								T				$\surd$	$\surd$	$\surd$					
	[99]		Public										T				$\surd$	$\surd$	$\surd$					
Pairing based	[113]		Public	$\surd$	$\surd$			$\surd$					T	$\surd$			$\surd$	$\surd$	$\surd$					
	[15]		Public	$\surd$	$\surd$			$\surd$					T	$\surd$			$\surd$	$\surd$	$\surd$					
	[54]		Public	$\surd$	$\surd$					$\surd$			T				$\surd$	$\surd$	$\surd$					
	[45]		Public	$\surd$	$\surd$			$\surd$					T				$\surd$	$\surd$	$\surd$					
	[14]		Both	$\surd$	$\surd$			$\surd$		$\surd$			T				$\surd$	$\surd$	$\approx$ [2]		$\surd$			
	[8]		Public					$\approx$		$\surd$	$\surd$		T				$\surd$	$\surd$	$\surd$					
	[73]	$\surd$	Private	$\surd$	$\surd$			$\surd$		$\surd$			T	$\surd$			$\surd$	$\surd$	$\surd$			$\surd$	$\surd$	
	[101] s.1		Public	$\surd$	$\surd$								T		$\surd$		$\surd$	$\surd$	$\surd$					
	[101] s.2		Public	$\surd$	$\surd$					$\surd$			T				$\surd$	$\surd$	$\surd$	$\surd$		$\surd$		
	[132]		Public	$\surd$	$\surd$			$\surd$ [3]		$\surd$			T	$\surd$	$\surd$		$\surd$	$\surd$	$\surd$	$\surd$				
	[103]		Public	$\surd$	$\surd$	$\surd$				$\surd$			ST					$\approx$						
	Sec. 5.4.2/ [22]		Public	$\surd$	$\surd$		$\surd$						T	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$					
Number theoretic	[27]		Public										T				$\surd$							
	[18]		Public	$\surd$	$\surd$								T				$\surd$	$\surd$	$\surd$					
	[46]		Public	$\surd$	$\surd$			$\surd$					T				$\surd$	$\surd$	$\surd$		$\surd$			
	[59]		Public	$\surd$	$\surd$								T				$\surd$	$\surd$	$\surd$		$\surd$			
	[98]		Both	$\surd$	$\surd$			$\surd$		$\surd$			T				$\surd$	$\surd$	$\surd$					
	[134]		Private	$\surd$				$\surd$					T				$\surd$	$\surd$	$\surd$					
	[35]		Public	$\surd$	$\surd$			$\surd$		$\surd$			T				$\surd$	$\surd$	$\surd$	$\surd$				

Table 5.2: Comparison of evaluation and witness creation according to the type of accumulator instantiation.

Type	Evaluation	Witness Generation
Hash based	Public Public	Public Public
Lattices	Public	Private
Number Theoretic	Public	Public [4]
Pairing based	Public Private	Public Private

In this section we formally define symmetric accumulators and prove the lower bound on their size. Additionally we propose some discussions about relations between

accumulators security, about *undeniability* [102] and *delegatable* [7] properties. We complete this section with a discussion on future possible applications of accumulators, following our new usage of cryptographic accumulators for encryption that we present more in details in Section 6.2.

5.2.1 Symmetric Accumulators

Informally, an *asymmetric* accumulator requires witness for verification, while a *symmetric* accumulator does not require a witness for verification [88, 119] (and is mostly trapdoor-less meaning that $\mathsf{sk}_{\mathsf{acc}} = \varnothing$). The two types of accumulators were formally named in [88]. In this section we fill a gap in the accumulator literature: while it was admit that symmetric accumulators constructions produce large size accumulated value, this has never been properly prove. We first formally define symmetric accumulator and then prove that their size cannot be less than linear in the number of accumulated elements.

Definition 5.2.1 ***Symmetric cryptographic accumulator*** *[27]. A symmetric cryptographic accumulator scheme consists in three algorithms:*

- $\mathsf{Gen}(\lambda)$*: the generation algorithm takes as input a security parameter* λ *and outputs a pair of public-secret keys* (pk_{acc}, sk_{acc})*.*
- $\mathsf{Eval}((\mathsf{sk}_{\mathsf{acc}}), \mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$*: the evaluation algorithm takes as input a (secret key* sk_{acc}*, a) public key* pk_{acc} *and a set of elements* $\mathcal{X}$*. It outputs the accumulator* $acc_{\mathcal{X}}$ *of* $\mathcal{X}$*.*
- $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, x)$*: the verification algorithm takes as input a public key* pk_{acc}*, an accumulator* $acc_{\mathcal{X}}$ *and an element* x*. It outputs* 1 *if* $x \in \mathcal{X}$*, and* 0 *otherwise.*

Definition 5.2.2 ***Correctness.*** *A symmetric cryptographic accumulator scheme is said to be* correct *if for all security parameter* λ*, every honestly generated key pair* $(sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda)$*, every set of elements* $\mathcal{X}$ *and every element* $x \in \mathcal{X}$*:*

$$\Pr\left[\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{Eval}((\mathsf{sk}_{\mathsf{acc}},)\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}), x) = 1\right] = 1.$$

Definition 5.2.3 ***Symmetric accumulator one-wayness*** *[27]. A symmetric accumulator scheme is said to be* one-way *if all PPT adversaries* $\mathcal{A}$ *have at most negligible advantage in the game presented in Figure 5.2, where* $\mathcal{A}$*'s advantage is defined as*

$$\mathsf{Adv}_{\mathcal{A}}^{\text{One-Way}}(\lambda) := \Pr\left[\begin{array}{c} acc_{\mathcal{X}^*} = acc_{\mathcal{X}} \wedge \mathcal{X} \neq \mathcal{X}^*\mathcal{X}^* \leftarrow \mathcal{A}(pk_{acc}, \mathcal{X}, acc_{\mathcal{X}}): \\ acc_{\mathcal{X}} \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}, \mathcal{X}) \\ acc_{\mathcal{X}^*} \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}, \mathcal{X}^*) \end{array}\right].$$

Let $\mathcal{C}$ *be a challenger.*

SETUP: on input λ, $\mathcal{C}$ runs Gen(λ) to get $(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}})$. She chooses a set of elements $\mathcal{X}$, runs Eval$(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$ to get $\mathsf{acc}_{\mathcal{X}}$ and sends $(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}})$ to $\mathcal{A}$.
GUESS: $\mathcal{A}$ returns a set $\mathcal{X}^*$ and wins if $\mathcal{X}^* \neq \mathcal{X}$ and $\mathsf{acc}_{\mathcal{X}^*} = \mathsf{acc}_{\mathcal{X}}$ where $\mathsf{acc}_{\mathcal{X}^*} \leftarrow$ Eval$(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}, \mathcal{X}^*)$.

Figure 5.2: Symmetric accumulators one-wayness security game.

Theorem 5.2.1 *A symmetric accumulator cannot produce accumulated value with size less than linear in the number of elements in the accumulated set.*

Proof 5.2.1 *Let $N \in \mathbb{N}$, $[N] = \{1, \cdots, N\}$ and D the set of all possible sets over $[N]$. Let X be a random variable over D. X is uniformly distributed over D, thus as $|D| = 2^N$ we have that the entropy $H(X)$ is equal to $log_2(2^N) = N$. Suppose that there exists an accumulator that accumulated X into a short value, i.e. a value with size less than linear in N. Then, as the minimal amount of information needed to represent X is $H(X) = N$, there exists two sets (i.e. two values of X) that have the same accumulated value thanks to the pigeonhole principle. Thus the accumulator is not one-way.*□

5.2.2 Relations Between Security Properties

When looking at accumulators' security properties we can classify them into two categories: one that protects the witness (i.e. that prevents forgery of witnesses), and one that protects the accumulated set (i.e. that hides information about the set). We present this classification in Table 5.3.

Table 5.3: Classification of accumulator security properties.

Protect the witness	Protect the accumulated set
(Strong) One-wayness	Indistinguishability
Collision resistance	Zero knowledge
One-way domain	Obliviousness
Undeniability	

Plus we notice that the properties in the first column are computational ones while in the second column they are decisional.

Note 5.2.1 *Properties that protect the accumulated set define privacy security for accumulators schemes. As already observed in [56, 106, 57], when formulating a notion of privacy for cryptographic accumulators the fact that the accumulation value computation must be randomized becomes evident.*

Comparison between indistinguishability and zero-knowledge. The notion of zero-knowledge differs from the privacy notion of [57], by protecting not only the originally accumulated set but also all subsequent updates.

In fact, [73] formally proved that, for cryptographic accumulators, zero-knowledge is a strictly stronger property than indistinguishability. In other words: every zero-knowledge dynamic universal accumulator is also indistinguishable under the definition of [57], while the opposite is not always true.

Comparison between zero-knowledge and obliviousness. While being really similar at first glance, both properties are actually different in the sense that they require that different elements protect the information about the set: on one hand, a zero-knowledge adversary is given accumulators and witnesses while on the other hand an oblivious adversary is given accumulators and update information.

Therefore there is no relation between zero-knowledge and obliviousness.

Relations between other properties At first, as the adversary is given more and more flexibility, it is easy to see that the following holds: One-Way Domain $\Longrightarrow$ Collision Resistance $\Longrightarrow$ Strong One-Wayness $\Longrightarrow$ One-Wayness, while the opposite is not true.

Regarding undeniability, it has been proven in Appendix C.1 of [57] that every undeniable universal accumulator is collision-resistant. As mentioned in [102], a black-box reduction in the other direction is impossible. In particular, [43] provides a collision-resistant universal accumulator and exhibit an example to show that their scheme is not undeniable.

It remains to make the link between undeniability and one-way domain. At first, we focus on the scheme based on sorted hash tree given in [43]. This one is proven to be universal and collision resistant, and as state before it is not undeniable. It can moreover be used for domain that is in the span of a one-way function. Hence, one-way domain does not imply undeniability. For the opposite, we do not succeed in proving that this is

true or false, and we leave it as an open problem.

In Figure 5.3, we summarize all the above properties and their relation, based on related work, but also on our new results. In the figure an arrow means “implies”, a crossed out arrow means “does not imply” and a dash arrow means “not proven”. Notice that as there is no relation between *obliviousness* and other properties we do not include the former in the figure.

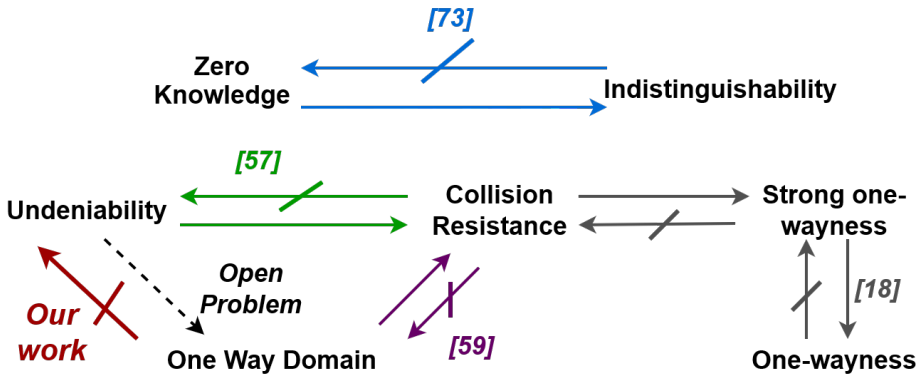


Figure 5.3: Relations between security properties of accumulators.

5.2.3 Discussion About Undeniability

As explained in Section 5.1.2, there are usually three cases regarding the setup: schemes without a trapdoor key, schemes with a trapdoor key without a trusted setup, and eventually schemes with a trapdoor key and a trusted setup. We also need to take into account the fact that the accumulated value can be computed publicly (without knowing the trapdoor) or privately.

In the trapdoor-less setting, as anyone can compute an accumulator, undeniability is a required property. Indeed, an adversary can compute itself an accumulator while the set of pre-image is not none, and not necessarily unique. In the trapdoor setting, when the accumulated value can be computed publicly, or when the setup is in the no-trusted setting, undeniability is needed as any adversary can compute its own (possibly fake) accumulated value. We now focus on the trusted setup setting, and prove, as state in [73], that this is an overkill in terms of security.

In the undeniability security game, when adversary can compute itself accumulated value (or if the setup is untrusted), there is no distinction to make between $x^* \in \mathcal{X}^*$ or

$x^* \notin \mathcal{X}^*$. However, when the only way for the adversary $\mathcal{A}$ to compute acc^* is to request the challenger by giving a set $\mathcal{X}^*$ we need to consider both cases:

- If $x^* \in \mathcal{X}^*$, then $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}^*, x^*, \mathsf{wit}_{x^*}, 0) = 1$ by definition. To win the game, $\mathcal{A}$ must find a non-membership witness wit'_{x^*} such that $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}^*, x^*, \mathsf{wit}'_{x^*}, 1) = 1$. This means that $\mathcal{A}$ wins the undeniability game if it wins the collision resistant game.
- If $x^* \notin \mathcal{X}^*$, then $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}^*, x^*, \mathsf{wit}'_{x^*}, 1) = 1$ by definition. To win the game, $\mathcal{A}$ must find a membership witness such that $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}^*, x^* \mathsf{wit}_{x^*}, 0) = 1$. That means that $\mathcal{A}$ wins the undeniability game if it wins the collision resistance game.

In both cases, collision-resistance is enough, and then undeniability is not required.

5.2.4 Discussion About Delegatable Accumulators

To the best of our knowledge, only one accumulator provides delegatable non-membership proofs: [8]. For their purpose they only consider the delegatable capability for non-membership proofs but it can be defined for membership proofs as well, and will consider both cases in this section. Our goal here is to find a generic way to build accumulators schemes with delegation of proofs.

Accumulators and proof systems. To understand [8]'s construction, let us have a look at their definition of accumulators, given in the extended version of their paper [7]. In their definition an accumulator is constructed from another primitive, called a *proof system*. Briefly, such primitive is a protocol between a prover and a verifier where an honest prover can convince a verifier about the truth of a *statement* with the help of a *witness*, while an adversary cannot convince a verifier of a false statement. The formal definition of proof systems is given below.

Definition 5.2.4 ***Proof System*** *[7]. Let $\mathcal{R}$ be an efficiently computable relation of $(Para, Sta, Wit)$ with setup parameters Para, a statement Sta, and a witness Wit. A non-interactive proof system for $\mathcal{R}$ consists of 3 PPT algorithms: a* Setup*, a prover* Prove*, and a verifier* Verify*. A non-interactive proof system* $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ *must be* complete *and* sound*. Completeness means that for every PPT adversary $\mathcal{A}$,*

$$\left| \Pr \left[\begin{array}{c} Para \leftarrow \mathsf{Setup}(\lambda); (\mathsf{Sta}, \mathsf{Wit}) \leftarrow \mathcal{A}(\mathsf{Para}); \\ Proof \leftarrow \mathsf{Prove}(\mathsf{Para}, \mathsf{Sta}, \mathsf{Wit}) : \\ \mathsf{Verify}(\mathsf{Para}, \mathsf{Sta}, \mathsf{Proof}) = 1 \text{ if } (\mathsf{Para}, \mathsf{Sta}, \mathsf{Wit}) \in \mathcal{R} \end{array} \right] - 1 \right|$$

is negligible. Soundness means that for every PPT adversary $\mathcal{A}$,

$$\left| \Pr \left[\begin{array}{c} \textit{Para} \leftarrow \mathsf{Setup}(\lambda); (\mathsf{Sta}, \mathsf{Proof}) \leftarrow \mathcal{A}(\mathsf{Para}) : \\ \mathsf{Verify}(\mathsf{Para}, \mathsf{Sta}, \mathsf{Proof}) = 0 \text{ if } (\mathsf{Para}, \mathsf{Sta}, \mathsf{Wit}) \notin \mathcal{R}, \forall \mathsf{Wit} \end{array} \right] - 1 \right|$$

is negligible.

We now give two properties of proof systems that we will need in the rest of this section.

Definition 5.2.5 ***Witness indistinguishability****[80, 7]. A proof system is said to satisfy* witness indistinguishability *if the advantage of any malicious verifier $\mathcal{V}$ of winning the security game presented in Figure 5.4 is negligible. Let $\mathcal{P}$ be an honest prover.*

- `SETUP`: $\mathcal{P}$, on input security parameter λ and relation $\mathcal{R}$, runs $\mathsf{Setup}(\lambda)$ to get Para and send it to $\mathcal{V}$ along with $\mathcal{R}$.
- `CHALLENGE`: $\mathcal{V}$ choose a statement Sta along with two witnesses Wit_0 and Wit_1 and sends $(\mathsf{Sta}, \mathsf{Wit}_0, \mathsf{Wit}_1)$ to $\mathcal{P}$. The latter chooses $b \leftarrow \{0, 1\}$ and runs $\mathsf{Prove}(\mathsf{Para}, \mathsf{Sta}, \mathsf{Wit}_b)$ to get Proof_b. $\mathcal{P}$ sends Proof_b to $\mathcal{V}$.
- `GUESS`: $\mathcal{V}$ outputs a guess bit $b^{'} \in \{0, 1\}$ and wins the security game if $b^{'} = b$.

Figure 5.4: Adaptive witness indistinguishability security game for proof systems.

Definition 5.2.6 ***Randomizable proof system*** *[7]. A proof system is said to be* randomizable *if has another PPT algorithm RandProof that takes as input a tuple $(\textit{Para}, \textit{Sta}, \textit{Proof})$ of setup parameters Para, statement Sta and proof Proof and returns another valid proof $\textit{Proof}^{'}$, which is indistinguishable from a proof produced by* Prove*.*

In [8, 7] accumulators are *universal* as the authors require non-membership proofs to build delegatable anonymous credentials. According to their definition, a universal accumulator is composed of a setup algorithm Setup that defines all parameters of the accumulator scheme, an evaluation algorithm Accu that aggregates a large size set of values into a constant size value (the accumulator), a membership proof system $(\mathsf{Setup}, \mathsf{ProveMem}, \mathsf{VerifyMem})$ that proves membership of elements given as input, and a non-membership proof system $(\mathsf{Setup}, \mathsf{ProveNM}, \mathsf{VerifyNM})$ that proves non-membership of elements given as input. In their definition there are also two other PPT algorithms CompMemWit and CompNMWit that take as input the scheme public key, the accumulated set along with its accumulator and an element, and return respectively a witness for membership and non-membership proof for the given element.
As we will see in the next paragraph, proof systems' properties are essential to obtain

delegatable (non-)membership proof. That is why we now rewrite our definition of accumulators (Definition 5.1.1, in Section 5.1) to highlight the underlying proof systems.

Definition 5.2.7 ***Static universal accumulator with proof systems****. We consider* $(\mathsf{Setup}, \mathsf{ProveMem}, \mathsf{VerifyMem})$ *and* $(\mathsf{Setup}, \mathsf{ProveNM}, \mathsf{VerifyNM})$ *respectively a membership and a non-membership proof system along with their associated algorithms* CompMemWit *and* CompNMWit*. A static universal cryptographic accumulator scheme is a tuple of efficient algorithms defined as follows:*

- $\mathsf{Gen}(\lambda, \mathfrak{b})$*: this generation algorithm takes as input a security parameter* λ *and a bound* $\mathfrak{b} \in \mathbb{N} \cup \{\infty\}$ *such that if* $\mathfrak{b} \neq \infty$ *then the number of elements that can be accumulated is bounded by* $\mathfrak{b}$*. It returns a key pair* (sk_{acc}, pk_{acc})*, where* $sk_{acc} = \varnothing$ *if no trapdoor exists and* pk_{acc} *contains the parameter* $\mathfrak{b}$*.*
- $\mathsf{Eval}((\mathsf{sk}_{\mathsf{acc}},)\mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$*: this evaluation algorithm takes as input the accumulator (secret key* sk_{acc} *and) public key* pk_{acc} *and a set* $\mathcal{X}$ *to be accumulated. It returns an accumulator* $acc_{\mathcal{X}}$ *together with some auxiliary information aux.*
- $\mathsf{WitCreate}((\mathsf{sk}_{\mathsf{acc}},)\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}}, \mathsf{aux}, x, \mathsf{Type})$*: this witness creation algorithm takes as input the accumulator (secret key* sk_{acc} *and) public key* pk_{acc}*, an accumulator* $acc_{\mathcal{X}}$*, the associated set* $\mathcal{X}$*, auxiliary information aux, an element* x *and a boolean Type. If Type* $= 0$ *and* $x \in \mathcal{X}$ *it runs* $\mathsf{CompMemWit}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}}, x)$ *to get a membership witness* $mwit_x^{\mathcal{X}}$ *and returns it, if Type* $= 1$ *and* $x \notin \mathcal{X}$ *it run* $\mathsf{CompNMWit}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}}, x)$ *to get a non-membership witness* $nmwit_x^{\mathcal{X}}$ *and returns it.*
- $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{wit}_x^{\mathcal{X}}, x, \mathsf{Type})$*: this verification algorithm takes as input the accumulator public key* pk_{acc}*, an accumulator* $acc_{\mathcal{X}}$*, a witness* $wit_x^{\mathcal{X}}$*, an element* x *and a boolean Type. If* $wit_x^{\mathcal{X}} = mwit_x^{\mathcal{X}}$*, and Type* $= 0$ *it runs* $\mathsf{ProveMem}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{mwit}_x^{\mathcal{X}})$ *to get ProofMem and returns the output of* $\mathsf{VerifyMem}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}\mathcal{X}, \mathsf{ProofMem})$*. If* $wit_x^{\mathcal{X}} = nmwit_x^{\mathcal{X}}$*, and Type* $= 1$ *it runs* $\mathsf{ProveNM}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{nmwit}_x^{\mathcal{X}})$ *to get ProofNM and returns the output of* $\mathsf{VerifyNM}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}\mathcal{X}, \mathsf{ProofNM})$*. Otherwise it returns* 0*.*

In the sequel we will refer to accumulators in the sense of Definition 5.2.7.

Delegatable (non-)membership proofs. Now we formally define the *delegatable proofs* property, adapted from [8] and for both membership and non-membership proofs. In the following, $\mathcal{D}$ denotes the space of values to be accumulated.

Definition 5.2.8 ***Delegatable proofs*** *. An accumulator allows delegatable proofs if it additionally provides the following algorithms.*

- $\mathsf{Dele}(\mathsf{pk}_{\mathsf{acc}}, x, \mathsf{Type})$*: the delegation algorithm takes as input the public key* pk_{acc}*, an element* x *and boolean parameter Type. It outputs a delegating key* Del_y *and auxiliary information* $req = Type$*.*
- $\mathsf{Vali}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{Del}_y, \mathsf{req})$*: the validation algorithm takes as input the public key* pk_{acc}*, a delegating key* Del_y *and auxiliary information req. If* Del_y *is valid it returns* 1*, otherwise it returns 0.*
- $\mathsf{Rede}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{Del}_y, \mathsf{req})$*: the re-delegation algorithm takes as input the public key* pk_{acc}*, a delegating key* Del_y *and auxiliary information req. If* $\mathsf{Vali}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{Del}_y, \mathsf{req}) = 1$*, the algorithm returns an other delegating key* Del'_x *and auxiliary information req.*
- $\mathsf{CompProof}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{Del}_y, \mathsf{req}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}})$*: the proof computation algorithm takes as input the public key* pk_{acc}*, a delegating key* Del_y *and auxiliary information req, a set* $\mathcal{X}$ *and the associated accumulated value* $acc_{\mathcal{X}}$*. If* $x, \mathcal{X} \models req$ *it returns a proof (similar to those output by* ProveMem *and* ProveNM*) according to req,* $\mathcal{X}$ *and* x*.*

These algorithms verify, for every PPT algorithms $\mathcal{A}, \mathcal{A}_1, \mathcal{A}_2$*:*

- *Delegability: the following is negligible*

$$\left| \Pr \left[\begin{array}{c} (sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda); \\ (x, Type, \mathcal{X}) \leftarrow \mathcal{A}_1(pk_{acc}); acc_{\mathcal{X}} \leftarrow \mathsf{Eval}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}); \\ wit_x \leftarrow \mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}}, x, \mathsf{Type}); \\ Proof_0 \leftarrow \mathsf{Prove}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{aux}, \mathsf{wit}_x); \\ (Del_y, req) \leftarrow \mathsf{Dele}(\mathsf{pk}_{\mathsf{acc}}, x, \mathsf{Type}); \\ Proof_1 \leftarrow \mathsf{CompProof}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{Del}_y, \mathsf{req}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}}); \\ b \leftarrow \{0,1\}; b' \leftarrow \mathcal{A}_2(acc_{\mathcal{X}}, wit_x, Del_y, Proof_b) : \\ (x, \mathcal{X} \models req) \wedge b = b' \end{array} \right] - \frac{1}{2} \right|$$

- *Unlinkability: the following is negligible*

$$\left| \Pr \left[\begin{array}{c} (sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda); \\ (y_0, y_1) \leftarrow \mathcal{D}; Type \leftarrow \mathcal{A}, \\ (Del_y, req) \leftarrow \mathsf{Dele}(\mathsf{pk}_{\mathsf{acc}}, y_0, \mathsf{Type}); \\ b \leftarrow \{0,1\}; \\ (Del_{y_b}, req_b) \leftarrow \mathsf{Dele}(\mathsf{pk}_{\mathsf{acc}}, y_b, \mathsf{Type}); \\ b' \leftarrow \mathcal{A}(pk_{acc}, Del_y, Del_{y_b}) : \\ b = b' \end{array} \right] - \frac{1}{2} \right|$$

- *Redelegability: the following is negligible*

$$\left| \Pr \left[\begin{array}{c} (sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda); (\mathsf{Type}, x) \leftarrow \mathcal{A}_1(\mathsf{pk_{acc}}); \\ (Del_y, req) \leftarrow \mathsf{Dele}(\mathsf{pk_{acc}}), x, \mathsf{Type}); \\ (Del_y^0, req^0) \leftarrow \mathsf{Dele}(\mathsf{pk_{acc}}), x, \mathsf{Type}); \\ (Del_y^1, req^1) \leftarrow \mathsf{Rede}(\mathsf{pk_{acc}}, \mathsf{Del}_y, \mathsf{req}); \\ b \leftarrow \{0,1\}; b' \leftarrow \mathcal{A}_2(pk_{acc}, Del_y, Del_y^b) : \\ b = b' \end{array} \right] - \frac{1}{2} \right|$$

- *Verifiability: the following is negligible*

$$\left| \Pr \left[\begin{array}{c} (sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda); (\mathsf{Type}, x) \leftarrow \mathcal{A}(\mathsf{pk_{acc}}); \\ (Del_y, req) \leftarrow \mathsf{Dele}(\mathsf{pk_{acc}}, x, \mathsf{Type}) : \\ \mathsf{Vali}(\mathsf{pk_{acc}}, \mathsf{Del}_y, \mathsf{req}) = 1 \text{ if } x \in \mathcal{D} \end{array} \right] - 1 \right|$$

and

$$\left| \Pr \left[\begin{array}{c} (sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda); \mathsf{Del}' \leftarrow \mathcal{A}(\mathsf{pk_{acc}}) : \\ \mathsf{Vali}(\mathsf{pk_{acc}}, \mathsf{Del}') = 0 \\ \textit{if } Del' \notin \left\{ Del \;\middle|\; \begin{array}{c} Del \leftarrow \mathsf{Dele}(\mathsf{pk_{acc}}, x', \mathsf{Type}); \\ x' \in \mathcal{D} \end{array} \right\} \end{array} \right] - 1 \right|,$$

where the condition $Del' \notin \{Del | Del \leftarrow \mathsf{Dele}(\mathsf{pk_{acc}}, x', \mathsf{Type}); x' \in \mathcal{D}\}$ means that the delegation key Del' does not correspond to a delegation key correctly computed, for any element x' of the domain $\mathcal{D}$.

Delegatable (non-)membership proofs and proof systems. Let us see that the additional algorithms Dele, Rede, Vali, CompProof required to obtain an accumulator with delegatable (non-)membership proofs can be rewrite to highlight the underlying proof systems algorithms. As we are dealing with both kinds of proofs, we use (Setup, Prove, PSVerify) to denote a general proof system, representing either the membership or the non-membership proof system. The output of Prove is denoted Proof. The setup algorithm Setup of the proof system is run by the accumulator generation algorithm Gen thus Para the setup parameters of the proof system are included in $\mathsf{pk_{acc}}$. Let the proof system be randomizable, i.e. it has another algorithm RandProof that takes as input a proof and outputs a randomized one.

- Dele($\mathsf{pk}_{\mathsf{acc}}, x, \mathsf{Type}$) : the algorithm creates a proof system statement Sta from the public parameters, a proof system witness Wit for x, Type and computes a proof system proof Proof_0 from Prove($\mathsf{Para}, \mathsf{Sta}, \mathsf{Wit}$). It outputs Proof and $\mathsf{req} = \{\mathsf{Type}, \mathsf{Sta}\}$.
- Vali($\mathsf{pk}_{\mathsf{acc}}, \mathsf{Proof}, \mathsf{req}$) : if the verification algorithm of PSVerify on inputs ($\mathsf{Para}, \mathsf{Sta}$, Proof outputs 1, the algorithm outputs 1 otherwise it outputs 0.
- Rede($\mathsf{pk}_{\mathsf{acc}}, \mathsf{Proof}, \mathsf{req}$): if Vali($\mathsf{pk}_{\mathsf{acc}}, \mathsf{Proof}, \mathsf{req}$) $= 1$, it runs RandProof($\mathsf{pk}_{\mathsf{acc}}, \mathsf{Sta}$, Proof) to get a randomized proof $\mathsf{Proof}^{'}$.
- CompProof($\mathsf{pk}_{\mathsf{acc}}, \mathsf{Proof}_y, \mathsf{req}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}}, $): if $\mathcal{X}, x \models \mathsf{req}$, where x is the witness used to create Proof_y, it uses homomorphic property of the proof system to obtain a new proof $\mathsf{Proof}^{'}_x$, computed from and from the underlying witness x and the new statement $\mathsf{Sta}^{'}$, corresponding to $\mathcal{X}$ and $\mathsf{acc}_{\mathcal{X}}$.

We now prove that Unlinkability, Redelegability and Verifiability properties are satisfied if the underlying proof system is witness indistinguishable and randomizable.

Lemma 5.2.1 *Unlinkability is satisfied thanks to the witness indistinguishability property of the (non-)membership proof system.*

Proof 5.2.2 *We rewrite the unlinkability property, using the (non-)membership proof system, which gives us:*

$$\Pr\left[\begin{array}{c} (\mathit{sk}_{\mathit{acc}}, \mathit{pk}_{\mathit{acc}}) \leftarrow \mathsf{Gen}(1\lambda); \\ (\mathit{Wit}_0, \mathit{Wit}_1) \leftarrow \mathcal{D}; \mathit{Sta} \leftarrow \mathcal{A}, \\ \mathit{Proof}_0 \leftarrow \mathsf{Prove}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{Wit}_0, \mathsf{Sta}); \\ b \leftarrow \{0,1\}\,; \mathit{Proof}_b \leftarrow \mathsf{Prove}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{Wit}_b, \mathsf{Sta}); \\ b^{'} \leftarrow \mathcal{A}(\mathit{pk}_{\mathit{acc}}, \mathit{Proof}_0, \mathit{Proof}_b) : \\ b = b^{'} \end{array}\right]$$

This corresponds to the probability in the witness indistinguishability security game of the (non-)membership proof system, which is negligible. Thus, so is the probability defined in Unlinkability. □

Lemma 5.2.2 *Redelegability is satisfied thanks to the randomizable property of the (non-)membership proof system.*

Proof 5.2.3 *We rewrite the redelegability property, using the (non-)membership proof system's algorithms, which gives us*

$$\Pr\left[\begin{array}{c}(sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda);\\ Sta, Wit \leftarrow \mathcal{A}_1(pk_{acc});\\ Proof \leftarrow \mathsf{Prove}(\mathsf{pk_{acc}}, \mathsf{Wit}, \mathsf{Sta});\\ Proof_0 \leftarrow \mathsf{Prove}(\mathsf{pk_{acc}}, \mathsf{Wit}, \mathsf{Sta});\\ Proof_1 \leftarrow RandProof(pk_{acc}, Sta, Proof);\\ b \leftarrow \{0,1\}; b' \leftarrow \mathcal{A}_2(pk_{acc}, Proof, Proof_b):\\ b = b'\end{array}\right]$$

This corresponds to the probability in the randomized security game of the (non-)membership proof system, which is negligible. Thus, so is the probability defined in Redelegability. □

Lemma 5.2.3 *Verifiability is satisfied thanks to the completeness and soundness of the (non-)membership proof system.*

Proof 5.2.4 *We rewrite the verifiability property, using the (non-)membership proof system's algorithms, which gives us*

$$\Pr\left[\begin{array}{c}(sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda); (\mathsf{Sta}, \mathsf{Wit}) \leftarrow \mathcal{A}(\mathsf{pk_{acc}});\\ Proof \leftarrow \mathsf{Prove}(\mathsf{pk_{acc}}, \mathsf{Sta}, \mathsf{Wit}):\\ \mathsf{PSVerify}(\mathsf{pk_{acc}}, \mathsf{Sta}, \mathsf{Proof}) = 1 \text{ if } (\mathsf{pk_{acc}}, \mathsf{Sta}, \mathsf{Proof}) \in R\end{array}\right]$$

and

$$\Pr\left[\begin{array}{c}(sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda); (\mathsf{Sta}, \mathsf{Proof}) \leftarrow \mathcal{A}(\mathsf{pk_{acc}}):\\ \mathsf{PSVerify}(\mathsf{pk_{acc}}, \mathsf{Sta}, \mathsf{Proof}) = 0 \text{ if } (\mathsf{pk_{acc}}, \mathsf{Sta}, \mathsf{Proof}) \notin R, \forall t_t\end{array}\right]$$

This corresponds to the soundness probability of the proof system, and is overwhelming. Thus so is the second probability of Verifiability. □

Note 5.2.2 *In [7], they proved that their accumulator satisfies unlinkability as they used a composable ZK proof system. Actually, only witness indistinguishability is required.*

How to obtain delegability? The witness indistinguishability and randomizable property of proof systems are not enough to obtain an accumulator with delegatable (non-)membership proof as *delegability* cannot be proven. To solve this issue, [8] uses a primitive they introduced: homomorphic proofs.

Definition 5.2.9 ***Homomorphic proofs*** *[8]. Let* $(\mathsf{PSSetup}, \mathsf{Prove}, \mathsf{PSVerify})$ *be a proof system for a relation* R *and* $\mathsf{Para} \leftarrow \mathsf{PSSetup}(\lambda)$*. Consider a subset* Π *of all* $(\mathit{Sta}, \mathit{Wit}, \mathit{Proof})$ *such that* $(\mathit{Para}, \mathit{Sta}, \mathit{Wit}) \in R$ *and* $\mathsf{PSVerify}(\mathsf{Para}, \mathsf{Sta}, \mathit{Proof}) = 1$*, and an operation* $+_\Pi : \Pi \times \Pi \to \Pi$*.* Π *is a set of homomorphic proofs if* $(\Pi, +_\Pi)$ *satisfies* ***closure, associativity*** *and* ***commutativity****. Consider an* $I_\Pi = (\mathit{Sta}_0, \mathit{Wit}_0, \mathit{Proof}_0) \in \Pi$*.* Π *is a set of strongly homomorphic proofs if* $(\Pi, +_\Pi, I_\Pi)$ *forms an Abelian group where* I_Π *is the identity element.*

Note 5.2.3 *As stated in [8], one can randomize a proof computed from the homomorphic operation to get another proof, indistinguishable from a proof generated by* Prove*.*

Lemma 5.2.4 *Delegability is satisfied thanks to the homomorphic proofs and the randomizable property of the (non-)membership proof system.*

Proof 5.2.5 *We rewrite the delegability property, using the (non-)membership proof system which gives us:*

$$\Pr\left[\begin{array}{c} (\mathit{sk}_{acc}, \mathit{pk}_{acc}) \leftarrow \mathsf{Gen}(\lambda); \\ (\mathit{Sta}_0, \mathit{Sta}, \mathit{Wit}) \leftarrow \mathcal{A}_1(\mathit{pk}_{acc}); \\ \mathit{Proof}_0 \leftarrow \mathsf{Prove}(\mathsf{pk}_{acc}, \mathsf{Sta}, \mathsf{Wit}); \\ \mathit{Proof} \leftarrow \mathsf{Prove}(\mathsf{pk}_{acc}, \mathsf{Sta}_0, \mathsf{Wit}); \\ \mathit{Proof}_1 \leftarrow \mathit{RandProof}(\mathit{pk}_{acc}, +_\Pi(\mathit{Proof})); \\ b \leftarrow \{0,1\}; b' \leftarrow \mathcal{A}_2(\mathit{Sta}, \mathit{Wit}, \mathit{Proof}, \mathit{Proof}_b) : \\ (\mathit{Sta}, \mathit{Wit}, \mathit{Proof}_b) \in R \wedge b = b' \end{array}\right]$$

Thanks to the homomorphic proofs and randomizable property of the (non-)membership proof system, this probability is negligible, so is the probability in Delegability. □

Conclusion. Combining the above paragraphs we obtain the following theorem.

Theorem 5.2.2 *If the membership (resp. non-membership) proof system is witness indistinguishable, randomizable and has homomorphic proofs then the accumulator satisfies Delegability, Unlinkability, Redelegability and Verifiability for membership (resp. non-membership) proofs.*

Therefore we can conclude with the following way to obtain an accumulator with delegatable (non-)membership proofs:

1. Find the underlying (non-)membership prove system
2. If the proof system is not witness indistinguishable or randomizable or does not have homomorphic proofs, change it to obtain the required properties.

Applying the above on an example. We tried to apply the above transformation to the accumulator scheme of [73]. To do so, we used Groth Sahai (GS) proof system [81] as, as proven by [7], GS proof system [81] satisfies all required properties to obtain a delegatable accumulator (i.e. it produces homomorphic proofs). Thus finding the correct GS statement for membership will lead to delegatable membership proofs, and the same goes for non-membership proofs. Unfortunately we were only able to obtain delegation for membership proofs but not for non-membership proofs. We leave this question as an open problem.

5.2.5 Discussion on Accumulator Applications

Originally, accumulators were used for timestamping and membership testing [27] but over time their usage become multiple: fail stop signatures [18], membership revocation in group signature [46], ID based ring signatures [113], anonymous credentials (delegatable) [8] or attribute-based anonymous credentials [14]), distributed public key infrastructure [125], e-cash [15]. Refer to several surveys on a cryptographic accumulator, such as [123], for details on accumulators' applications.

We wanted to draw attention on one surprising thing: while the purpose of cryptographic accumulators is to make constant the size of cryptographic objects, few attempts have been done to use them for encryption schemes, such as [11, 72, 135]. The works of [72, 11] propose broadcast encryption schemes that use (RSA based) cryptographic accumulator, to manage users' secret keys: let $N \in \mathbb{N}$ be the number of users in the broadcast encryption scheme. For an index $i \in [N]$, a secret key is created for each subset of $[N]$ that contains i and the secret key of user identified by index i is an accumulator of the secret keys created for the subsets. To decrypt a message, user "extract" (with a process similar to the generation of a witness) the secret key corresponding to the subset associated to the ciphertext. More recently, Wang and Chow [135] present an identity-based broadcast encryption scheme that uses a degenerated notion of accumulators, composed only of algorithms Gen and Eval. In their scheme, the evaluation algorithm Eval is used during encryption to hide some randomness (the latter being used to mask the message) and the compactness of its output is required for the scheme efficiency. However they do not take into account the other functionality of cryptographic accumulators, the efficient membership proof, in their scheme. Also

notice that some works propose to use accumulators to add revocation functionality previously existing encryption scheme, such as [83] who adds revocation to Lewko and Water's hierarchical identity-based encryption scheme [96].

In Section 6.2 we proposed an encryption scheme that uses cryptographic accumulator for both key management and encryption. While our scheme suffers from large public key size, it opens a door to a new field of works that build encryption scheme from accumulator.

5.3 Our Contributions to Accumulators' Formalism

We now present two of our contributions regarding cryptographic accumulators with *private evaluation* and *public witness generation*. The first one is a new security property, called *unforgeability of private evaluation*, that protects the accumulator itself when the latter is computed using the secret key of the scheme. Our second contribution is a new kind of accumulators, called *dually computable accumulators*, that has two evaluation algorithms: Eval that takes as input the secret key of the scheme, and PublicEval that takes as input the public key of the scheme.

5.3.1 New Security Property of Unforgeability of Private Evaluation

Surprisingly when looking at all security properties presented in Sections 5.1.2 and 5.2.2 we notice that there is no property that protect the accumulator itself, especially when the latter computation requires the knowledge of the scheme's secret key. In the undeniability property the accumulator might be forged but only in order to help forging both membership and non-membership witnesses for the same element. In some applications, the accumulator itself can be an important data to protect: for example when the accumulator represents a revocation list, having a property proving that only the trusted authority can compute this revocation list will improve the general security of the protocol. This is an important gap in cryptographic accumulator schemes security.

We fill this gap by providing the following property that states that it must be hard to "forge" a privately computed accumulator that passes the verification algorithm with an honestly computed witness.

Definition 5.3.1 ***Unforgeability of private evaluation*** *(UPE). An accumulator scheme with private evaluation is said to satisfy* unforgeability of private evaluation *if all PPT adversaries $\mathcal{A}$ have at most negligible advantage in the game presented in Figure 5.5, where $\mathcal{A}$'s advantage is defined as*

$$\mathsf{Adv}^{\mathsf{UPE}}_{\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{c} \mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}^*, x, \mathsf{wit}_x) = 1 : \\ (\mathit{sk}_{\mathit{acc}}, \mathit{pk}_{\mathit{acc}}) \leftarrow \mathsf{Gen}(\lambda), \\ (\mathcal{X}^*, \mathit{acc}^*) \leftarrow \mathcal{A}(\mathit{pk}_{\mathit{acc}}); \\ \mathit{acc}_{\mathcal{X}^*} \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{acc}}, \mathcal{X}^*), x \leftarrow \mathcal{X}^*; \\ \mathit{wit}_x \leftarrow \mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}^*}, \mathcal{X}^*, x) \end{array}\right].$$

Let $\mathcal{C}$ be a challenger.

SETUP: on input λ, $\mathcal{C}$ runs $\mathsf{Gen}(\lambda)$ to get $(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}})$ and sends $\mathsf{pk}_{\mathsf{acc}}$ to $\mathcal{A}$.
GUESS: $\mathcal{A}$ returns a set $\mathcal{X}^*$ and a forged accumulator acc^* and wins if for any $x \in \mathcal{X}^*$, $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}^*, x, \mathsf{wit}_x) = 1$ where $\mathsf{wit}_x \leftarrow \mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}^*}, \mathcal{X}^*, x)$ and $\mathsf{acc}_{\mathcal{X}^*} \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{acc}}, \mathcal{X}^*)$.

Figure 5.5: Unforgeability of private evaluation security game.

5.3.2 Introducing Dually Computable Accumulators

We now introduce a new functionality for accumulators with private evaluation and public witness generation: *dually computable*. Informally, a dually computable accumulator is a cryptographic accumulator scheme with an additional evaluation algorithm PublicEval that uses solely the scheme public key.

Definition 5.3.2 ***Dually computable accumulator.*** *Starting from a static, universal accumulator Acc $= (\mathsf{Gen}, \mathsf{Eval}, \mathsf{WitCreate}, \mathsf{Verify})$, we say that Acc is* dually computable *if it also provides two algorithms* PublicEval *and* PublicVerify *such that:*

- $\mathsf{PublicEval}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$*: this evaluation algorithm takes as input the accumulator public key $\mathit{pk}_{\mathit{acc}}$ and a set $\mathcal{X}$. It outputs an accumulator $\mathit{accp}_{\mathcal{X}}$ of $\mathcal{X}$ and auxiliary information auxp.*
- $\mathsf{PublicVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}}, \mathsf{wit}_x, x, \mathsf{Type})$*: this verification algorithm takes as input the accumulator public key $\mathit{pk}_{\mathit{acc}}$, a publicly computed accumulator $\mathit{accp}_{\mathcal{X}}$ of $\mathcal{X}$, an element x, a witness wit_x for x and boolean Type, computed from* $\mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{accp}_{\mathcal{X}}, \mathit{auxp}, x, \mathit{Type})$*. If wit_x is a correct membership witness (*i.e. $x \in \mathcal{X}$*) and Type $= 0$, or if wit_x is a correct non-membership witness (*i.e.

$x \notin \mathcal{X}$) *and* Type $= 1$ *then the algorithm outputs* 1*, otherwise it outputs* 0*.*

Note 5.3.1 *When the algorithm* WitCreate *is run with a publicly computed accumulator (accp) instead of a privately computed one(acc), the output witness will be written witp.*

A dually computable accumulator must satisfy four properties: *correctness* and *dual collision resistance* as any cryptographic accumulator, *correctness of duality* and *distinguishability* two properties that we introduced. In the correctness and collision resistance definitions, we highlight in blue the parts related to PublicEval and PublicVerify. Without these parts, the definitions are exactly as Definitions 5.1.2 and 5.1.3.

Definition 5.3.3 ***Correctness.*** *A dually computable accumulator is said to be* correct *if for all security parameters λ, all integer $\mathfrak{b} \in \mathbb{N} \cup \{\infty\}$, all set of values $\mathcal{X}$, all boolean type and all element x such that if* Type $= 0$ *then $x \in \mathcal{X}$ and if* Type $= 1$ *then $x \notin \mathcal{X}$ the following holds*

$$\Pr\left[\begin{array}{c} (sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda, \mathfrak{b}), \\ (acc_{\mathcal{X}}, aux) \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}, \mathcal{X}), \\ (accp_{\mathcal{X}}, auxp) \leftarrow \mathsf{PublicEval}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}), \\ wit_x \leftarrow \mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}}, \mathsf{aux}, x, \mathsf{Type}), \\ witp_x \leftarrow \mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{accp}_{\mathcal{X}}, \mathsf{auxp}, x, \mathsf{Type}) : \\ \mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{wit}_x, x, \mathsf{Type}) = 1 \\ \wedge \mathsf{PublicVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}}, \mathsf{witp}_x, x, \mathsf{Type}) = 1 \end{array}\right] = 1$$

The following property states that a witness computed for a privately (resp. publicly) computed accumulator as input of the WitCreate algorithm must pass the PublicVerify (resp. Verify) algorithm, with publicly (resp. privately) computed accumulator for the same set as the privately (resp. publicly) computed accumulator.

Definition 5.3.4 ***Correctness of duality.*** *A dually computable accumulator is said to satisfy* correctness of duality *if for all security parameters λ, all integer $\mathfrak{b} \in \mathbb{N} \cup \{\infty\}$, all set of values $\mathcal{X}$ all boolean type and all element x such that if* Type $= 0$ *then $x \in \mathcal{X}$ and*

if Type $= 1$ *then* $x \notin \mathcal{X}$ *the following holds*

$$\Pr\left[\begin{array}{c}(sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda, \mathfrak{b}),\\ (acc_{\mathcal{X}}, aux) \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}, \mathcal{X}),\\ (accp_{\mathcal{X}}, auxp) \leftarrow \mathsf{PublicEval}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}),\\ wit_x \leftarrow \mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{acc}_{\mathcal{X}}, \mathsf{aux}, x, \mathsf{Type}),\\ witp_x \leftarrow \mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}, \mathsf{accp}_{\mathcal{X}}, \mathsf{auxp}, x, \mathsf{Type}) :\\ (\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{witp}_x, x, \mathsf{Type}) = 1)\\ \wedge(\mathsf{PublicVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}}, \mathsf{wit}_x, x, \mathsf{Type}) = 1)\end{array}\right] = 1$$

Definition 5.3.5 ***Distinguishability.*** *A dually computable accumulator satisfies* distinguishability*, if for any security parameter* λ *and integer* $\mathfrak{b} \in \mathbb{N} \cup \{\infty\}$*, any keys* (sk_{acc}, pk_{acc}) *generated by* $\mathsf{Gen}(\lambda, \mathfrak{b})$*, and any set* $\mathcal{X}$*:* $acc_{\mathcal{X}} \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$ *and* $accp_{\mathcal{X}} \leftarrow \mathsf{PublicEval}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$ *are trivially distinguishable.*

Definition 5.3.6 ***Dual collision resistance*** *(DCR). A dually computable accumulator is said to satisfy* dual collision resistance *if all PPT adversaries* $\mathcal{A}$ *have at most negligible advantage in the game presented in Figure 5.6, where* $\mathcal{A}$*'s advantage is defined as*

$$\mathsf{Adv}^{DCR}_{\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{c}\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 0) = 1 \wedge x^* \notin \mathcal{X}^*\\ \vee(\mathsf{PublicVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 0) = 1 \wedge x^* \notin \mathcal{X}^*)\\ \vee\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 1) = 1 \wedge x^* \in \mathcal{X}^*\\ \vee(\mathsf{PublicVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 1) = 1 \wedge x^* \in \mathcal{X}^*) :\\ (sk_{acc}, pk_{acc}) \leftarrow \mathsf{Gen}(\lambda, \mathfrak{b}), (\mathcal{X}^*, \mathsf{wit}_{x^*}, x^*) \leftarrow \mathcal{A}(\mathsf{pk}_{\mathsf{acc}}),\\ acc_{\mathcal{X}^*} \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{acc}}, \mathcal{X}^*),\\ accp_{\mathcal{X}^*} \leftarrow \mathsf{PublicEval}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}^*)\end{array}\right]$$

Let $\mathcal{C}$ *be a challenger.*

Note 5.3.2 *Notice that in the above security game the adversary does not query the challenger for witnesses as witness generation is done publicly in a dually computable accumulator.*

5.4 Another Contribution: Our New Accumulators Schemes

We now present our two new cryptographic accumulators schemes. Our first scheme improves the state of the art by being the first accumulator scheme with private eval-

SETUP: on input λ, $\mathcal{C}$ runs Gen(λ) to get $(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}})$ and sends $\mathsf{pk}_{\mathsf{acc}}$ to $\mathcal{A}$.
ACCUMULATOR QUERY: $\mathcal{A}$ chooses a set $\mathcal{X}$ and sends it to $\mathcal{C}$. $\mathcal{C}$ returns to $\mathcal{A}$ $\mathsf{acc}_{\mathcal{X}}$, where $\mathsf{acc}_{\mathcal{X}} \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{acc}}, \mathcal{X})$.
GUESS: $\mathcal{A}$ returns a set $\mathcal{X}^*$, an element x^* and a witness wit_{x^*} and wins if $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 0) = 1 \wedge x^* \notin \mathcal{X}^*$ or $\mathsf{PublicVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 0 = 1 \wedge x^* \notin \mathcal{X}^*)$ or $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 1) = 1 \wedge x^* \in \mathcal{X}^*$ or $\mathsf{PublicVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}^*}, \mathsf{wit}_{x^*}, x^*, \mathsf{Type} = 1) = 1 \wedge x^* \in \mathcal{X}^*$.
ACCUMULATOR QUERY: $\mathcal{A}$ can continue to query accumulator for sets $\mathcal{X}$.

Figure 5.6: Collision resistance security game for dually computable accumulators.

uation while having public witness generation. Our second scheme is the first dually computable accumulator in the literature and is based on our first scheme.

5.4.1 Our Universal Accumulator with Private Evaluation and Public Witness Generation

Our first construction fills the gap raised in Note 5.1.5 and is a unique combination of dual pairing vector spaces (Definitions 2.3.1, 2.3.2) and the accumulator of Nguyen [113] that we present below.

Nguyen's accumulator. In 2005, Nguyen proposed the first cryptographic accumulator based on bilinear pairing. His scheme is bounded by $q \in \mathbb{N}$ and we present it in Figure 5.7.

- Gen(λ, q): run a bilinear group generation algorithm to get $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ and choose a random $s \leftarrow \mathbb{Z}_p^*$. Set $\mathsf{sk}_{\mathsf{acc}} = s$ and $\mathsf{pk}_{\mathsf{acc}} = g_1, g_1^s, \cdots, g_1^{s^q}, g_2, g_2^s, \cdots, g_2^{s^q}$.
- $\mathsf{Eval}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$: compute the coefficients $\{a_i\}_{i=0,\cdots,q}$ of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (Z + x)$. Then compute $\mathsf{acc}_{\mathcal{X}} = g_1^{\sum_{i=0}^{q} a_i s^i}$, and return $\mathsf{acc}_{\mathcal{X}}$.
- $\mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathcal{X}, \underline{x})$: let $\{b_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{X}\setminus\{\underline{x}\}}[Z] = \prod_{x \in \mathcal{X}, x \neq \underline{x}} (x + Z)$. Compute $\mathsf{wit}_{\underline{x}} = g_2^{\sum_{i=0}^{q} b_i s^i}$, and return $\mathsf{wit}_{\underline{x}}$.
- $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{wit}_{\underline{x}}, \underline{x})$: return 1 if $e(\mathsf{acc}_{\mathcal{X}}, g_2) = e(g_1^{\underline{x}} \cdot g_1^s, \mathsf{wit}_{\underline{x}})$, 0 otherwise.

Figure 5.7: Nguyen's [113] cryptographic accumulator scheme.

Correctness of the Nguyen's scheme can easily be shown: let $\mathcal{X}$ be a set and $\underline{x}$ be an element of $\mathcal{X}$. As $\underline{x} \in \mathcal{X}$, we have that $\sum_{i=0}^{q} a_i s^i = (\underline{x} + s) \cdot \sum_{i=0}^{q} b_i s^i$ and thus

$$\begin{aligned} e(g_1^{\underline{x}} \cdot g_1^s, \mathsf{wit}_{\underline{x}}) &= e(g_1^{(\underline{x}+s)}, g_2^{\sum_{i=0}^{q} b_i s^i}) &= e(g_1, g_2)^{(\underline{x}+s)\sum_{i=0}^{q} b_i s^i} \\ & & = e(g_1, g_2)^{\sum_{i=0}^{q} a_i s^i} \\ & & = e(\mathsf{acc}_{\mathcal{X}}, g_2). \end{aligned}$$

Regarding security, the scheme was proven to be *collision resistant* under the q-SBDH problem (Definition 2.2.12).

Note 5.4.1 *Nguyen's accumulator was originally in the symmetric bilinear setting. We present it in the asymmetric bilinear setting to make the comparison with our accumulator scheme easier. Notice that the choice of representing accumulators with elements of $\mathbb{G}_1$ and witnesses with elements of $\mathbb{G}_2$ is ours and groups can easily be swapped.*

Note 5.4.2 *Notice that [113]'s accumulator supports subset queries and multiset settings as characteristic polynomial Ch can be defined for a multiset and for the set $\mathcal{X} \setminus \mathcal{I}$, where $\mathcal{I} \subset \mathcal{X}$.*

Originally, Nguyen [113]'s accumulator was *non-universal*. Later, Damgard *et al.* [55] and Au *et al.* [14] proposed a way to compute non-membership witnesses: while membership witnesses use the fact that the polynomial $(\underline{x} + Z)$ divides $\mathsf{Ch}_{\mathcal{X}}[Z]$ if $\underline{x} \in \mathcal{X}$, meaning that there exits a polynomial $Q[Z]$ such that $\mathsf{Ch}_{\mathcal{X}}[Z] = (\underline{x} + Z) \cdot Q[Z]$, they decided to exploit the fact that if $\underline{x} \notin \mathcal{X}$, then there exists an integer r such that $\mathsf{Ch}_{\mathcal{X}}[Z] = (\underline{x} + Z) \cdot Q[Z] + r$. Thus the non-membership witness is composed of $g_2^{Q(s)}$ (which is the same element than in membership witness) and g_2^r. Verification is done by checking if $e(\mathsf{acc}_{\mathcal{X}}, g_2) = e(g_1^{\underline{x}} \cdot g_1^s, g_2^{Q(s)}) \cdot e(g_1, g_2^r)$. Unfortunately, recently Biryukov *et al.* [28] proved that this way to compute non-membership witness breaks the collision resistance property of the scheme.
For this reason (and as they wanted to reach another level of privacy security), Ghosh *et al.* [73] defined non-membership witnesses with Bezout coefficients: let $\mathcal{X}$ be a set and $\underline{x} \notin \mathcal{X}$. As $\underline{x} \notin \mathcal{X}$, the $\gcd$ of $\mathsf{Ch}_{\mathcal{X}}[Z]$ and $(\underline{x} + Z)$ is 1. Then with the Extended Euclidean algorithm, compute the polynomials $q_1[Z], q_2[Z]$ (the Bezout coefficients) such that $\mathsf{Ch}_{\mathcal{X}} \cdot q_1[Z] + q_2[Z] \cdot (\underline{x} + Z) = 1$. The non-membership witness is composed of $g_2^{q_1(s)}$ and $g_2^{q_2(s)}$ and verification is done by checking if $e(\mathsf{acc}_{\mathcal{X}}, g_2^{q_1(s)}) \cdot e(g_1^{\underline{x}} \cdot g_1^s, g_2^{q_2(s)}) = e(g_1, g_2)$.
We will use their idea to make our scheme universal.

Intuition. Let us now present the intuition of our construction. It is easy to see that with Nguyen's accumulator, both evaluation and witness generation are either done publicly or privately but it is not possible to have one algorithm executed privately while the other is executed publicly. Indeed, a trivial idea to have private evaluation and public witness creation would be to keep secret the elements of $\mathbb{G}_1$ for the private evaluation and use the elements of $\mathbb{G}_2$ as public elements. But this does not work as g_1^s must be given publicly for verification.
That is why we use dual pairing vector spaces (see Section 2.3), with dimension $n = 2$. Let $\mathbb{D}, \mathbb{D}^*$ be two dual orthonormal bases such that $\mathbb{D} = (\boldsymbol{d}_1, \boldsymbol{d}_2)$ and $\mathbb{D}^* = (\boldsymbol{d}_1^*, \boldsymbol{d}_2^*)$. By playing with the bases $\boldsymbol{d}_1, \boldsymbol{d}_1^*, \boldsymbol{d}_2$ and $\boldsymbol{d}_2^*$, we can keep secret some elements and publish some others as follows:

- $g_1^{\boldsymbol{d}_1}, g_1^{\boldsymbol{d}_1 s}, \cdots g_1^{\boldsymbol{d}_1 s^q}$ are not publicly given since used for private evaluation;
- $g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_1^* s}, \cdots, g_2^{\boldsymbol{d}_1^* s^q}, g_2^{\boldsymbol{d}_2^*}, g_2^{\boldsymbol{d}_2^* s}, \cdots g_2^{\boldsymbol{d}_2^* s^q}$ are publicly used for witness creation; and
- $g_2^{\boldsymbol{d}_1^*}$, $g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 s}, g_1^{\boldsymbol{d}_2 s^q}$ are publicly used for verification.

Thanks to that and the transformation from $\prod_{x \in \mathcal{X}} (x + s)$ to $\sum_{i=0}^{q} a_i s^i$, using the characteristic polynomial result given in Definition 2.1.2, the above public elements are easily computable from the knowledge of the successive powers of s in groups $\mathbb{G}_1$ or $\mathbb{G}_2$, as it is done in Nguyen. We obtain our scheme presented in Figure 5.8. Our scheme allows multiset setting and subset queries, thus we replace $\underline{x}$ by $\mathcal{I}$ in the construction. However, a non-membership witness for subset $\mathcal{I}$ is only computed if $\mathcal{X} \cap \mathcal{I} = \varnothing$. Our scheme is, as Nguyen's, bounded: let $q \in \mathbb{N}$ be its bound. Notice that we write in green elements needed when considering the scheme universal. Those elements can be removed when considering our accumulator scheme non-universal. Highlighting our scheme when non-universal will be helpful to build a data sharing scheme, that we present in Section 6.2.

Theorem 5.4.1 *Our accumulator is correct.*

Proof 5.4.1 *Correctness of membership: let $\mathcal{X}, \mathcal{I}$ be two sets such that $\mathcal{I} \subset \mathcal{X}$. Let $\{a_i, b_i, c_i\}_{i=0}^{q}$ be respectively the coefficients of polynomials $Ch_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (x + Z)$, $Ch_{\mathcal{X} \setminus \mathcal{I}}[Z] = \prod_{x \in \mathcal{X} \setminus \mathcal{I}} (x + Z)$ and $Ch_{\mathcal{I}}[Z] = \prod_{x \in \mathcal{I}} (x + Z)$. Let $acc_{\mathcal{X}} \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{acc}}, \mathcal{X})$ and $mwit_{\mathcal{I}} \leftarrow \mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathcal{X}, \mathcal{I}, 0)$. We have that*

$$e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, mwit_{\mathcal{I}}) = e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, g_2^{\boldsymbol{d}_2^* \sum_{i=0}^{q} b_i s^i}) = e(g_1, g_2)^{\psi \sum_{i=0}^{q} c_i s^i \cdot \sum_{i=0}^{q} b_i s^i}.$$

- Gen(λ, q): run a bilinear group generation algorithm to get $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$. Then choose a random $s \leftarrow \mathbb{Z}_p^*$, and run Dual$(\mathbb{Z}_p^2)$ to get $\mathbb{D} = (\boldsymbol{d}_1, \boldsymbol{d}_2), \mathbb{D}^* = (\boldsymbol{d}_1^*, \boldsymbol{d}_2^*)$. Let $\psi \in \mathbb{Z}_p^*$ be a random such that $\boldsymbol{d}_1 \cdot \boldsymbol{d}_1^* = \boldsymbol{d}_2 \cdot \boldsymbol{d}_2^* = \psi$. Set $\mathsf{sk}_{\mathsf{acc}} = (s, \mathbb{D}, \mathbb{D}^*)$,

$$\mathsf{pk}_{\mathsf{acc}} = \left(\begin{array}{c} \Gamma, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 s}, \cdots, g_1^{\boldsymbol{d}_2 s^q}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_1^* s}, \cdots, \\ g_2^{\boldsymbol{d}_1^* s^q}, g_2^{\boldsymbol{d}_2^*}, g_2^{\boldsymbol{d}_2^* s}, \cdots, g_2^{\boldsymbol{d}_2^* s^q}, e(g_1, g_2)^{\psi} \end{array} \right),$$

  and return $\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}$.
- Eval$(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$: compute the coefficients $\{a_i\}_{i=0,\cdots,q}$ of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (Z + x)$. Then compute $\mathsf{acc}_{\mathcal{X}} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i}$, and return $\mathsf{acc}_{\mathcal{X}}$.
- WitCreate$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathcal{X}, \mathcal{I}, \mathsf{Type})$: let $\{b_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{X}\setminus\mathcal{I}}[Z] = \prod_{x \in \mathcal{X}\setminus\mathcal{I}} (x + Z)$ and let $\{c_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{I}}[Z] = \prod_{x \in \mathcal{I}} (x + Z)$
  - If $\mathsf{Type} = 0$ (membership), compute $\mathsf{mwit}_{\mathcal{I}} = g_2^{\boldsymbol{d}_2^* \sum_{i=0}^{q} b_i s^i}$, and return $\mathsf{mwit}_{\mathcal{I}}$.
  - If $\mathsf{Type} = 1$ (non-membership), using Extended Euclidean algorithm, compute polynomials $q_1[Z], q_2[Z]$ such that $\mathsf{Ch}_{\mathcal{X}}[Z]q_1[Z] + \mathsf{Ch}_{\mathcal{I}}[Z]q_2[Z] = 1$. Then, set $W_1 = g_2^{\boldsymbol{d}_1^* q_1(s)}$ and $W_2 = g_2^{\boldsymbol{d}_2^* q_2(s)}$. Output $\mathsf{nmwit}_{\mathcal{I}} = (W_1, W_2)$.
- Verify$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{wit}_{\mathcal{I}}, \mathcal{I}, \mathsf{Type})$:
  - If $\mathsf{Type} = 0$, and $\mathsf{wit}_{\mathcal{I}}$ is not a membership witness, it outputs 0. Else, the algorithm returns 1 if $e(\mathsf{acc}_{\mathcal{X}}, g_2^{\boldsymbol{d}_1^*}) = e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, \mathsf{mwit}_{\mathcal{I}})$, 0 otherwise.
  - If $\mathsf{Type} = 1$, and $\mathsf{wit}_{\mathcal{I}}$ is not a non-membership witness, it outputs 0. Else, the algorithm returns 1 if $e(\mathsf{acc}_{\mathcal{X}}, W_1) \cdot e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, W_2) = e(g_1, g_2)^{\psi}$, 0 otherwise.

Figure 5.8: Our universal accumulator scheme, with private evaluation and public witness generation.

Here notice that Nguyen's scheme correctness comes into play, and as $\mathcal{I} \subset \mathcal{X}$ we have

$$e(g_1^{\boldsymbol{d}_2 \sum_{i=1}^{q} c_i s^i}, \mathit{mwit}_{\mathcal{I}}) = e(g_1, g_2)^{\psi \sum_{i=0}^{q} a_i s^i} = e(\mathit{acc}_{\mathcal{X}}, g_2^{\boldsymbol{d}_1^*}).$$

Correctness of non-membership: let $\mathcal{X}, \mathcal{I}$ be two sets such that $\mathcal{I} \not\subset \mathcal{X}$ and $\mathcal{I} \cap \mathcal{X} = \varnothing$. Let $\{a_i, c_i\}_{i=0}^{q}$ be respectively the coefficients of polynomials $\mathit{Ch}_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (x + Z)$, and $\mathit{Ch}_{\mathcal{I}}[Z] = \prod_{x \in \mathcal{X}} (x + Z)$. Let $\mathit{acc}_{\mathcal{X}} \leftarrow$ Eval$(\mathsf{sk}_{\mathsf{acc}}, \mathcal{X})$ and $\mathit{nmwit}_{\mathcal{I}} = (W_1, W_2) \leftarrow$ WitCreate

$(pk_{acc}, acc_{\mathcal{X}}, \mathcal{X}, \mathcal{I}, 1)$. *We have that*

$$\begin{aligned} e(acc_{\mathcal{X}}, W_1) \cdot e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, W_2) &= e(g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i}, g_2^{\boldsymbol{d}_1^* q_1(s)}) \cdot e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, g_2^{\boldsymbol{d}_2^* q_2(s)}) \\ &= e(g_1, g_2)^{\psi Ch_{\mathcal{X}}(s) q_1(s)} \cdot e(g_1, g_2)^{\psi q_2(s) Ch_{\mathcal{I}}(s)} \\ &= e(g_1, g_2)^{\psi (Ch_{\mathcal{X}}(s) q_1(s) + q_2(s) Ch_{\mathcal{I}}(s))}. \end{aligned}$$

Again, thanks to Nguyen's scheme correctness and as $\mathcal{I} \not\subset \mathcal{X}$*, we have*

$$e(acc_{\mathcal{X}}, W_1) \cdot e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, W_2) = e(g_1, g_2)^{\psi \cdot 1} = e(g_1, g_2)^{\psi}.$$

Regarding security our scheme is, as Nguyen's scheme, collision resistant under the q-SBDH problem (Definition 2.2.12).

Theorem 5.4.2 *Our accumulator satisfies collision resistance under the* q*-SBDH problem.*

Proof 5.4.2 *We prove the contrapositive. Let* $\mathcal{C}$ *be a* q*-SBDH challenger,* $\mathcal{B}$ *an adversary against collision resistance of the accumulator, that wins with non-negligible advantage. In Figure 5.9 we build* $\mathcal{A}$ *an adversary that breaks the* q*-SBDH assumption using* $\mathcal{B}$*.*

- On input $\lambda, q \in \mathbb{N}$, $\mathcal{C}$ runs bilinear group generation to get $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ and chooses $\alpha \leftarrow \mathbb{Z}_p^*$. It sends $\Gamma, g_1^{\alpha}, \cdots, g_1^{\alpha^q}, g_2, g_2^{\alpha}, \cdots, g_2^{\alpha^q}$ to $\mathcal{A}$.
- $\mathcal{A}$ runs Dual($\mathbb{Z}_p^2$) to get $\mathbb{D} = (\boldsymbol{d}_1, \boldsymbol{d}_2)$ and $\mathbb{D}^* = (\boldsymbol{d}_1^*, \boldsymbol{d}_2^*)$ such that $\boldsymbol{d}_1 \cdot \boldsymbol{d}_1^* = \boldsymbol{d}_2 \cdot \boldsymbol{d}_2^* = \psi$, where $\psi \in \mathbb{Z}_p^*$. Then it sets

$$\mathsf{pk}_{\mathsf{acc}} = \begin{pmatrix} \Gamma, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 \alpha}, \cdots, g_1^{\boldsymbol{d}_2 \alpha^q}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_1^* \alpha}, \cdots, \\ g_2^{\boldsymbol{d}_1^* \alpha^q}, g_2^{\boldsymbol{d}_2^*}, g_2^{\boldsymbol{d}_2^* \alpha}, \cdots, g_2^{\boldsymbol{d}_2^* \alpha^q}, e(g_1, g_2)^{\psi} \end{pmatrix}$$

  and sends it to $\mathcal{B}$.
- $\mathcal{B}$ makes an accumulator query: it chooses set $\mathcal{X}$ and sends it to $\mathcal{A}$. The latter uses its knowledge of $\boldsymbol{d}_1$ to return to $\mathcal{B}$ $\mathsf{acc}_{\mathcal{X}} = g_1^{\boldsymbol{d}_1 \mathsf{Ch}_{\mathcal{X}}(\alpha)}$. This step can be repeated an unbounded number of times.
- At some point, $\mathcal{B}$ answers either with $(\mathcal{X}^*, x^*, \mathsf{mwit}_{x^*})$ where $x^* \notin \mathcal{X}^*$ and mwit_{x^*} is a membership witness of x^* for set $\mathcal{X}^*$, or with $(\mathcal{X}, x^*, \mathsf{nmwit}_{x^*} = (W_1, W_2))$ where $x^* \in \mathcal{X}^*$ and nmwit_{x^*} is a non-membership witness of x^* for set $\mathcal{X}^*$.
- $\mathcal{A}$ returns to $\mathcal{C}$, either $(x^*, e(g_1, (\mathsf{mwit}_{x^*}^{\boldsymbol{d}_2^*})^{1/\psi r} \cdot (g_2^{-Q(\alpha)})^{1/r}))$ or $(x^*, e(g_1^{\boldsymbol{d}_1 Q(\alpha)}, W_1)^{\psi^{-1}} \cdot e(g_1^{\boldsymbol{d}_2}, W_2)^{\psi^{-1}})$ as its answer to break the assumption.

Figure 5.9: Construction of q-SBDH adversary from collision resistance adversary.

Let us see that the solution output by $\mathcal{A}$ is correct.

Suppose that $\mathcal{B}$ outputs $(\mathcal{X}^*, x^*, \mathsf{mwit}_{x^*})$. As $x^* \notin \mathcal{X}^*$, there exist $Q[Z], r$ such that $\mathsf{Ch}_{\mathcal{X}^*}[Z] = Q[Z] \cdot (x^* + Z) + r$. As mwit_{x^*} is a membership witness, we have that $e(g_1^{\boldsymbol{d}_2(x^*+\alpha)}, \mathsf{mwit}_{x^*}) = e(\mathsf{acc}_{\mathcal{X}^*}, g_2^{\boldsymbol{d}_1^*})$. Therefore, $e(g_1^{\boldsymbol{d}_2(x^*+\alpha)}, \mathsf{mwit}_{x^*}) = e(g_1, g_2)^{\psi(\alpha+x^*)Q(\alpha)+\psi r}$ and

$$(e(g_1, (\mathsf{mwit}_{x^*}^{\boldsymbol{d}_2^*})^{1/\psi r} \cdot (g_2^{-Q(\alpha)})^{1/r}))^{\alpha+x^*} = e(g_1, g_2)^{\frac{(\alpha+x^*)Q(\alpha)}{r+1}} \cdot (g_1, g_2)^{\frac{-(\alpha+x^*)Q(\alpha)}{r}} = e(g_1, g_2).$$

Notice that $\mathcal{A}$ knows $\boldsymbol{d}_2^*, \psi$ and r and can compute $g_2^{-Q(\alpha)}$ from the challenge tuple. Thus, $(x^*, e(g_1, (\mathsf{mwit}_{x^*}^{\boldsymbol{d}_2^*})^{1/\psi r} \cdot (g_2^{-Q(\alpha)})^{1/r}))$ is a solution to the q-SBDH problem.

Now, suppose that $\mathcal{B}$ outputs $(\mathcal{X}^*, x^*, \mathsf{nmwit}_{x^*} = (W_1, W_2))$. As $x^* \in \mathcal{X}^*$, there exists $Q[Z]$ such that $\mathsf{Ch}_{\mathcal{X}^*}[Z] = Q[Z](x^* + Z)$. As (W_1, W_2) is a non-membership witness, we have that $e(\mathsf{acc}_{\mathcal{X}^*}, W_1) \cdot e(g_1^{\boldsymbol{d}_2(x^*+\alpha)}, W_2) = e(g_1, g_2)^{\psi}$. Therefore,

$$\begin{aligned}
& e(\mathsf{acc}_{\mathcal{X}^*}, W_1) \cdot e(g_1^{\boldsymbol{d}_2(x^*+\alpha)}, W_2) = e(g_1, g_2)^{\psi} \\
\iff\ & e(g_1^{\boldsymbol{d}_1 Q(\alpha)(x^*+\alpha)}, W_1) \cdot e(g_1^{\boldsymbol{d}_2(x^*+\alpha)}, W_2) = e(g_1, g_2)^{\psi} \\
\iff\ & e(g_1^{\boldsymbol{d}_1 Q(\alpha)}, W_1)^{(x^*+\alpha)} \cdot e(g_1^{\boldsymbol{d}_2}, W_2)^{(x^*+\alpha)} = e(g_1, g_2)^{\psi} \\
\iff\ & (e(g_1^{\boldsymbol{d}_1 Q(\alpha)}, W_1) \cdot e(g_1^{\boldsymbol{d}_2}, W_2))^{(x^*+\alpha)} = e(g_1, g_2)^{\psi} \\
\iff\ & (e(g_1^{\boldsymbol{d}_1 Q(\alpha)}, W_1)^{\psi^{-1}} \cdot e(g_1^{\boldsymbol{d}_2}, W_2)^{\psi^{-1}})^{(x^*+\alpha)} = e(g_1, g_2)
\end{aligned}$$

Notice that $\mathcal{A}$ knows $\boldsymbol{d}_1, \boldsymbol{d}_2$ and ψ and can compute $g_1^{Q(\alpha)}$ from the challenge tuple. Thus $(x^*, e(g_1^{\boldsymbol{d}_1 Q(\alpha)}, W_1)^{\psi^{-1}} \cdot e(g_1^{\boldsymbol{d}_2}, W_2)^{\psi^{-1}})$ is a solution to the q-SBDH problem.

As $\mathcal{A}$ breaks the assumption when $\mathcal{B}$ breaks the collision resistance of the accumulator, we have that $\mathcal{A}$'s advantage is equal to $\mathcal{B}$'s advantage, meaning that $\mathcal{A}$ breaks the q-SBDH assumption with non-negligible advantage. □

Comparison with Nguyen's accumulator. As already stated, by construction our cryptographic accumulator is closed to Nguyen's accumulator [113]. We now compare in Table 5.4 our accumulator to [113]'s when considering the latter universal and in the asymmetric bilinear pairing setting. We highlight in red the differences between both schemes.

Table 5.4: Comparison between Nguyen's accumulator and ours.

Operation	Nguyen [113]	Ours
Evaluation	$\mathsf{acc}_{\mathcal{X}} = g_1^{\prod_{x\in\mathcal{X}}(x+s)}$	$\mathsf{acc}_{\mathcal{X}} = g_1^{\boldsymbol{d}_1 \prod_{x\in\mathcal{X}}(x+s)}$
Membership Witness	$\mathsf{mwit}_{\underline{x}} = g_2^{\prod_{x\in\mathcal{X}\setminus\{\underline{x}\}}(x+s)}$	$\mathsf{mwit}_{\underline{x}} = g_2^{\boldsymbol{d}_2^* \prod_{x\in\mathcal{X}\setminus\{\underline{x}\}}(x+s)}$.
Non-Membership Witness	$\mathsf{nmwit}_{\underline{x}} = (W_1, W_2) = (g_2^{q_1(s)}, g_2^{q_2(s)})$	$\mathsf{nmwit}_{\underline{x}} = (W_1, W_2) = (g_2^{\boldsymbol{d}_1^* q_1(s)}, g_2^{\boldsymbol{d}_2^* q_2(s)})$.
Membership Verification	$e(\mathsf{acc}_{\mathcal{X}}, g_2) \stackrel{?}{=} e(g_1^{\underline{x}} \cdot g_1^s, \mathsf{mwit}_{\underline{x}})$	$e(\mathsf{acc}_{\mathcal{X}}, g_2^{\boldsymbol{d}_1^*}) \stackrel{?}{=} e(g_1^{\boldsymbol{d}_2\underline{x}} \cdot g_1^{\boldsymbol{d}_2 s}, \mathsf{mwit}_{\underline{x}})$
Non-Membership Verification	$e(\mathsf{acc}_{\mathcal{X}}, W_1) \cdot e(g_1^{\underline{x}} \cdot g_1^s, W_2) \stackrel{?}{=} e(g_1, g_2)$	$e(\mathsf{acc}_{\mathcal{X}}, W_1) \cdot e(g_1^{\boldsymbol{d}_2\underline{x}} \cdot g_1^{\boldsymbol{d}_2 s}, W_2) \stackrel{?}{=} e(g_1, g_2)^{\psi}$

Unforgeability of private evaluation. We now prove that our accumulator (presented in Figure 5.8) satisfies our new security property, presented in Definition 5.3.1, *unforgeability of private evaluation*, if the fixed argument dual pairing vector spaces inversion problem holds (see Definition 2.3.10.)

Theorem 5.4.3 *If the fixed argument dual pairing vector spaces inversion problem holds, then our accumulator satisfies unforgeability of private evaluation.*

Proof 5.4.3 *We prove the contrapositive. Let $\mathcal{B}$ be an adversary that breaks UPE security with non negligible advantage. We build $\mathcal{A}$ an adversary that uses $\mathcal{B}$ to break FA-DPVS-I assumption.*
$\mathcal{A}$ is given $(\Gamma, g_1^{\boldsymbol{d}_2}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_2^*})$. She chooses $s \leftarrow \mathbb{Z}_p$, creates pk_{acc} and sends it to $\mathcal{B}$. $\mathcal{B}$ answer to $\mathcal{A}$ with a tuple of message-forged accumulator $(\mathcal{X}^*, \mathsf{acc}^*)$. $\mathcal{A}$ knows that for any $x \in \mathcal{X}^*$, $e(\mathsf{acc}^*, g_2^{\boldsymbol{d}_1^*}) = e(g_1^{\boldsymbol{d}_2(x+s)}, \mathsf{wit}_x)$ and that $e(g_1^{\boldsymbol{d}_2(x+s)}, \mathsf{wit}_x) = e(g_1, g_2)^{\psi \sum_{i=1}^q a_i s^i}$. Thus $e(\mathsf{acc}^*, g_2^{\boldsymbol{d}_1^*}) = e(g_1, g_2)^{\psi \sum_{i=1}^q a_i s^i}$. As $\mathcal{A}$ knows $\mathcal{X}^*$ she can recover $\{a_i\}_{i=0}^q$ and as she knows s, she can compute $(\sum_{i=0}^q a_i s^i)^{-1}$ and obtains that $e((\mathsf{acc}^*)^{(\sum_{i=0}^q a_i s^i)^{-1}}, g_2^{\boldsymbol{d}_1^*}) = e(g_1, g_2)^{\psi}$.
Thus $\mathcal{A}$ outputs $(\mathsf{acc}^*)^{(\sum_{i=0}^q a_i s^i)^{-1}}$ as her answer and wins the game with an advantage equal to $\mathcal{B}$'s advantage, therefore with non-negligible advantage. □

Note 5.4.3 *Notice that by construction in our scheme an accumulator is an element of $\mathbb{G}_1^2$.*

5.4.2 Our Dually Computable Accumulator

We now present our second cryptographic accumulator scheme, which is the first *dually computable* accumulator. We start with an overview of our construction and then give its presentation.

Overview of the construction. We started from our first accumulator scheme, presented in Figure 5.8. Our idea is to set $\mathsf{accp}_{\mathcal{X}} = g_2^{\boldsymbol{d}_1^* \prod_{x \in \mathcal{X}} (x+s)}$, which is publicly computable as $g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_1^* s}, \cdots, g_2^{\boldsymbol{d}_1^* s^q}$ are given in $\mathsf{pk}_{\mathsf{acc}}$ [1]. With the description of Eval as in the previous scheme, we directly obtain what we need. Moreover, the two accumulators are easily distinguishable as the secretly computed one is composed of two elements in $\mathbb{G}_1$ while the publicly generated one is composed of two elements in $\mathbb{G}_2$. Indeed, as stated in Definition 2.2.2, an asymmetric bilinear pairing group possesses efficient algorithms for deciding membership of the groups.
Regarding witnesses, membership witnesses can be computed as in our first accumulator: $\mathsf{mwit}_{\underline{x}} = g_2^{\boldsymbol{d}_2^* \prod_{x \in \mathcal{X} \setminus \{\underline{x}\}} (x+s)}$. For both privately and publicly computed accumulators, we are able to provide two very close verification equations. In fact, we remark that we obtain a sort of symmetry between the two accumulators, as $e(\mathsf{acc}_{\mathcal{X}}, g_2^{\boldsymbol{d}_1^*}) = e(g_1^{\boldsymbol{d}_1}, \mathsf{accp}_{\mathcal{X}})$, which two are equals to $e(g_1^{\boldsymbol{d}_2 \underline{x}} \cdot g_1^{\boldsymbol{d}_2 s}, \mathsf{mwit}_{\underline{x}})$, which is computable from the knowledge of the witness.
For non-membership witness, an issue raised. Indeed, if we define non-membership witnesses as in our first scheme, *i.e.* $\mathsf{nmwit}_{\underline{x}} = (W_1, W_2) = (g_2^{\boldsymbol{d}_1^* q_1(s)}, g_2^{\boldsymbol{d}_2^* q_2(s)})$, verification with publicly computed accumulator is not working. That is because in the non-membership verification, we have $e(\mathsf{acc}_{\mathcal{X}}, W_1)$. Therefore, as $\mathsf{acc}_{\mathcal{X}}$ is replaced by $\mathsf{accp}_{\mathcal{X}}$ which is composed of two elements of $\mathbb{G}_2$, the pairing with W_1 cannot work. The only solution to solve this problem is to have a different witness for accp and acc. As this does not correspond to our definition of dually computable accumulator (we recall that we require *correctness of duality*, meaning that a witness is working with both kind of accumulator in the verification algorithm), we decided to present a non-universal dually computable accumulator and leave as an open problem to build a *universal* dually computable accumulator.

Construction. In Figure 5.10, we present the full description of our first dually computable scheme, from the above intuition. We highlight in blue the algorithms and elements associated to public evaluation.

[1] We could have also chosen to define PublicEval such that it returns $g_2^{\boldsymbol{d}_2^* a_i s^i}$

- Gen(λ, q): run a bilinear group generation algorithm to get $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$. Then choose a random $s \leftarrow \mathbb{Z}_p^*$, and run Dual$(\mathbb{Z}_p^2)$ to get $\mathbb{D} = (\boldsymbol{d}_1, \boldsymbol{d}_2), \mathbb{D}^* = (\boldsymbol{d}_1^*, \boldsymbol{d}_2^*)$. Let $\psi \in \mathbb{Z}_p^*$ be the random such that $\boldsymbol{d}_1 \cdot \boldsymbol{d}_1^* = \boldsymbol{d}_2 \cdot \boldsymbol{d}_2^* = \psi$. Set $\mathsf{sk}_{\mathsf{acc}} = (s, \mathbb{D}, \mathbb{D}^*)$,

$$\mathsf{pk}_{\mathsf{acc}} = \left(\begin{array}{c} \Gamma, g_1^{\boldsymbol{d}_1}, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 s}, \cdots, g_1^{\boldsymbol{d}_2 s^q}, g_2^{\boldsymbol{d}_1^*}, \\ g_2^{\boldsymbol{d}_1^* s}, \cdots, g_2^{\boldsymbol{d}_1^* s^q}, g_2^{\boldsymbol{d}_2^*}, g_2^{\boldsymbol{d}_2^* s}, \cdots, g_2^{\boldsymbol{d}_2^* s^q} \end{array} \right)$$

  and return $\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}$.
- Eval$(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$: compute the coefficients $\{a_i\}_{i=0,\cdots,q}$ of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (Z + x)$. Then compute $\mathsf{acc}_{\mathcal{X}} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i}$, and return $\mathsf{acc}_{\mathcal{X}}$.
- PublicEval$(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$: compute the coefficients $\{a_i\}_{i=0,\cdots,q}$ of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (Z + x)$. Then compute $\mathsf{accp}_{\mathcal{X}} = g_2^{\boldsymbol{d}_1^* \sum_{i=0}^{q} a_i s^i}$, and return $\mathsf{accp}_{\mathcal{X}}$.
- WitCreate$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}} \backslash \mathsf{accp}_{\mathcal{X}}, \mathcal{X}, \mathcal{I})$: let $\{b_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{X}\backslash\mathcal{I}}[Z] = \prod_{x \in \mathcal{X}\backslash\mathcal{I}} (x + Z)$. Compute $\mathsf{wit}_{\mathcal{I}} = \mathsf{witp}_{\mathcal{I}} = g_2^{\boldsymbol{d}_2^* \sum_{i=0}^{q} b_i s^i}$, and return $\mathsf{wit}_{\mathcal{I}} = \mathsf{witp}_{\mathcal{I}}$.
- Verify$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{wit}_{\mathcal{I}}, \mathcal{I})$: let $\{c_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{I}}[Z] = \prod_{x \in \mathcal{I}} (x + Z)$ and return 1 if $e(\mathsf{acc}_{\mathcal{X}}, g_2^{\boldsymbol{d}_1^*}) = e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, \mathsf{wit}_{\mathcal{I}})$, 0 otherwise.
- PublicVerify$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}}, \mathsf{witp}_{\mathcal{I}}, \mathcal{I})$: let $\{c_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{I}}[Z] = \prod_{x \in \mathcal{I}} (x + Z)$ and return 1 if $e(g_1^{\boldsymbol{d}_1}, \mathsf{accp}_{\mathcal{X}}) = e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, \mathsf{witp}_{\mathcal{I}})$, 0 otherwise.

Figure 5.10: Our dually computable accumulator scheme.

Theorem 5.4.4 *Our scheme is correct, satisfies dual collision resistance under q-SBDH problem,* distinguishability *and correctness of duality.*

Lemma 5.4.1 *Our dually computable accumulator is correct.*

Proof 5.4.4 *Correctness for privately computed accumulator is done as for membership witnesses in Proof 5.4.1. Let us do the proof for publicly computed accumulators.*

Let $\mathcal{X}, \mathcal{I}$ be two sets such that $\mathcal{I} \subset \mathcal{X}$. Let $\{a_i, b_i, c_i\}_{i=0}^{q}$ be respectively the coefficients

of polynomials $Ch_{\mathcal{X}}[Z] = \prod_{x\in\mathcal{X}}(x+Z)$, $Ch_{\mathcal{X}\setminus\mathcal{I}}[Z] = \prod_{x\in\mathcal{X}\setminus\mathcal{I}}(x+Z)$ *and* $Ch_{\mathcal{I}}[Z] = \prod_{x\in\mathcal{I}}(x+Z)$. *Let* $accp_{\mathcal{X}} \leftarrow \mathsf{PublicEval}(\mathsf{sk}_{\mathsf{acc}}, \mathcal{X})$ *and* $witp_{\mathcal{I}} \leftarrow \mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}}, \mathcal{X}, \mathcal{I})$. *We have that*

$$e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, witp_{\mathcal{I}}) = e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, g_2^{\boldsymbol{d}_2^* \sum_{i=0}^{q} b_i s^i}) = e(g_1, g_2)^{\psi \sum_{i=0}^{q} c_i s^i \cdot \sum_{i=0}^{q} b_i s^i}.$$

Here notice that Nguyen's scheme correctness comes into play, and as $\mathcal{I} \subset \mathcal{X}$ *we have*

$$e(g_1^{\boldsymbol{d}_2 \sum_{i=1}^{q} c_i s^i}, witp_{\mathcal{I}}) = e(g_1, g_2)^{\psi \sum_{i=0}^{q} a_i s^i} = e(g_1^{\boldsymbol{d}_1}, accp_{\mathcal{X}}).$$

Lemma 5.4.2 *Our dually computable accumulator satisfies dual collision resistance under* q*-SBDH problem.*

Proof 5.4.5 *Dual collision resistance for privately computed accumulator is done as for membership witnesses in Proof 5.4.2. Let us do the proof for publicly computed accumulators.*

The proof is done by proving the contrapositive as in Figure 5.9. At the end of the game, $\mathcal{B}$ *outputs* $(\mathcal{X}^*, x^*, witp_{x^*})$ *where* $x^* \notin \mathcal{X}^*$ *and* $witp_{x^*}$ *is a membership witness for* x^*. *As* $x^* \notin \mathcal{X}^*$, *there exist* $Q[Z], r$ *such that* $Ch_{\mathcal{X}^*}[Z] = Q[Z]\cdot(x^*+Z)+r$. *As* $witp_{x^*}$ *is a membership witness, we have that* $e(g_1^{\boldsymbol{d}_2(x^*+s)}, witp_{x^*}) = e(g_1^{\boldsymbol{d}_1}, accp_{\mathcal{X}^*})$, *where* $accp_{\mathcal{X}^*} \leftarrow \mathsf{PublicEval}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X}^*)$. *Therefore,* $e(g_1^{\boldsymbol{d}_2(x^*+\alpha)}, witp_{x^*}) = e(g_1, g_2)^{\psi(\alpha+x^*)Q(\alpha)+\psi r}$ *and*

$$(e(g_1, (witp_{x^*}^{\boldsymbol{d}_2^*})^{1/\psi r}\cdot(g_2^{-Q(\alpha)})^{1/r}))^{\alpha+x^*} = e(g_1,g_2)^{\frac{(\alpha+x^*)Q(\alpha)}{r+1}}\cdot(g_1,g_2)^{\frac{-(\alpha+x^*)Q(\alpha)}{r}} = e(g_1,g_2).$$

Notice that $\mathcal{A}$ *knows* $\boldsymbol{d}_2^*, \psi$ *and* r *and can compute* $g_2^{-Q(\alpha)}$ *from the challenge tuple. Thus,* $(x^*, e(g_1, (witp_{x^*}^{\boldsymbol{d}_2^*})^{1/\psi r}\cdot(g_2^{-Q(\alpha)})^{1/r}))$ *is a solution to the* q*-SBDH problem.* □

Lemma 5.4.3 *Our dually computable accumulator satisfies distinguishability.*

Our accumulator satisfies *distinguishability* as a privately computed accumulator is composed of two elements in $\mathbb{G}_1$ while a publicly computed accumulator is composed of two elements in $\mathbb{G}_2$. In fact, in a bilinear environment, we know that there are an efficient algorithm for deciding membership of the groups (see e.g., Definition 2.2.2).

Lemma 5.4.4 *Our dually computable accumulator satisfies correctness of duality.*

Correctness of duality is satisfied as we have one unique witness (*i.e.* wit $=$ witp) and, as explained above, we have a symmetry between the two accumulators:

$$\underbrace{e(\mathsf{acc}_{\mathcal{X}}, g_2^{\boldsymbol{d}_1^*})}_{\text{from Eval}} = \underbrace{e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, \mathsf{wit}_{\mathcal{I}})}_{\text{from WitCreate}(\mathsf{pk}_{\mathsf{acc}},\mathsf{acc}_{\mathcal{X}},\mathcal{X},\mathcal{I})} = \underbrace{e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, \mathsf{witp}_{\mathcal{I}})}_{\text{from WitCreate}(\mathsf{pk}_{\mathsf{acc}},\mathsf{accp}_{\mathcal{X}},\mathcal{X},\mathcal{I})} = \underbrace{e(g_1^{\boldsymbol{d}_1}, \mathsf{accp}_{\mathcal{X}})}_{\text{from PublicEval}}.$$

Thus, the proof is exactly the same than in Proof 5.4.1.

5.5 Conclusion of This Chapter

This chapter introduced cryptographic accumulators schemes. Our contributions regarding that primitive is the definition of a new security property, call *unforgeability of private evaluation* along with the introduction of a new kind of accumulators: dually computable accumulators. In such scheme there are two evaluation algorithms Eval and PublicEval that take as input respectively only the secret key of the scheme ($\mathsf{sk}_{\mathsf{acc}}$) and the public key of the scheme ($\mathsf{pk}_{\mathsf{acc}}$).
We also proposes two instantiations of accumulators: one scheme that satisfies our new security property and one dually computable scheme.
Regarding cryptographic accumulators literature we provide a formal proof for the lower bound on symmetric accumulator size along with discussions on accumulators' applications and the delegatable property. As for the latter, we investigate a generic way to bring the property to any accumulator, and we leave as an open problem to know if (and how) all proof systems can be turned into homomorphic proofs. Doing the same thing for other properties, such as *asynchronous*, might be an interesting challenge and is therefore a lead for future works.
In Chapter 6.2 we will see how our dually computable accumulator scheme can be used to build a specific data sharing scheme, called *attribute-based encryption*.

6

Applications to Data Sharing

Contents

In this chapter we present two advanced cryptographic schemes used for data sharing: *broadcast encryption* and *attribute-based encryption*. For each of them, we give a formal definition along with associated properties, then we present our constructions from the primitives studied in the previous sections. The primitive of broadcast encryption is presented in Section 6.1 while in Section 6.2 we focus on the primitive of attribute-based encryption. We end this chapter in Section 6.3 with a use case for attribute-based encryption.

6.1 Broadcast Encryption

The first data sharing scheme we study in this thesis is *broadcast encryption* (BE), introduced in 1993 by Fiat and Naor [66]. In a broadcast encryption scheme the encryption algorithm takes as input the public key pk, a message m, a subset $S \subseteq [N]$ of users (N being the number of users in the system), and such that the output ciphertext can be decrypted by any user in the subset S. Regarding related work, Boneh *et al.* [38] were the first to achieve constant size ciphertext (i.e., independent of the number of users in the set), but the security was only selective and proven in the generic group model. Recently, Agrawal *et al.* ([9]) achieves constant size parameters with a security proven in the standard model. But it is only selective secure and their scheme combines both pairings and lattices. Lastly, Gay *et al.* ([71]) proposes a scheme based on pairings with constant size ciphertext. As far as we know, this is the only BE scheme with a constant-size ciphertext and providing adaptive security in the standard model.
In this manuscript, we also study a variant of broadcast encryption: *augmented broadcast encryption* (AugBE). An augmented broadcast encryption scheme is a broadcast encryption scheme in which the encryption algorithm takes as additional input an index ind $\in [N+1]$. As for any BE scheme, the output ciphertext can be decrypted by any user

in the subset S, but it is additionally required that the user's index is greater or equal to ind. In particular, if ind $= N + 1$, no one can decrypt. The first AugBE constructions [40, 70] give a ciphertext's size in $O(\sqrt{N})$. In [75], using both pairings and lattices, Goyal *et al.* propose a selectively secure construction with ciphertext size in $O(N^{\epsilon})$ $(0 < \epsilon \leq 1/2)$. Goyal *et al.* also propose in [77] a generic construction of an AugBE based on *positional witness encryption* (PWE). Their scheme is the first one providing constant parameters. However, currently only few instantiations of positional witness encryption exist and all rely on multilinear maps.
In the following, we first give broadcast encryption and augmented broadcast encryption definitions and properties, including our new definition of *anonymous* augmented broadcast encryption scheme. Then in Section 6.1.3 we present our contributions regarding BE and AugBe: new generic constructions from identity-based encryption with wildcards schemes. Finally we end this section with a presentation of our new broadcast encryption and augmented broadcast encryption schemes, and we compare them to the state of the art. We also present briefly Boneh *et al.* [40]'s generic construction of a third primitive, called *broadcast and trace*, from augmented broadcast encryption scheme. Using our new AugBE scheme and this construction, we obtain a new broadcast and trace scheme that we compare with existing ones.

6.1.1 Definitions and Properties

Definition 6.1.1 ***Broadcast encryption*** *[66, 71]. A* broadcast encryption *scheme consists of four algorithms.*

- Setup(λ, N)*: the setup algorithm takes as input a security parameter* $\lambda \in \mathbb{N}$ *and a number of users* $N \in \mathbb{N}$*. It outputs public parameters pk and a master secret key msk.*
- KeyGen(msk, i)*: the key generation algorithm takes as input a master secret key msk and an index* $i \in [N]$*. It outputs a secret key* sk_i *for user* i*.*
- Encrypt$(\mathsf{pk}, S, \mathsf{m})$*: the encryption algorithm takes as input a public key pk, a message m and a subset* $S \subseteq [N]$*. It outputs a ciphertext* ct_S*.*
- Decrypt$(\mathsf{pk}, S, i, \mathsf{sk}_i, \mathsf{ct}_S)$*: the decryption algorithm takes as input a public key pk, a subset* S*, an index* i*, a secret key* sk_i *and a ciphertext* ct_S*, and it returns a message* m'*.*

Definition 6.1.2 ***Correctness*** *[71]. A broadcast encryption scheme is said to be* correct *if for all security parameter* $\lambda \in \mathbb{N}$*, all number of users* $N \in \mathbb{N}$*, every honestly generated*

key pairs $(msk, pk) \leftarrow \mathsf{Setup}(\lambda, N)$*, every messages* m*, every set* $S \subseteq [N]$ *and every* $i \in [N]$ *such that* $i \in S$*:*

$$\Pr\left[\mathsf{Decrypt}(\mathsf{pk}, S, i, \mathsf{KeyGen}(\mathsf{msk}, i), \mathsf{Encrypt}(\mathsf{pk}, S, \mathsf{m})) = \mathsf{m}\right] = 1.$$

Definition 6.1.3 ***Adaptive indistinguishability*** *(IND-BE) [71]. A broadcast encryption scheme is said to satisfy* adaptive indistinguishability security *if all PPT adversaries* $\mathcal{A}$ *have at most negligible advantage in winning the game presented in Figure 6.1, where* $\mathcal{A}$*'s advantage is defined as* $\mathsf{Adv}_{\mathcal{A}}^{\text{IND-BE}}(\lambda) := \left|\Pr\left[b' = b\right] - 1/2\right|$*. Let* $\mathcal{C}$ *be a challenger.*

`SETUP`: $\mathcal{C}$ on inputs (λ, N) runs $\mathsf{Setup}(\lambda, N)$ to generate pk and msk, and gives pk to $\mathcal{A}$.
`KEY QUERY`: $\mathcal{A}$ issues queries to $\mathcal{C}$ for index $i \in [N]$. $\mathcal{C}$ returns $\mathsf{sk}_i \leftarrow \mathsf{KeyGen}(\mathsf{msk}, i)$.
`CHALLENGE`: $\mathcal{A}$ selects two challenge messages $\mathsf{m}_0, \mathsf{m}_1$ and a challenge set $S^* \subseteq [N]$ of users. $\mathcal{A}$ sends $\mathsf{m}_0, \mathsf{m}_1$ and S^* to $\mathcal{C}$. The latter picks $b \in \{0, 1\}$ randomly and computes $\mathsf{ct}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}, S^*, \mathsf{m}_b)$ which is returned to $\mathcal{A}$.
`KEY QUERY`: $\mathcal{A}$ makes queries for index $i \in [N]$. $\mathcal{C}$ returns $\mathsf{sk}_i \leftarrow \mathsf{KeyGen}(\mathsf{msk}, i)$ to $\mathcal{A}$.
`GUESS`: $\mathcal{A}$ outputs her guess $b' \in \{0, 1\}$ for b, and wins the game if $b' = b$ and if, for all index $i \in [N]$ for which a key was queried, the condition $i \notin S^*$ holds.

Figure 6.1: Adaptive indistinguishability security game for broadcast encryption schemes.

Definition 6.1.4 ***Anonymous broadcast encryption scheme*** *(*ANO-BE*) [19, 100]. A broadcast encryption scheme is said to be adaptively* anonymous *if all PPT adversaries* $\mathcal{A}$ *have at most negligible advantage in the game presented in Figure 6.2, where* $\mathcal{C}$ *is a challenger and* $\mathcal{A}$*'s advantage is defined as* $\mathsf{Adv}_{\mathcal{A}}^{\text{ANO-BE}}(\lambda) = \left|\Pr\left[b' = b\right] - 1/2\right|$*.*

Note 6.1.1 *Many broadcast encryption schemes require the encryption set* S *to be publicly given as an input of decryption algorithm. Otherwise even authorized users will not be able to decrypt. However, anonymous schemes do not need the encryption set description as an input for the decryption algorithm.*

Definition 6.1.5 ***Augmented broadcast encryption scheme*** *(AugBE) [40, 77]. An augmented broadcast encryption scheme consists of three algorithms.*

- $\mathsf{Setup}(\lambda, N)$*: the setup algorithm takes as input a security parameter* $\lambda \in \mathbb{N}$ *and a number of users* $N \in \mathbb{N}$*. It outputs a master secret key msk, a public key pk and secret keys* $\{sk_1, \cdots, sk_N\}$*, where* sk_i *is the secret key for user* i*.*

SETUP: $\mathcal{C}$ on inputs λ, N runs Setup(λ, N) to generate pk and msk, and gives pk to $\mathcal{A}$.
KEY QUERY: $\mathcal{A}$ issues queries to $\mathcal{C}$ for index $i \in [N]$. $\mathcal{C}$ returns $\mathsf{sk}_i \leftarrow$ KeyGen(msk, i).
CHALLENGE: $\mathcal{A}$ selects two challenge messages $\mathsf{m}_0, \mathsf{m}_1$ and two distinct challenge sets $S^0, S^1 \subseteq [N]$ of users. $\mathcal{A}$ passes $\mathsf{m}_0, \mathsf{m}_1$ and S^0, S^1 to $\mathcal{C}$. The latter picks $b \in \{0, 1\}$ randomly and computes $\mathsf{ct}^* \leftarrow$ Encrypt$(\mathsf{pk}, S^b, \mathsf{m}_b)$ which is returned to $\mathcal{A}$.
KEY QUERY: $\mathcal{A}$ continues to make queries for index $i \in [N]$ $\mathcal{C}$ answers $\mathsf{sk}_i \leftarrow$ KeyGen(msk, i) to $\mathcal{A}$.
GUESS: $\mathcal{A}$ outputs her guess $b' \in \{0, 1\}$ for b, and wins the game if $b' = b$ and if, for all index $i \in [N]$ for which a key was queried, the condition $i \notin S^0 \wedge i \notin S^1$ or $i \in S^0 \wedge i \in S^1$ and if $\mathsf{m}_0 \neq \mathsf{m}_1$ then $i \notin S^0 \cap S^1$ holds.

Figure 6.2: Adaptive anonymous security game for broadcast encryption schemes.

- Encrypt$(\mathsf{pk}, S, \mathsf{m}, \mathsf{ind})$*: the encryption algorithm takes as input a public key pk, a set of users* $S \subseteq [N]$*, a message m and an index* ind $\in [N + 1]$*. It outputs a ciphertext ct.*
- Decrypt$(\mathsf{pk}, \mathsf{sk}_i, S, \mathsf{ct})$*: the decryption algorithm takes as input a public key pk, a secret key* sk_i*, a set of users* $S \subseteq [N]$*, and a ciphertext ct, and it returns a message* m'*.*

Note 6.1.2 *More recently, Goyal* et al. *[75] gave another definition of augmented broadcast encryption in which there are two encryption algorithms called* Encrypt *and* Index*-Encrypt. The former takes as input the scheme public key, a set of users and a message to encrypt while the latter as an additional input which is an index* ind*. In the following, we will consider only the original AugBE definition, i.e., the one with one encryption algorithm only.*

Definition 6.1.6 ***Correctness*** *[77]. An augmented broadcast encryption scheme is said to be* correct *if for every security parameter* $\lambda \in \mathbb{N}$*, all number of users* $N \in \mathbb{N}$*, every message m, every set* S *such that* $S \subseteq [N]$*, every index* ind $\in [N]$*, all* $i \in [N]$ *such that* $i \in S \cap \{ind, \cdots, N\}$*, and every honestly generated keys* $(\mathsf{msk}, \mathsf{pk}, \{\mathsf{sk}_1, \cdots, \mathsf{sk}_N\}) \leftarrow$ Setup(λ, N)*:*

$$\Pr\left[\mathsf{Decrypt}(\mathsf{pk}, \mathsf{sk}_i, S, \mathsf{Encrypt}(\mathsf{pk}, S, \mathsf{m}, \mathsf{ind})) = \mathsf{m}\right] = 1.$$

Definition 6.1.7 ***Message-hiding security*** *(MH-AugBE) [77]. An augmented broadcast encryption scheme is said to satisfy* adaptive message-hiding security *if all*

PPT adversaries $\mathcal{A}$ have at most negligible advantage in the game presented in Figure 6.3, where $\mathcal{C}$ is a challenger and $\mathcal{A}$'s advantage is defined as $\mathsf{Adv}_{\mathcal{A}}^{\text{MH-AugBE}}(\lambda) := \left|\Pr\left[b' = b\right] - 1/2\right|$.

SETUP: $\mathcal{C}$ on inputs λ, N runs Setup(λ, N) to obtain msk, pk, $\{\mathsf{sk}_i\}_{i\in[N]}$ and gives pk to $\mathcal{A}$.
KEY QUERY: $\mathcal{A}$ chooses an index $i \in [N]$ and sends it to $\mathcal{C}$, who responds with sk_i.
CHALLENGE: $\mathcal{A}$ chooses two challenge messages $\mathsf{m}_0, \mathsf{m}_1$ and a challenge set S^* and sends it to $\mathcal{C}$. $\mathcal{C}$ chooses $b \in \{0,1\}$ randomly, runs $\mathsf{ct}^* \leftarrow$ Encrypt$(\mathsf{pk}, S^*, \mathsf{m}_b, N+1)$ and gives ct^* to $\mathcal{A}$.
KEY QUERY: $\mathcal{A}$ chooses an index $i \in [N]$ and sends it to $\mathcal{C}$, who responds with sk_i.
GUESS: $\mathcal{A}$ outputs her guess $b' \in \{0,1\}$ for b, and wins the game if $b' = b$.

Figure 6.3: Adaptive message-hiding security game for augmented broadcast encryption schemes.

Definition 6.1.8 ***Index-hiding security*** *(IH-AugBE) [77]. An augmented broadcast encryption scheme is said to satisfy* adaptive index-hiding security *if all PPT adversaries $\mathcal{A}$ have at most negligible advantage in the game presented in Figure 6.4, where $\mathcal{C}$ is a challenger and $\mathcal{A}$'s advantage is defined as* $\mathsf{Adv}_{\mathcal{A}}^{\text{IH-AugBE}}(\lambda) := \left|\Pr\left[b' = b\right] - 1/2\right|$.

SETUP: $\mathcal{C}$ on inputs λ, N runs Setup(λ, N) to obtain pk, msk, $\{\mathsf{sk}_i\}_{i\in[N]}$ and gives pk to $\mathcal{A}$.
KEY QUERY: at each query, $\mathcal{A}$ chooses an index $i \in [N]$ and sends it to $\mathcal{C}$. $\mathcal{C}$ responds with sk_i. Let S be the set of indices for which a key is queried by $\mathcal{A}$. $\mathcal{C}$ adds i to S.
CHALLENGE: $\mathcal{A}$ chooses a challenge message m, a challenge set S^* and a challenge index ind $\in [N]$ and sends them to $\mathcal{C}$. $\mathcal{C}$ chooses $b \in \{0,1\}$ randomly, runs $\mathsf{ct}^* \leftarrow$ Encrypt$(\mathsf{pk}, S^*, \mathsf{m}, \mathsf{ind} + b)$ and gives ct^* to $\mathcal{A}$.
KEY QUERY: at each query, $\mathcal{A}$ chooses an index $i \in [N]$ and sends it to $\mathcal{C}$ who adds i to S. $\mathcal{C}$ responds with sk_i.
GUESS: $\mathcal{A}$ outputs her guess $b' \in \{0,1\}$ for b, and wins the game if $b' = b$ and if, for index $i \in [N]$ for which a key was queried, the condition ind $\notin S \cap S^*$ holds.

Figure 6.4: Adaptive index-hiding security game for augmented broadcast encryption schemes.

6.1.2 Our Contribution: Anonymous Augmented Broadcast Encryption

In this thesis, we introduce a new security property for augmented broadcast encryption scheme: *anonymity*. The below definition, close to the one for broadcast schemes ([100]), provides the adaptive version.

Definition 6.1.9 ***Anonymous augmented broadcast encryption*** *(ANO-AugBE). An augmented broadcast encryption scheme is said to be adaptively* anonymous *if all PPT adversaries $\mathcal{A}$ have at most negligible advantage in the game presented in Figure 6.5, where $\mathcal{C}$ is a challenger and $\mathcal{A}$'s advantage is defined as* $\mathsf{Adv}_{\mathcal{A}}^{\text{ano-augbe}}(\lambda) := \left|\Pr\left[b' = b\right] - 1/2\right|$.

SETUP: $\mathcal{C}$ on inputs λ, N runs Setup(λ, N) to obtain pk, msk, $\{\mathsf{sk}_i\}_{i\in[N]}$, and gives pk to $\mathcal{A}$.
KEY QUERY: $\mathcal{A}$ can issue queries to the challenger for index $i \in [N]$. $\mathcal{C}$ responds with sk_i.
CHALLENGE: $\mathcal{A}$ selects a challenge message m, two distinct challenge sets $S^0, S^1 \subseteq [N]$ of users and a challenge index ind $\in [N+1]$. $\mathcal{A}$ passes $\mathsf{m}, S^0, S^1, \mathsf{ind}$ to $\mathcal{C}$. The latter picks $b \in \{0,1\}$ randomly and computes $\mathsf{ct}^* \leftarrow$ Encrypt$(\mathsf{pk}, S^b, \mathsf{m}, \mathsf{ind})$ which is returned to $\mathcal{A}$.
KEY QUERY: $\mathcal{A}$ makes queries for index $i \in [N]$ $\mathcal{C}$ responds with sk_i.
GUESS: $\mathcal{A}$ outputs her guess $b' \in \{0,1\}$ for b, and wins the game if $b' = b$ and if, for all index $i \in [N]$ for which a key was queried, if $i \geq$ ind then the condition $i \in S^0 \cap S^1$ holds.

Figure 6.5: Adaptive anonymous security game for augmented broadcast encryption schemes.

In the following theorem we prove that this new anonymity property implies index-hiding security.

Theorem 6.1.1 *If an augmented broadcast encryption scheme is anonymous, then it is also index-hiding.*

Proof 6.1.1 *Let $\mathcal{C}$ be a challenger and $\mathcal{B}$ be an adversary that wins the index-hiding security game with non negligible advantage.*
Informally, index-hiding means that

- *without the key $\mathsf{sk}_{\mathsf{ind}}$, an adversary cannot distinguish between an encryption for index* ind *and one for index* ind $+ 1$*;*

- *and if* ind *is not in the target set* S^**, then no adversary can distinguish an encryption for index* ind *and one for index* ind $+ 1$ *([40]).*

Thus $\mathcal{B}$ *can either distinguish which index was used in encryption when* ind $\in S^*$ *and without knowing* sk_{ind}*, or she can distinguish the encryption index when* ind $\notin S^*$*, knowing* sk_{ind}*. Therefore she either chooses* ind $\in S^*$ *or* ind $\notin S^*$ *but in this case she asks* sk_{ind} *otherwise she would have advantage equal to* $1/2$*.*

We construct, in Figure 6.6, an adversary $\mathcal{A}$ *that wins the anonymous security game with non negligible advantage.*

`SETUP`: $\mathcal{C}$ on inputs λ, N runs Setup$(\lambda, N) \rightarrow (\text{msk}, \text{pk}, \text{sk}_1, \cdots, \text{sk}_N)$, and sends pk to $\mathcal{A}$, and $\mathcal{B}$.
`KEY QUERY`: $\mathcal{B}$ chooses $i \in [N]$, sends it to $\mathcal{A}$ who sends it to $\mathcal{C}$. The later sends sk_i to $\mathcal{A}$ who sends it to $\mathcal{B}$.
`CHALLENGE`: $\mathcal{B}$ chooses a challenge message m, a challenge set S^* of users and a challenge index ind $\in [N]$ and sends m, S^*, ind to $\mathcal{A}$. The latter creates the sets $S^0 = S^* \cap \{\text{ind}, \cdots, N\}$ and $S^1 = S^* \cap \{\text{ind} + 1, \cdots, N\}$. $\mathcal{A}$ sends m, S^0, S^1 to $\mathcal{C}$. $\mathcal{C}$ chooses $b \leftarrow \{0, 1\}$ randomly and sets $\text{ct}^* \leftarrow$ Encrypt$(\text{pk}, S^b, \text{m}, 1)$. It sends ct^* to $\mathcal{A}$ who sends it to $\mathcal{B}$.
`KEY QUERY`: $\mathcal{A}$ and $\mathcal{B}$ act like in the previous `KEY QUERY` step. $\mathcal{C}$ sends sk_i to $\mathcal{A}$ who sends it to $\mathcal{B}$.
`GUESS`: $\mathcal{B}$ outputs her guess b' to $\mathcal{A}$, who outputs it as her own guess.

Figure 6.6: Construction of ANO-AugBE adversary from IH-AugBE adversary.

We have that if all $\mathcal{B}$*'s queries satisfy the game constraints, then all* $\mathcal{A}$*'s queries have the same property. Thus* $\mathcal{A}$*'s simulation is perfect and the advantage of* $\mathcal{A}$ *is the same as* $\mathcal{B}$*'s. This concludes the proof.* □

Note 6.1.3 *If* ind $\in S^*$*, then* ind $\in S^0 \wedge$ ind $\notin S^1$ *thus adversary cannot query* sk_{ind}*. If* ind $\notin S^*$*, then* ind $\notin S^0 \wedge$ ind $\notin S^1$ *thus adversary can query* sk_{ind}*.*

Note 6.1.4 *Index-hiding does not imply anonymous. Indeed, in the index-hiding security game, in the case where* ind *is not in the challenge, knowing the challenge set does not help determining if* ind *or* ind $+ 1$ *was used for encryption.*

6.1.3 From WIBE to (Aug)BE: Our Generic Constructions

In this section we present two generic broadcast encryption constructions from identity-based encryption with wildcards: one for a basic BE scheme and the other for an AugBE scheme. We also formalize which properties of WIBE are needed in order to obtain a secure BE (resp. AugBE). As in a broadcast encryption scheme there is no key delegation from user to another, we do not consider this feature of the underlying identity-based encryption with wildcards scheme, and thus replace KeyDer by KeyGen in the WIBE definition. For sake of simplicity, we admit in proofs that the number of keys queried is always lower or equal to the maximal number Q of keys that an adversary is allowed to query. All proofs are done for adaptive security definitions and can be adapted to the selective case. The length of patterns is $L \in \mathbb{N}$.

First notice that any subset $S^* \subseteq [N]$ can be represented as a pattern $\boldsymbol{P} \in \{0, \star\}^N$, where for $j \in [N]$, $P_j = \star$ if $j \in S^*$ and $P_j = 0$ otherwise. This fact can then be used to associate such pattern to the BE encryption set S. Additionally, any user identity $i \in [N]$ can be represented as a pattern $\boldsymbol{P}^i \in \{0,1\}^N$ such that for $j \in [N]$, $P^i_j = 1$ if $i = j$ and $P^i_j = 0$ otherwise. This finally gives us that $i \in S$ iff $\boldsymbol{P}^i$ belongs to $\boldsymbol{P}$. Regarding AugBE, we have noticed that the decrypting condition $i \geq \mathsf{ind}$ for any $i \in [N]$, $\mathsf{ind} \in [N+1]$ can be rewritten as $i \in \{\mathsf{ind}, \mathsf{ind}+1, \cdots, N+1\}$. It follows that the AugBE decrypting condition becomes $i \in S \cap \{\mathsf{ind}, \mathsf{ind}+1, \cdots, N+1\}$.
In the sequel, we set L the length pattern of the WIBE schemes to be equal to N the number of users in the BE and AugBE schemes.

Broadcast Encryption from WIBE. Let $WIBE = (Setup, KeyGen, Encrypt, Decrypt)$ be an identity-based encryption with wildcards scheme for key pattern space $\{0,1\}^N \backslash \{\mathbf{0}^N\}$ and ciphertext pattern space $\{0, \star\}^N \backslash \{\mathbf{0}^N\}$. We construct a broadcast encryption scheme $\mathsf{BE} = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt})$ in Figure 6.7.

Setup(λ, N): run $Setup(\lambda, N)$ to get pk, msk and set $\mathsf{pk} = pk$ and $\mathsf{msk} = msk$.
KeyGen$(\mathsf{msk}, i \in [N])$: define $\boldsymbol{P}' \in \{0,1\}^N$ such that for $j \in [N]$, $P'_j = 0$ if $j \neq i$ and $P'_j = 1$ if $i = j$. Then run $KeyGen(msk, \boldsymbol{P}')$ to get sk_i and set $\mathsf{sk}_i = sk_i$. It outputs sk_i.
Encrypt$(\mathsf{pk}, S, \mathsf{m})$: first, associate S with a pattern $\boldsymbol{P}$ in $\{0, \star\}^N$ such that for $j \in [N]$, $P_j = \star$ if $j \in S$ and $P_j = 0$ otherwise. Then compute $ct = Encrypt(pk, \boldsymbol{P}, \mathsf{m})$ and outputs $\mathsf{ct} = ct$.
Decrypt$(\mathsf{pk}, \mathsf{sk}_i, \mathsf{ct}, S)$: gets $\mathsf{m}' \leftarrow Decrypt(sk_i, \boldsymbol{P}, ct)$.

Figure 6.7: Generic construction of broadcast encryption scheme from identity-based encryption with wildcards scheme.

Note 6.1.5 *Encryption for pattern $\mathbf{0}^N$ is not relevant here as it means that no one can decrypt, that is why we excluded this pattern of encryption pattern space. Secret key for pattern $\mathbf{0}^N$ is not relevant either as it does not match any of the users.*

Theorem 6.1.2 *The broadcast encryption scheme obtained is correct if the underlying identity-based encryption with wildcards is correct.*

Proof 6.1.2 *$\boldsymbol{P}^i \in_\star \boldsymbol{P}$ implies that $P^i_i = P_i$ or $P^i = \star$. As $P^i_i = 1$, we have that $P^i = \star$ and thus $i \notin \bar{W}(\boldsymbol{P})$, i.e. $i \in S$. Suppose that $i \in S$. By construction for all $j \in [N], j \neq i$, $P^i_j = 0$ and either $P^i_j = 0 = P_j$ or $P_j = \star$, and $P_i = \star$, i.e. $\boldsymbol{P}^i \in_\star \boldsymbol{P}$. Then correctness follows from WIBE's correctness.* □

Theorem 6.1.3 *If $WIBE$ satisfies* IND-WIBE *security, then the obtained BE scheme satisfies* IND-BE *security.*

Proof 6.1.3 *Let $\mathcal{B}$ be an adversary against* IND-BE *security, that wins with non negligible advantage. In Figure 6.8 we construct $\mathcal{A}$ an adversary against* IND-WIBE *that uses $\mathcal{B}$ and wins with non negligible advantage. Let $\mathcal{C}$ be a challenger.*

`SETUP`: $\mathcal{C}$ on inputs (λ, N) runs Setup$(\lambda, N) \rightarrow (\mathsf{msk}, \mathsf{pk})$ and gives pk to $\mathcal{A}$, who gives it to $\mathcal{B}$.
`KEY QUERY`: $\mathcal{B}$ chooses an index $i \in [N]$ and sends it to $\mathcal{A}$, who creates $\boldsymbol{P}^i$, for $j \in [N]$, such that $P^i_j = 1$ if $i = j$ and $P^i_j = 0$ otherwise. $\mathcal{A}$ sends $\boldsymbol{P}^i$ to $\mathcal{C}$. The latter runs KeyDer$(\mathsf{msk}, \boldsymbol{P}^i) \rightarrow \mathsf{sk}_{\boldsymbol{P}^i}$ and sends $sk_{\boldsymbol{P}^i}$ to $\mathcal{A}$, who sends it as sk_i to $\mathcal{B}$.
`CHALLENGE`: $\mathcal{B}$ chooses two challenge messages $\mathsf{m}_0, \mathsf{m}_1$ and a challenge set S^*; it sends it to $\mathcal{A}$ who creates the pattern $\boldsymbol{P}^*$, for $j \in [N]$ s.t. $\boldsymbol{P}^*_j = 0$ if $j \notin S^*$, $P^*_j = \star$ otherwise, and sends $\boldsymbol{P}^*, \mathsf{m}_0, \mathsf{m}_1$ to $\mathcal{C}$. $\mathcal{C}$ chooses $b \in \{0, 1\}$ randomly and runs $\mathsf{ct}^* \leftarrow$ Encrypt$(\mathsf{pk}, \boldsymbol{P}^*, \mathsf{m}_b)$. It sends ct^* to $\mathcal{A}$ who sends it to $\mathcal{B}$.
`KEY QUERY`: $\mathcal{B}$ chooses index $i \in [N]$, sends it to $\mathcal{A}$, who creates $\boldsymbol{P}^i$, for $j \in [N]$, s.t. $P^i_j = 1$ if $i = j$ and $P^i_j = 0$ otherwise. $\mathcal{A}$ sends $\boldsymbol{P}^i$ to $\mathcal{C}$. $\mathcal{C}$ runs KeyDer$(\mathsf{msk}, \boldsymbol{P}^i) \rightarrow \mathsf{sk}_{\boldsymbol{P}^i}$ and sends $sk_{\boldsymbol{P}^i}$ to $\mathcal{A}$, who sends it as sk_i to $\mathcal{B}$.
`GUESS`: $\mathcal{B}$ outputs a bit b' to $\mathcal{A}$ who outputs it as its guess.

Figure 6.8: Construction of IND-WIBE adversary from IND-BE adversary.

If all $\mathcal{B}$'s queries satisfy the game constraints, then all $\mathcal{A}$'s queries have the same property. Thus, $\mathcal{A}$'s simulation is perfect and the advantage of $\mathcal{A}$ is the same as $\mathcal{B}$'s. □

Theorem 6.1.4 *If the underlying identity-based encryption with wildcards satisfies pattern-hiding security, then the obtained broadcast encryption scheme is anonymous.*

Proof 6.1.4 *Let $\mathcal{B}$ be an adversary against anonymous security, that wins with non negligible advantage. In Figure 6.9 we construct $\mathcal{A}$, an adversary against pattern-hiding security that uses $\mathcal{B}$ and wins with non negligible advantage. Let $\mathcal{C}$ be a challenger.*

SETUP: $\mathcal{C}$ on inputs λ, N runs Setup$(\lambda, N) \rightarrow (\mathsf{msk}, \mathsf{pk})$ and sends pk to $\mathcal{A}$, who sends it to $\mathcal{B}$.
KEY QUERY: $\mathcal{B}$ chooses $i \in [N]$, sends it to $\mathcal{A}$ who creates the pattern $\boldsymbol{P}^i$ such that for $j \in [N]$, $P^i_j = 1$ if $i = j$ and $P^i_j = 0$ otherwise. $\mathcal{A}$ sends $\boldsymbol{P}^i$ to $\mathcal{C}$ who runs KeyDer$(\mathsf{msk}, \boldsymbol{P}^i) \rightarrow \mathsf{sk}_{\boldsymbol{P}^i}$. $\mathcal{A}$ receives $\mathsf{sk}_{\boldsymbol{P}^i}$ and sends it to $\mathcal{B}$ as sk_i.
CHALLENGE: $\mathcal{B}$ chooses challenge message m, two challenge sets S_0, S_1 and sends them to $\mathcal{A}$ who creates patterns $\boldsymbol{P}^0, \boldsymbol{P}^1$ s.t. for $j \in [N]$, $P^0_j = \star$ if $j \in S_0$, $P^0_j = 0$ otherwise, and $P^1_j = \star$ if $j \in S_1$, $P^1_j = 0$ otherwise. m, $\boldsymbol{P}^0, \boldsymbol{P}^1$ are sent to $\mathcal{C}$. $\mathcal{C}$ chooses $b \leftarrow \{0,1\}$ randomly and runs $\mathsf{ct}^* \leftarrow$ Encrypt$(\mathsf{pk}, \boldsymbol{P}^b, \mathsf{m})$. $\mathcal{C}$ sends ct^* to $\mathcal{A}$, who sends it to $\mathcal{B}$.
KEY QUERY: $\mathcal{B}$ and $\mathcal{A}$ proceeds as in the first KEY QUERY step. $\mathcal{C}$ runs KeyDer$(\mathsf{msk}, \boldsymbol{P}^i) \rightarrow \mathsf{sk}_{\boldsymbol{P}^i}$. $\mathsf{sk}_{\boldsymbol{P}^i}$ is sent to $\mathcal{A}$, who sends it to $\mathcal{B}$ as sk_i.
GUESS: $\mathcal{B}$ outputs its guess b' to $\mathcal{A}$, who outputs it as its guess.

Figure 6.9: Construction of PH-WIBE-CPA adversary from ANO-BE adversary.

If all $\mathcal{B}$'s queries satisfy the game constraints, then all $\mathcal{A}$'s queries have the same property. Thus $\mathcal{A}$'s simulation is perfect, and the advantage of $\mathcal{A}$ is the same as $\mathcal{B}$'s. This concludes the proof. □

Augmented Broadcast Encryption from WIBE. Let $WIBE = (Setup, KeyGen, Encrypt, Decrypt)$ be an identity-based encryption with wildcards scheme for key pattern space $\{0,1\}^N \backslash \{\mathbf{0}^N\}$ and ciphertext pattern space $\{0, \star\}^N$. We now construct an augmented broadcast encryption scheme AugBE $=$ (Setup, Encrypt, Decrypt) in Figure 6.10.

Note 6.1.6 *Here encryption for pattern $\mathbf{0}^N$ corresponds to encryption for index $N + 1$.*

Note 6.1.7 *As the underlying WIBE is pattern-hiding, the AugBE decryption algorithm does not take as input the set for which the message was encrypted.*

Theorem 6.1.5 *The augmented broadcast encryption scheme obtained is correct if the underlying identity-based encryption with wildcards scheme is correct.*

Setup(λ, N): and run $Setup(\lambda, N)$ to obtain pk, msk. Then for each $i \in [N]$, define $\boldsymbol{P}^{'} \in \{0,1\}^N$ such that for $j \in [N]$, $P^{'}_j = 0$ if $j \neq i$ and $P^{'}_j = 1$ if $i = j$. Then set $\mathsf{sk}_i = sk_i = KeyGen(msk, \boldsymbol{P}^{'})$, $(\mathsf{pk}, \mathsf{msk}) = (pk, msk)$. It outputs msk, pk and $\{\mathsf{sk}_i\}_{i\in[N]}$.

Encrypt(pk, S, ind, m): here ind $\in [N+1]$. Associate S with a pattern $\boldsymbol{P}^*$ in $\{0, \star\}^N$ such that for $j \in [N]$, $P^*_j = \star$ if $j \in S$ and $P^*_j = 0$ otherwise. Then define the pattern $\boldsymbol{P}^{\mathsf{ind}} \in \{0, \star\}^N$ such that for $j \in [N]$, $P^{\mathsf{ind}}_j = 0$ if $j <$ ind and $P^{\mathsf{ind}}_j = \star$ otherwise. Define $\boldsymbol{P} \in \{0, \star\}^N$ such that for $j \in [N]$, $P_j = P^*_j \wedge P^{\mathsf{ind}}_j$ with the following rule : $\star \wedge 0 = 0$. Finally compute $\mathsf{ct} = ct = Encrypt(pk, \boldsymbol{P}, \mathsf{m})$ and outputs ct.

Decrypt(pk, sk_i, ct): compute $\mathsf{m}^{'} \leftarrow Decrypt(sk_i, ct)$.

Figure 6.10: Generic construction of augmented broadcast encryption scheme from identity-based encryption with wildcards scheme.

Proof 6.1.5 *$\boldsymbol{P}^i \in_\star \boldsymbol{P}$ implies that $P^i_i = P_i$ or $P_i = \star$. As $P^i_i = 1$, we have that $P^i = \star$ and thus $i \notin \bar{W}(\boldsymbol{P})$, i.e. $i \in S \wedge i \geq$ ind. Suppose that $i \in S \wedge i \geq$ ind. By construction for all $j \in [N], j \neq i$, $P^i_j = 0$ and either $P^i_j = 0 = P_j$ or $P_j = \star$, and $P_i = \star$, i.e. $\boldsymbol{P}^i \in_\star \boldsymbol{P}$. Then correctness follows from WIBE's correctness.* □

Theorem 6.1.6 *If the underlying identity-based encryption with wildcards scheme satisfies adaptive* IND-WIBE *security, then the obtained augmented broadcast encryption scheme satisfies adaptive message-hiding security.*

Proof 6.1.6 *Let $\mathcal{B}$ be an adversary against message-hiding security, that wins with non negligible advantage. In Figure 6.11 we construct $\mathcal{A}$ an adversary against* IND-WIBE *that uses $\mathcal{B}$ and wins with non negligible advantage. Let $\mathcal{C}$ be a challenger.*

`SETUP`: $\mathcal{C}$ on inputs λ, N runs Setup$(\lambda, N) \rightarrow (\mathsf{msk}, \mathsf{pk})$ and sends pk to $\mathcal{A}$, who sends it to $\mathcal{B}$.

`KEY QUERY`: $\mathcal{B}$ chooses $i \in [N]$, sends it to $\mathcal{A}$ who creates the pattern $\boldsymbol{P}^i$ such that for $j \in [N]$, $P^i_j = 1$ if $i = j$, $P^i_j = 0$ otherwise. $\mathcal{A}$ sends $\boldsymbol{P}^i$ to $\mathcal{C}$, who responds with $\mathsf{sk}_{\boldsymbol{P}^i} \leftarrow$ KeyDer$(\mathsf{msk}, \boldsymbol{P}^i)$. $\mathcal{A}$ sends $\mathsf{sk}_{\boldsymbol{P}^i}$ to $\mathcal{B}$ as sk_i.

`CHALLENGE`: $\mathcal{B}$ chooses two challenge messages $\mathsf{m}_0, \mathsf{m}_1$ and a challenge set S^*. It sends $\mathsf{m}_0, \mathsf{m}_1, S^*$ to $\mathcal{A}$, who creates pattern $\boldsymbol{P}^*$, such that for $j \in [N]$, $P^*_j = 0$. $\mathcal{A}$ sends $\mathsf{m}_0, \mathsf{m}_1, P^*$ to $\mathcal{C}$, who chooses $b \leftarrow \{0, 1\}$ randomly and runs $\mathsf{ct}^* \leftarrow$ Encrypt$(\mathsf{pk}, \boldsymbol{P}^*, \mathsf{m}_b)$. $\mathcal{C}$ gives ct^* to $\mathcal{A}$, who sends it to $\mathcal{B}$.

`KEY QUERY`: $\mathcal{A}, \mathcal{B}, \mathcal{C}$ act like in the previous `KEY QUERY` step.

`GUESS`: $\mathcal{B}$ outputs its guess $b^{'}$ to $\mathcal{A}$, who outputs it as its guess.

Figure 6.11: Construction of IND-WIBE adversary from MH-AugBE adversary.

If all $\mathcal{B}$'s queries satisfy the game constraints, then all $\mathcal{A}$'s queries have the same property. Thus $\mathcal{A}$'s simulation is perfect and the advantage of $\mathcal{A}$ is the same as $\mathcal{B}$'s. This concludes the proof. □

Note 6.1.8 *Pattern $\boldsymbol{P}^*$ is equal to $\mathbf{0}^N$. Then, for all $i \in [N]$, $\boldsymbol{P}^i \notin_\star \boldsymbol{P}^*$: the WIBE adversary's constraint is always verified and we do not specify it in the proof.*

Theorem 6.1.7 *If the underlying WIBE satisfies adaptive pattern-hiding security, then the obtained AugBE scheme satisfies adaptive anonymous security.*

Proof 6.1.7 *Let $\mathcal{C}$ be a challenger and $\mathcal{B}$ be an adversary that wins the anonymous security game with non negligible advantage. We construct, in Figure 6.12, an adversary $\mathcal{A}$ that uses $\mathcal{B}$ and wins the pattern-hiding security game with non negligible advantage.*

`SETUP`: $\mathcal{C}$ on inputs λ, N runs Setup$(\lambda, N) \rightarrow (\mathsf{msk}, \mathsf{pk})$ and sends pk to $\mathcal{A}$, who sends it to $\mathcal{B}$.
`KEY QUERY`: $\mathcal{B}$ chooses $i \in [N]$, sends it to $\mathcal{A}$ who creates the pattern $\boldsymbol{P}^i$ such that for $j \in [N]$, $P^i_j = 1$ if $i = j$, $P^i_j = 0$ otherwise. $\mathcal{A}$ sends $\boldsymbol{P}^i$ to $\mathcal{C}$, who responds with $\mathsf{sk}_{\boldsymbol{P}^i} \leftarrow$ KeyDer$(\mathsf{msk}, \boldsymbol{P}^i)$. $\mathcal{A}$ sends $\mathsf{sk}_{\boldsymbol{P}^i}$ to $\mathcal{B}$ as sk_i.
`CHALLENGE`: $\mathcal{B}$ chooses a challenge message m, two challenge sets S^0, S^1 and sends m, S^0, S^1 to $\mathcal{A}$. The latter creates the patterns $\boldsymbol{P}^0$, $\boldsymbol{P}^1$ such that for $j \in [N]$, $P^0_j = \star$ if $j \in S^0$, $P^0_j = 0$ otherwise, and $P^1_j = \star$ if $j \in S^1$, $P^1_j = 0$ otherwise. $\mathcal{A}$ sends $\mathsf{m}, \boldsymbol{P}^0, \boldsymbol{P}^1$ to $\mathcal{C}$. $\mathcal{C}$ chooses $b \leftarrow \{0, 1\}$ randomly and sets $\mathsf{ct}^* \leftarrow$ Encrypt$(\mathsf{pk}, \boldsymbol{P}^b, \mathsf{m})$. It sends ct^* to $\mathcal{A}$ who sends it to $\mathcal{B}$.
`KEY QUERY`: $\mathcal{A}$ and $\mathcal{B}$ act like in the previous `KEY QUERY` step. $\mathcal{C}$ runs KeyDer$(\mathsf{msk}, \boldsymbol{P}^i) \rightarrow \mathsf{sk}_{\boldsymbol{P}^i}$ and sends $\mathsf{sk}_{\boldsymbol{P}^i}$ to $\mathcal{A}$ who sends it as sk_i to $\mathcal{B}$.
`GUESS`: $\mathcal{B}$ outputs its guess b' to $\mathcal{A}$, who outputs it as its guess.

Figure 6.12: Construction of PH-WIBE adversary from ANO-AugBE adversary.

If all $\mathcal{B}$'s queries satisfy the game constraint, then all $\mathcal{A}$'s queries have the same property. Thus $\mathcal{A}$'s simulation is perfect, and the advantage of $\mathcal{A}$ is the same as $\mathcal{B}$'s. This concludes the proof. □

Combining theorem 6.1.1 and 6.1.7 we obtain that if the underlying WIBE satisfies adaptive pattern-hiding security then the built AugBE scheme satisfies adaptive index-hiding security.

6.1.4 Concrete (Augmented) Broadcast Encryption Schemes

In this section we present our new broadcast encryption scheme and augmented broadcast encryption scheme, both obtained thanks to the combination of our generic constructions and our identity-based encryption with wildcards schemes presented in Section 4.3.

A New Broadcast Encryption Scheme. Using our generic construction in Section 6.1.3 and our first identity-based encryption with wildcards scheme (Section 4.3, Figure 4.9) with $L = N$ we obtain an instantiation of a broadcast encryption scheme that we present in Figure 6.13.

Setup(λ, N): generate an asymmetric bilinear pairing group $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ for prime order p. Sample random dual orthonormal bases $(\mathbb{D}, \mathbb{D}^*) \leftarrow$ Dual$(\mathbb{Z}_p^4)$. Let $\boldsymbol{d}_1, \cdots, \boldsymbol{d}_4$ denote the elements of $\mathbb{D}$ and $\boldsymbol{d}_1^*, \cdots, \boldsymbol{d}_4^*$ denote the elements of $\mathbb{D}^*$. Pick $\alpha, a_1, \cdots, a_N \leftarrow \mathbb{Z}_p$. The master secret key is msk $= (\alpha, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_2^*}, a_1, \cdots, a_N)$ and the public key is

$$\text{pk} = (\Gamma, p, e(g_1, g_2)^{\alpha \boldsymbol{d}_1 \cdot \boldsymbol{d}_1^*}, g_1^{\boldsymbol{d}_1}, \boldsymbol{h}_1 = g_1^{a_1 \cdot \boldsymbol{d}_2}, \cdots, \boldsymbol{h}_N = g_1^{a_N \cdot \boldsymbol{d}_2}).$$

KeyGen(msk, $i \in [N]$): pick $r \leftarrow \mathbb{Z}_p$. Compute $\boldsymbol{a} = g_2^{\alpha \boldsymbol{d}_1^* + r \cdot a_i \cdot \boldsymbol{d}_1^* - r \cdot \boldsymbol{d}_2^*}$ and $\boldsymbol{b}_j = g_2^{r \cdot a_j \cdot \boldsymbol{d}_1^*}$ for $j \in [N] \setminus \{i\}$. The secret key is $\text{sk}_i = (\boldsymbol{a}, \{\boldsymbol{b}_j\}_{j \in [N] \setminus \{i\}})$.

Encrypt(pk, S, m $\in \mathbb{G}_T$): choose $s \leftarrow \mathbb{Z}_p$ and compute ct $= (c_1, \boldsymbol{c}_2)$ where $c_1 = \text{m} \cdot (e(g_1, g_2)^{\alpha \boldsymbol{d}_1^* \cdot \boldsymbol{d}_1})^s$, $\boldsymbol{c}_2 = g_1^{s \boldsymbol{d}_1} \cdot \prod_{j \in S} \boldsymbol{h}_j^s$.

Decrypt(sk$_i$, ct, S): compute $\boldsymbol{a}' = \boldsymbol{a} \prod_{j \in S \cap ([N] \setminus \{i\})} \boldsymbol{b}_j$ and finally $\boldsymbol{C}_1 \cdot \frac{1}{e(\boldsymbol{c}_2, \boldsymbol{a}')}$.

Figure 6.13: Our adaptively secure broadcast encryption scheme in prime order group, with constant size ciphertext.

In Table 6.1, we give a comparison between our broadcast encryption scheme and existing schemes.

based on that table, we remark that our scheme is not as efficient as [9]'s scheme, which is currently the most efficient BE scheme in the literature. However, our scheme satisfies the stronger adaptive security notion, and is proven secure under standard assumption. Compare to the adaptively secure scheme given in [71], we have a bigger user secret key (sk$_i$) size, but a shorter public key (pk) size. To be exhaustive, [41] proposed a scheme with all parameters in poly$(\log(N))$, with adaptive security. However, this scheme is using multilinear maps and its security is proven in the generic group model (see paragraph 3.1.4) [42] proposed a scheme with all parameters in poly$(N, \log(N))$

Table 6.1: Broadcast Encryption schemes comparison; “GGM”, “Sym” and “Asym” stand for “Generic Group Model”, “Symmetric” and “Asymmetric” respectively. Here $t \in \mathbb{N}$, such that t divides N.

Scheme	|pk|	$\|sk_i\|$	|ct|	Security	Assumption	Model	Settings
[38]	$O(N)$	$O(1)$	$O(1)$	Selective	N-BDHE	GGM	Sym pairings
[49]	$O(t+N/t)$	$O(N/t)$	$O(t)$	Adaptive	k−Lin	Standard	Asym pairings
[71]	$O(N^2)$	$O(1)$	$O(1)$	Adaptive	k−Lin	Standard	Asym pairings
[9]	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$	Selective	LWE, KOALA	Standard	Lattices
Ours	$O(N)$	$O(N)$	$O(1)$	Adaptive	SXDH	Standard	Asym pairings

using lattices, but no security proof is given.

With our second identity-based encryption with wildcards scheme (Figure 4.11) and our generic constructions, setting $L = N$, we obtain a new anonymous broadcast encryption scheme. Our scheme does not improve the efficiency of the Libert *et al.* scheme [100], which is the best known so far. In particular, their scheme has pk and sk_i sizes in respectively $O(N)$ and $O(1)$ while in our scheme the same parameters have sizes in $O(N^2)$ and $O(N)$ respectively. Regarding security, their scheme achieves the stronger CCA security in the standard model while we only reach a CPA security. However, in Libert *et al.*'s scheme, each user has to try each element of the ciphertext to find the one she can truly decrypt, while this is not necessary in our construction.

Note 6.1.9 *In the anonymous broadcast encryption setting, notice that [100] said that achieving shorter than linear size for ciphertext is impossible when considering the used users set description as part of the ciphertext.*

A New Augmented Broadcast Encryption Scheme. Using our generic construction and our second identity-based encryption with wildcards scheme (Figure 4.11) with $L = N$ we obtain an instantiation of an augmented broadcast encryption scheme, presented in Figure 6.14.

Our augmented broadcast encryption scheme is the first proven adaptively secure in the standard model. We compare it to the literature in Table 6.2.
Notice the augmented broadcast encryption scheme obtained from our second WIBE instantiation is actually the first known anonymous AugBE.

Setup(λ, N): generate an asymmetric bilinear pairing group $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$ for prime order p. Sample random dual orthonormal bases $(\mathbb{D}, \mathbb{D}^*) \leftarrow \text{Dual}(\mathbb{Z}_p^{4N+2})$. Let $\boldsymbol{d}_0, \cdots, \boldsymbol{d}_{4N+1}$ denote the elements of $\mathbb{D}$ and $\boldsymbol{d}_0^*, \cdots, \boldsymbol{d}_{4N+1^*}$ denote the elements of $\mathbb{D}^*$. Pick $\alpha \leftarrow \mathbb{Z}_p$. The master secret key is $\text{msk} = (\alpha, g_2^{\boldsymbol{d}_0^*}, g_2^{\boldsymbol{d}_1^*}, \cdots g_2^{\boldsymbol{d}_N^*}, g_2^{\boldsymbol{d}_{3N+1}^*} \cdots, g_2^{\boldsymbol{d}_{4N}^*})$ and the public key is

$$\text{pk} = (\Gamma, p, e(g_1, g_2)^{\alpha \boldsymbol{d}_0 \cdot \boldsymbol{d}_0^*}, g_1^{\boldsymbol{d}_0}, g_1^{\boldsymbol{d}_{4L+1}}, \boldsymbol{h}_1 = g_1^{\boldsymbol{d}_1}, \cdots, \boldsymbol{h}_N = g_1^{\boldsymbol{d}_N}).$$

For each $i \in [N]$, pick $r \in \mathbb{Z}_p, \boldsymbol{\eta} \in \mathbb{Z}_p^L$ and set secret key sk_i to $g_2^{\alpha \boldsymbol{d}_0^* + r \boldsymbol{d}_i^* + \sum_{l=1}^{N} \eta_l \cdot \boldsymbol{d}_{3N+l}^*}$.

Encrypt(pk, S, ind, $\text{m} \in \mathbb{G}_T$): choose $s_1, s_2, s_3 \leftarrow \mathbb{Z}_p$ and compute $\text{ct} = (c_1, \boldsymbol{c}_2)$ where

$$c_1 = \text{m} \cdot (e(g_1, g_2)^{\alpha b_0^* \cdot b_0})^{s_1}, \qquad \boldsymbol{c}_2 = g_1^{s_1 \boldsymbol{d}_0 + s_2 \boldsymbol{d}_{4N+1}} \cdot \prod_{j \in [N] \setminus S, j \geq \text{ind}} \boldsymbol{h}_j^{s_3}.$$

Decrypt(sk_i, ct): compute $c_1 \cdot \frac{1}{e(\boldsymbol{c}_2, \text{sk}_i)}$.

Figure 6.14: Our adaptively anonymous augmented broadcast encryption scheme in prime order group.

Table 6.2: Augmented Broadcast Encryption schemes comparison; "Multi" and "PWE" mean respectively "Multilinear Map" and "Positional Witness Encryption".

Scheme	Enc Algo	|pk|	$\|\text{sk}_i\|$	|ct|	Security	Model	Settings
[40]	1	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(\sqrt{N})$	Adaptive	GGM	Pairing c.o.
[75]	2	$\Omega(N)$	$\Omega(N^2)$	$O(N^\epsilon)$	Selective	GGM	Pairing, lattices
[77]	1	poly(λ)	poly(λ)	poly(λ)	Adaptive	Multi	PWE
Ours	1	$O(N^2)$	$O(N)$	$O(N)$	Adaptive	Standard	Pairings p.o.

Broadcast and Trace Scheme. There exists another variant of broadcast encryption, called Broadcast and Trace (BT) [112]. Such a scheme is a primitive that combines broadcast encryption scheme and traitor tracing (TT) [52] schemes. We recall that the latter is a primitive in which a message is encrypted for the whole subset $[N]$ but if some subset of traitors uses their secret keys to produce a pirate decoder, then the tracing procedure can identify at least one of the traitors.
In [40, 77], it was demonstrated that a broadcast and trace scheme can be constructed from any message and index-hiding AugBE. Briefly, the idea of the construction is that for the broadcast part the AugBE encryption algorithm is used with index ind $= 1$, while the traitor tracing part is done by running the AugBE encryption algorithm several times,

with different indices. Let us see how the traitor tracing part works. Suppose that one has a black box access to a pirate decoder, meaning that she can learn the result of decryption output by the decoder. She then encrypts several messages with index equals to 1, then to 2, etc. At some point, for index $j \in [N]$ the decoder will output a correct decryption while for index $j+1$ the decryption will fail. It means that the (or one of the) traitor(s) has an index that is greater or equal to j but strictly lower than $j+1$, thus traitor's index is equal to j.
As for traitor tracing, broadcast and trace schemes can achieve either public (anyone can find the traitors) or private (traitors can only be retrieved by the owner of a specific master key) traceability, and both cases are useful for different kinds of use cases. Theoretically speaking, public traceability is however known to be harder to achieve [39]. By construction, the original (i.e. given by Boneh and Waters) AugBE definition [40] gives a publicly traceable broadcast and trace scheme while Goyal *et al.* [75]'s definition with two encryption algorithms is suitable for the private case.

Following the generic construction given in [40, 77], our augmented broadcast encryption scheme can itself be turned into a broadcast and trace scheme. Table 6.3 gives a comparison between our resulting broadcast and trace scheme and existing ones. We also specify in this table if the users set used for encryption must be given additionally to the ciphertext, in order to make the decryption working. A "$\times$" means that the set does not have to be given.

Table 6.3: Broadcast and Trace schemes comparison; tk, "p.o", "c.o", "PWE", "std" "Multi", "P" and "S" mean tracing key, "prime order" "composite order", "Positional Witness Encryption", "standard", "multilinear", "public" and "secret respectively, $0 < \epsilon \leq 1/2$. In the column "Users set" a "$\surd$" indicates that the set is given along with the ciphertext while a "$\times$" means that it is not the case.

Scheme	|pk|	|sk$_i$|	|ct|	Users set	Security	Model	tk	Settings
[40]	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(\sqrt{N})$	$\surd$	Adaptive	GGM	P	Pairing c.o.
[75]	$\Omega(N)$	$\Omega(N^2)$	$O(N^\epsilon)$	$\surd$	Selective	GGM	S	Pairing, lattices
[77]	poly(λ)	poly(λ)	poly(λ)	$\surd$	Adaptive	Multi	P	PWE
[139]	$O(N)$	$O(N)$	$O(1)$	$\surd$	Adaptive	GGM	S	Pairings p.o.
Ours	$O(N^2)$	$O(N)$	$O(N)$	$\times$	Adaptive	Std	P	Pairings p.o.

As we can see our scheme is the first BT scheme (as far as we know) that does not need the description of the user sets to be able to decrypt, and that has security proven in the standard model. Moreover, our scheme is publicly traceable, and uses pairings in prime order group while other existing publicly traceable schemes are using either pairings in composite order group (less secure), or positional witness encryption. Regarding efficiency, our resulting BT scheme has a complexity similar to a "trivial" scheme [120] (with all parameters sizes linear in the number of users). However, the claimed of our

work is not to provide a new efficient Broadcast and Trace scheme, but a new generic way to build AugBE schemes, and our generic construction could be more efficient than a “trivial” BT scheme, even if our current instantiation is not.

Moreover, we also consider that our proposal has the additional feature of anonymity, that the trivial construction could not have without being less efficient than ours. With such property, the users set is included in the ciphertext and no longer given in the clear which leads to a linear additional computational cost during decryption. Anonymity in the context of BT seems to be an overkill, but we think that for applications in which being in the used users set reveals some private information about users, it might be a real interest to use an anonymous scheme. Eventually, the derivation of our anonymous AugBE to an anonymous BT scheme is quite direct from the generic construction given in [40]. A formal definition of an anonymous BT scheme is quite straightforward from the one of anonymous AugBE and can be found in *e.g.*, [10]. The only existing anonymous BT is the one of [10], which is based on the anonymous BE scheme of [100]: it directly inherit advantages and drawbacks compare to our resulting scheme. Hence, if our new instantiation of a BT scheme is not more efficient than the “trivial” scheme, it has some specific features that could not be obtained so easily “trivially”. Notice that Blazy *et al.* [29] proposed an anonymous trace and revoke broadcast encryption scheme, but their definition differs slightly from the one we use, therefore we do not take into account their scheme in our comparison as it would not make sense.

6.2 Attribute-Based Encryption

The second data sharing scheme we study in this manuscript is *attribute-based encryption* (ABE) scheme, introduced by Sahai and Waters [126] in 2005. In such encryption scheme, secret keys and ciphertexts are associated to some subset of attributes, and decryption is possible if there exists a relation between the secret key’s attributes and the ciphertext’s attributes. In more details, in a *Ciphertext Policy ABE* (CP-ABE) the ciphertext is associated to an access policy while the secret key is associated to a set of attributes. Decryption becomes possible if the set of attributes satisfies the policy. There exists the dual of a CP-ABE, known as *Key Policy ABE* (KP-ABE) in which the roles of attributes and access policies are swapped. There exist several ways to define an access policy in the literature: through threshold structure [126], tree-based structure [78], boolean formulas [94], linear secret sharing schemes [136], circuits [36], ... The main aim of research in ABE is to build efficient schemes in terms of both time and space complexities, while supporting complex access policies. Unfortunately, most existing schemes propose ciphertexts with a size linear in the number of attributes in

the scheme [78, 92, 91], while some other constructions succeed in proposing constant size ciphertext, but at the cost of quadratic-size user private key [13].
In the sequel we first formally define attribute-based encryption and its properties. Then in Section 6.2.2 we present one of our contributions, which is the construction of a ciphertext policy attribute-based encryption scheme from *dually computable* accumulators. In Section 6.2.3, we detail the obtained CP ABE scheme, prove its correctness and security and compare it to the state of the art. We end this section with a key policy version of our attribute-based encryption scheme, and compare it to existing schemes, in Section 6.2.4.

6.2.1 Definitions and Properties

Definition 6.2.1 ***Ciphertext policy attribute-based encryption*** *(CP-ABE) [126]. A ciphertext policy attribute-based encryption scheme consists of four algorithms.*

- $\mathsf{Setup}(\lambda)$*: the setup algorithm takes as input a security parameter* λ *and outputs a master public key pk and a master secret key msk.*
- $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, \Upsilon)$*: the key generation algorithm takes as input a master public key pk, a master secret key msk, a key attribute* Υ *and outputs a private key* sk_{Υ}*.*
- $\mathsf{Encrypt}(\mathsf{pk}, \Pi, \mathsf{m})$*: the encryption algorithm takes as input a master public key pk, an access policy* Π*, and a message m. It outputs a ciphertext* ct_{Π}*.*
- $\mathsf{Decrypt}(\mathsf{pk}, \mathsf{sk}_{\Upsilon}, \Upsilon, \mathsf{ct}_{\Pi}, \Pi)$*: the decryption algorithm takes as input a master public key pk, a private key* sk_{Υ} *along with an associated set of attributes* Υ*, a ciphertext* ct_{Π} *and its associated access policy* Π*, and it returns a message* m'*.*

Definition of key policy attribute-based encryption can easily be obtained from the above definition, with the roles of attributes sets and access policies swapped.

Definition 6.2.2 ***Correctness.*** *A CP-ABE scheme is said to be* correct *if for all security parameter* $\lambda \in \mathbb{N}$*, every attributes set* Υ*, every access policy* Π *such that* Υ *satisfies* Π*, every messages m, and every honestly generated keys* $(pk, msk) \leftarrow \mathsf{Setup}(\lambda)$ *and* $sk_{\Upsilon} \leftarrow \mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, \Upsilon)$*:*

$$\Pr\left[\mathsf{Decrypt}(\mathsf{pk}, \mathsf{sk}_{\Upsilon}, \Upsilon, \mathsf{Encrypt}(\mathsf{pk}, \Pi, \mathsf{m}), \Pi) = \mathsf{m}\right] = 1.$$

Note 6.2.1 *ABE schemes can be* bounded*, meaning that during the setup phase a bound in the number of attributes allowed in the scheme is given and keys and ciphertexts can be created for an arbitrarily number of attributes at the condition that this number is lower than a given.*

Definition 6.2.3 ***Adaptive indistinguishability security*** *(Ada-IND). A CP-ABE scheme is said to satisfy* adaptive indistinguishability *security if all PPT adversaries $\mathcal{A}$ have at most negligible advantage in the game presented in Figure 6.15, where $\mathcal{C}$ is a challenger and $\mathcal{A}$'s advantage is defined as* $\mathsf{Adv}_{\mathcal{A}}^{Ada-IND}(\lambda) := \left|\Pr\left[b^{'} = b\right] - \frac{1}{2}\right|$.

SETUP: on input λ, $\mathcal{C}$ samples $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(\lambda)$ and gives pk to $\mathcal{A}$.
KEY QUERY: $\mathcal{A}$ chooses an attributes set Υ and sends it to $\mathcal{C}$ who replies with $\mathsf{sk}_{\Upsilon} \leftarrow \mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, \Upsilon)$.
CHALLENGE: $\mathcal{A}$ submits a pair of equal length challenge messages $\mathsf{m}_0, \mathsf{m}_1$ and a challenge access policy Π^* to $\mathcal{C}$. The latter samples $b \leftarrow \{0,1\}$ randomly and replies to $\mathcal{A}$ with $\mathsf{ct}_{\Pi^*} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \Pi^*, \mathsf{m}_b)$.
KEY QUERY: $\mathcal{A}$ chooses an attributes set Υ and sends it to $\mathcal{C}$ who replies with $\mathsf{sk}_{\Upsilon} \leftarrow \mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, \Upsilon)$.
GUESS: $\mathcal{A}$ outputs her guess $b^{'} \in \{0,1\}$ for b, and wins the game if $b^{'} = b$ and if, for all attributes set Υ for which a key was queried, Υ does not satisfy Π^*.

Figure 6.15: Adaptive indistinguishability security game for ciphertext policy attribute-based encryption schemes.

Again, adaptive indistinguishability security for key policy attribute-based encryption schemes can easily be derived from the above definition, by switching the roles of attributes sets and access policies.

6.2.2 CP-ABE From Dually Computable Accumulators: The Different Steps of Our Construction

In this section we propose a way to obtain an ABE scheme for which both the ciphertext and the user secret key are constant, while obtaining very good time complexities. To reach such objective of compactness, our idea is to employ our notion of dually computable accumulators in the following manner: the secret key, computed by the authority, corresponds to a privately computed accumulator of the users' attributes set, while the encryption corresponds to a one-time-pad with a mask derived from a publicly computed accumulator of the access policy. Decryption is then possible if the decryptor can demonstrate that the intersection of their accumulator and the one associated with the ciphertext is not empty, utilizing membership witnesses for both the privately computed and the publicly computed accumulators. However, while it is relatively straightforward to use accumulators to represent sets of attributes, understanding how they can serve as a concise representation of access policies is more complex. In this study, we introduce a way to represent monotone boolean formulas that is compatible with the use of accumulators, and then show how to employ our dually

computable accumulator to obtain a compact, efficient and secure ABE. Unfortunately as our construction relies on pairing-based accumulators' specific features we are not able to propose a generic construction of attribute-based encryption scheme from dually computable accumulators.

Basic idea. Having both private evaluation and public witness creation permits us to transform a cryptographic accumulator into an encryption scheme. More precisely, in our CP-ABE, the user secret key is a privately computed accumulator $\text{acc}_{\mathcal{X}} = g_1^{\boldsymbol{d}_1 \prod_{x \in \mathcal{X}} (x+s)}$, where $\mathcal{X}$ is a representation of the user's attributes. In parallel, the ciphertext is a one-time-pad between the message m and a mask $\boldsymbol{H}$ that is computed using a publicly computable accumulator $\text{accp}_{\mathcal{Y}}$, where $\mathcal{Y}$ is a representation of the access policy. However, with the dually computable accumulator of the previous section as given in Figure 5.10, this construction is not efficient and secure, thus we have to make some changes on the accumulator scheme. Before going into those details, we first explain how we can define $\mathcal{X}$ and $\mathcal{Y}$. In the sequel let $Q = 2^q - 1$, where $q \in \mathbb{N}$ is the bound on the number of attributes in the ABE.

Representation of boolean formulas and attributes with cryptographic accumulators. In our ABE, access policies are expressed as disjunctions of conjunctions (DNF), without "NO" gates. Hence, a policy could be written $\Pi = \pi_1 \vee \pi_2 \vee \cdots \vee \pi_l$, where $l \in \mathbb{N}$, and π_i is a conjunction of attributes. Let $\mathcal{Y}_i$ be the set of attributes present in clause π_i, for $i = 1, \cdots, l$. Our idea is to define $\mathcal{Y}$ as the set $\{\mathcal{H}(\mathcal{Y}_i)\}_{i=1}^{l}$, where $\mathcal{H}$ is a hash function that takes as input a set of elements and returns an element in $\mathbb{Z}_p$, for a prime p. During the encryption process, we create the accumulator $\text{accp}_{\mathcal{Y}}$ using PublicEval (see below).

For a set Υ of attributes for a given user, we create $\mathcal{X}$ as the set of hash values (using $\mathcal{H}$) of all non-empty subsets of Υ[1]. During the key generation process, the authority hence creates the accumulator $\text{acc}_{\mathcal{X}}$ using Eval.

Encryption and decryption. For a given user, if her set of attributes Υ satisfies the policy Π, it means that there exists a non-empty subset of Υ that corresponds to a clause π_i in Π. As $\mathcal{H}$ is deterministic, it follows that one element, called ξ in the sequel, is present in both accumulators: $\text{acc}_{\mathcal{X}}$ (the one corresponding to the non-empty subsets of Υ) and $\text{accp}_{\mathcal{Y}}$ (the one that corresponds to Π). Based on that, we propose that during the encryption process, the mask $\boldsymbol{H}$ is computed using the public verification equation

[1] It follows that if $|\Upsilon| = k$, then $|\mathcal{X}| = 2^k - 1$.

PublicVerify, as $e(g_1^{\boldsymbol{d}_1}, \text{accp}_{\mathcal{Y}})^{\alpha}$, where α is some randomness.

During decryption, a user having a valid set of attributes precisely knows both the clause π_i and the element in Υ that match together. The next step is then for the user to generate a witness for such element, and thanks to the verification algorithms, retrieve $\boldsymbol{H}$ and then the message. But as both accumulators are not related to the same sets, we cannot directly use the properties of a dually computable accumulator. The user hence needs to compute two witnesses (one for each accumulator), and we need to find a way to combine them appropriately for the decryption to work.

Let $\text{wit}_{\xi}^{\mathcal{Y}}, \text{wit}_{\xi}^{\mathcal{X}}$ be membership witnesses for ξ and respectively $\mathcal{Y}$, and $\mathcal{X}$. For verification, we compute $A = e(\text{wit}_{\xi}^{\mathcal{Y}}, (g_2^{\boldsymbol{d}_2^* \alpha})^{\xi} \cdot g_2^{\boldsymbol{d}_2^* s\alpha})$ and $B = e(\text{wit}_{\xi}^{\mathcal{X}}, (g_2^{\boldsymbol{d}_2^* \alpha})^{\xi} \cdot g_2^{\boldsymbol{d}_2^* s\alpha})$. Multiplying A and B gives us

$$e(\text{wit}_{\xi}^{\mathcal{Y}} \cdot \text{wit}_{\xi}^{\mathcal{X}}, (g_2^{\boldsymbol{d}_2^* \alpha})^{\xi} \cdot g_2^{\boldsymbol{d}_2^* s\alpha})$$

thanks to bilinear pairing properties. To prove simultaneously that ξ is in both accumulators, we need to "force" the decryptor to compute $e(A \cdot B, (g_2^{\boldsymbol{d}_2^* \alpha})^{\xi} \cdot g_2^{\boldsymbol{d}_2^* s\alpha})$ instead of $e(A, (g_2^{\boldsymbol{d}_2^* \alpha})^{\xi} \cdot g_2^{\boldsymbol{d}_2^* s\alpha}) \cdot (B, (g_2^{\boldsymbol{d}_2^* \alpha})^{\xi} \cdot g_2^{\boldsymbol{d}_2^* s\alpha})$.

An easy way to do that is to give with the ciphertext $(A \cdot B)^{\alpha}$ instead of $g_2^{\boldsymbol{d}_2^* \alpha}, g_2^{\boldsymbol{d}_2^* s\alpha}$. But this implies to know witnesses during encryption whereas they are only known during decryption. Our idea is then to "anticipate" the witnesses or at least a part of them.

Notice that for any set $S = \{s_1, \cdots, s_T\}$, its polynomial representation $\prod_{i=1}^{T}(s + Z)$ is actually composed of the elementary symmetric polynomials for T variables: $\sigma_1 = \sum_{i=1}^{T} s_i, \cdots, \sigma_T = \prod_{i=1}^{T} s_i$ (see Definition 2.1.3). Indeed, $\prod_{i=1}^{T}(s + Z) = Z^T + \sigma_1 Z^{T-1} + \cdots \sigma_T$ (see Note 2.1.1). Thus, if we know one element $\tilde{s}$ of S, we know that $\tilde{s}$ is a factor of σ_T. We use this idea to anticipate a part of both witnesses for element ξ.

Let $\{c_i, t_i\}_{i=0}^{Q}$ be respectively the coefficients of $\text{Ch}_{\mathcal{X}\setminus\{\xi\}}[Z]$ and $\text{Ch}_{\mathcal{Y}\setminus\{\xi\}}[Z]$. Our first idea is to separate coefficients c_0 and t_0 of the others. Thus, in our scheme for clause π_j (and associated set $\mathcal{Y}_j$) and non-empty subset p_{j^*}, a witness that $\xi = \mathcal{H}(p_{j^*})$ is accumulated in $\text{acc}_{\mathcal{X}}$ is now equal to $(g_1^{\boldsymbol{d}_1 c_0}, g_2^{\boldsymbol{d}_2^* \sum_{i=1}^{Q} c_i s^i})$ and a witness that $\xi = \mathcal{H}(\mathcal{Y}_j)$ is accumulated in $\text{accp}_{\mathcal{Y}}$ is now equal to $(g_1^{\boldsymbol{d}_1 t_0}, g_2^{\boldsymbol{d}_2^* \sum_{i=1}^{Q} t_i s^i})$. This gives us our first intermediate accumulator, presented in Figure 6.16. In orange we highlight the differences with our accumulator

presented in Figure 5.10, in Section 5.4.2.

- Gen(λ, q): run a bilinear group generation algorithm to get $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$. Then choose a random $s \leftarrow \mathbb{Z}_p^*$, and run Dual$(\mathbb{Z}_p^2)$ to get $\mathbb{D} = (\boldsymbol{d}_1, \boldsymbol{d}_2), \mathbb{D}^* = (\boldsymbol{d}_1^*, \boldsymbol{d}_2^*)$. Let $\psi \in \mathbb{Z}_p^*$ be the random such that $\boldsymbol{d}_1 \cdot \boldsymbol{d}_1^* = \boldsymbol{d}_2 \cdot \boldsymbol{d}_2^* = \psi$. Set $\mathsf{sk}_{\mathsf{acc}} = (s, \mathbb{D}, \mathbb{D}^*)$, and

$$\mathsf{pk}_{\mathsf{acc}} = \left(\begin{array}{c} \Gamma, g_1^{\boldsymbol{d}_1}, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 s}, \cdots, g_1^{\boldsymbol{d}_2 s^q}, g_2^{\boldsymbol{d}_1^*}, \\ g_2^{\boldsymbol{d}_1^* s}, \cdots, g_2^{\boldsymbol{d}_1^* s^q}, g_2^{\boldsymbol{d}_2^* s}, \cdots, g_2^{\boldsymbol{d}_2^* s^q} \end{array} \right).$$

  Return $\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}$.

- Eval$(\mathsf{sk}_{\mathsf{acc}}, \mathcal{X})$: compute the coefficients $\{a_i\}_{i=0,\cdots,q}$ of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (Z + x)$. Then compute $\mathsf{acc}_{\mathcal{X}} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i}$, and return $\mathsf{acc}_{\mathcal{X}}$.

- PublicEval$(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$: compute the coefficients $\{a_i\}_{i=0,\cdots,q}$ of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (Z + x)$. Then compute $\mathsf{accp}_{\mathcal{X}} = g_2^{\boldsymbol{d}_1^* \sum_{i=0}^{q} a_i s^i}$, and return $\mathsf{accp}_{\mathcal{X}}$.

- WitCreate$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}} \backslash \mathsf{accp}_{\mathcal{X}}, \mathcal{X}, \mathcal{I})$: let $\{b_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{X} \setminus \mathcal{I}}[Z] = \prod_{x \in \mathcal{X} \setminus \mathcal{I}} (x + Z)$. Compute $W_1 = g_1^{\boldsymbol{d}_1 b_0}$ and $W_2 = g_2^{\boldsymbol{d}_2^* \sum_{i=1}^{q} b_i s^i}$, and return $\mathsf{wit}_{\mathcal{I}} = (W_1, W_2)$.

- Verify$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{wit}_{\mathcal{I}}, \mathcal{I})$: let $\{c_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{I}}[Z] = \prod_{x \in \mathcal{I}} (x + Z)$ and the return 1 if $e(\mathsf{acc}_{\mathcal{X}}, g_2^{\boldsymbol{d}_1^*}) = e(W_1, g_2^{\boldsymbol{d}_1^* \sum_{i=0}^{q} c_i s^i}) \cdot e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, W_2)$, 0 otherwise.

- PublicVerify$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}}, \mathsf{wit}_{\mathcal{I}}, \mathcal{I})$: let $\{c_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{I}}[Z] = \prod_{x \in \mathcal{I}} (x + Z)$ and return 1 if $e(g_1^{\boldsymbol{d}_1}, \mathsf{accp}_{\mathcal{X}}) = e(W_1, g_2^{\boldsymbol{d}_1^* \sum_{i=0}^{q} c_i s^i}) \cdot e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, W_2)$, 0 otherwise.

Figure 6.16: The first intermediate accumulator scheme.

However, the values c_0 and t_0 depend on $\mathcal{X} \setminus \{\xi\}$ and $\mathcal{Y} \setminus \{\xi\}$ respectively. While $\mathcal{Y}$ is known during encryption, $\mathcal{X}$ and ξ are only known during decryption. Therefore, we cannot anticipate c_0 and t_0 during decryption.

To solve this, we choose two values x_0, y_0 that do not correspond to an output of $\mathcal{H}$ and add them to the sets $\mathcal{X}$ and $\mathcal{Y}$ respectively. As $x_0, y_0 \notin Im(\mathcal{H})$, we know that x_0 and y_0 will always be in sets $\mathcal{X} \setminus \{\xi\}$ and $\mathcal{Y} \setminus \{\xi\}$ respectively. Thus, x_0 is a factor of c_0 while y_0 is a factor of t_0. Therefore, we can anticipate a part of the first element of both witnesses with $g_1^{\boldsymbol{d}_1 x_0}$ for the witness associated to $\text{acc}_{\mathcal{X}}$ and $g_1^{\boldsymbol{d}_1 y_0}$ for the witness associated to $\text{accp}_{\mathcal{Y}}$.

Now that we can anticipate a part of the witnesses, we can combine them by setting $\text{aux}_1 = g_1^{\boldsymbol{d}_1 \alpha (x_0+y_0)}$. Let $\delta, \delta' \in \mathbb{N}$ such that $c_0 = x_0\delta$ and $t_0 = y_0\delta'$. We can compute

$$\begin{aligned} & e(\text{aux}_1^{\delta\delta_0}, (g_2^{\boldsymbol{d}_1^*})^{\xi} \cdot g_2^{\boldsymbol{d}_1^* s}) \\ = \; & e((g_1^{\boldsymbol{d}_1 \alpha (x_0+y_0)})^{\delta\delta'}, g_2^{\boldsymbol{d}_1^*(\xi+s)}) \\ = \; & e(g_1^{\boldsymbol{d}_1 \alpha \delta\delta' (x_0+y_0)}, g_2^{\boldsymbol{d}_1^*(s+\xi)}) \\ = \; & e(g_1, g_2)^{\psi\alpha\delta' c_0(s+\xi)} \cdot e(g_1, g_2)^{\psi\alpha\delta t_0(s+\xi)} \end{aligned}$$

As the verification that $\xi \in \mathcal{X}$ will give $e(g_1, g_2)^{\psi\alpha \sum_{i=0}^{Q} a_i s^i}$ (if ξ is indeed in the set) we have to give in encryption the auxiliary information $\text{aux}_2 = g_2^{-\alpha \boldsymbol{d}_1}$ to remove this extra term and recover the mask. Notice that aux_2 will work only with privately computed accumulator, and will be used for the verification of membership in the accumulator of the secret key.

Managing the randomness α and a constant-size ciphertext. We now have to compute the rest of witness such that it is randomized by α. The trivial solution is to give $g_2^{\alpha \boldsymbol{d}_2^* s}, \cdots, g_2^{\alpha \boldsymbol{d}_2^* s^Q}$ but this will result in a linear size for the ciphertext. Thus, it seems more efficient to give $g_1^{\alpha \boldsymbol{d}_2 (s+\xi)}$. But as ξ is unknown at the time of encryption, we have to give $g_1^{\alpha \boldsymbol{d}_2 s}$ and $g_1^{\alpha \boldsymbol{d}_2}$. With the latter it is possible to cheat: with $g_2^{\boldsymbol{d}_2^*}, g_2^{\boldsymbol{d}_2^* s}, \cdots g_2^{\boldsymbol{d}_2^* s^Q}$ we can compute $g_2^{\boldsymbol{d}_2^* \sum_{i=0}^{Q} m_i s^i}$ and recover the mask.

Our idea to avoid this is to anticipate the value of ξ. We do as we did to anticipate c_0 and t_0. We choose another value z_0 that is not in $Im(\mathcal{H})$ that we add in $\mathcal{X}$ and $\mathcal{Y}$. Then z_0 is an element accumulated in both $\text{acc}_{\mathcal{X}}$ (the secret key) and $\text{accp}_{\mathcal{Y}}$ (used in the mask of the message). During decryption, we prove the membership of $\{\xi, z_0\}$, and thus we need the polynomial $Z^2 + Z(\xi + z_0) + \xi \cdot z_0 = Z^2 + Zz_0 + \xi(Z + z_0)$. Therefore, we give with the ciphertext the auxiliary information $ele_3 = g_1^{\alpha \boldsymbol{d}_2 (z_0 s + s^2)}$ and $ele_4 = g_1^{\alpha \boldsymbol{d}_2 (z_0 + s)}$. As s is secret, there is no way to cheat.

Auxiliary information in the ciphertext. As is, the scheme is not secure. Indeed, from $\text{aux}_1 = g_1^{\boldsymbol{d}_1 \alpha (x_0+y_0)}$ and $\left\{ g_2^{\boldsymbol{d}_1^* s^i} \right\}_{i=0}^{Q}$ (publicly known), anyone can compute $(e(g_1,g_2)^{\psi\alpha \sum_{i=0}^{Q} m_i s^i})^{x_0+y_0}$. As x_0, y_0 are publicly known, anyone can recover $e(g_1,g_2)^{\psi\alpha \sum_{i=0}^{Q} m_i s^i} = \boldsymbol{H}$ and thus the message.

To correct this we set: $\alpha = \alpha_1 \cdot \alpha_2$ for α_1, α_2 two randoms, $\text{aux}_1 = g_1^{\boldsymbol{d}_1 \alpha_2 (x_0+y_0)}$, $ele_3 = g_1^{\alpha_1\alpha_2 \boldsymbol{d}_2 (z_0 s + s^2)}$, $ele_4 = g_1^{\alpha_1\alpha_2 \boldsymbol{d}_2 (z_0+s)}$, and $\text{aux}_2 = g_2^{-\alpha_1\alpha_2 \boldsymbol{d}_1^*}$. To have correctness during membership verification, we need more auxiliary information (ele_1, ele_2) are equal to $(g_2^{\boldsymbol{d}_1^* \alpha_1 (z_0 s + s^2)}, g_2^{\boldsymbol{d}_1^* \alpha_1 (z_0+s)})$.

At this point we obtain a CP-ABE that is working, but unfortunately we are not able to prove its security. Therefore, we have to modify the underlying accumulator, as we explain in the next subsection.

Managing the dual system encryption framework. We want to prove adaptive security of our scheme with the dual system encryption framework, and the decisional subspace assumption in $\mathbb{G}_1$ and $\mathbb{G}_2$, as it relies on the hidden subspaces of dual pairing vector spaces. We now have to define *semi-functional* keys and ciphertexts, that will be used in the security proof. We recall that our CP-ABE secret key for attributes sets Υ is equal to $\text{acc}_{\mathcal{X}} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{Q} a_i s^i}$. During decryption we compute $e(\text{acc}_{\mathcal{X}}, \text{aux}_2)$ where $\text{aux}_2 = g_2^{-\alpha_1\alpha_2 \boldsymbol{d}_1^*}$ is given in the ciphertext, for $\alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p$. As aux_2 is the only part of the ciphertext to interact with the secret key, it is the only part that need semi-functional form. To define SF keys and aux_2, we need to double the dimension of the used DPVS: we now have $\mathbb{D} = (\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3, \boldsymbol{d}_4)$ and $\mathbb{D}^* = (\boldsymbol{d}_1^*, \boldsymbol{d}_2^*, \boldsymbol{d}_3^*, \boldsymbol{d}_4^*)$, where $\boldsymbol{d}_3, \boldsymbol{d}_4, \boldsymbol{d}_3^*, \boldsymbol{d}_4^*$ will be used for semi-functional space. Thus, trivially we can define for a secret key sk_Υ and ciphertext auxiliary information aux_2 their semi-functional forms as:

$$\text{sk}_\Upsilon^{(SF)} = \text{sk}_\Upsilon \cdot g_1^{\boldsymbol{d}_3 t_3} \quad \text{and} \quad \text{aux}_2^{(SF)} = \text{aux}_2 \cdot g_2^{\boldsymbol{d}_3^* z_3}$$

for $t_3, z_3 \leftarrow \mathbb{Z}_p$.

When using the DS2 assumption to change challenge ciphertext from normal form to semi-functional, we will use the element t_1, which is equal either to $g_2^{\tau_1 \boldsymbol{d}_1^*}$ or to $g_2^{\tau \boldsymbol{d}_1^* + \tau_2 \boldsymbol{d}_3^*}$, to build either aux_2 or $\text{aux}_2^{(SF)}$. However, the random τ_1 will have to appear in other parts of the ciphertext as $e(\text{acc}_{\mathcal{X}}, \text{aux}_2) = e(\text{acc}_{\mathcal{X}}, g_2^{\boldsymbol{d}_1^*})^{\tau_1}$ thus for membership verification we have to be able to reconstruct $\tau_1 \sum_{i=0}^{q} a_i s^i$. And as τ_1 is only given in exponent of the

assumption's challenge we do not know it and will not be able to use it for other parts of the ciphertext, especially because in the ciphertext there are elements of $\mathbb{G}_1$ and we only have τ_1 as exponent of an element of $\mathbb{G}_2$.

Thus, we have to change the way we define semi-functional keys and ciphertexts. To do so we have to also modify normal secret keys and ciphertexts. Let us now define normal and semi-functional keys and ciphertexts as follows:

$$\mathsf{sk}_\Upsilon = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i + z_2 \boldsymbol{d}_2} \quad , \quad \mathsf{aux}_2 = g_2^{r\boldsymbol{d}_1^* + t_2 \boldsymbol{d}_2^*}$$
$$\mathsf{sk}_\Upsilon^{(SF)} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i + z_2 \boldsymbol{d}_2 + z_4 \boldsymbol{d}_4} \quad , \quad \mathsf{aux}_2^{(SF)} = g_2^{r\boldsymbol{d}_1^* + t_2 \boldsymbol{d}_2^* + t_4 \boldsymbol{d}_4^*}$$

where $z_2, z_4, t_2, t_4 \leftarrow \mathbb{Z}_p$. We easily notice that during membership verification between a normal key and a normal ciphertext, we have an extra term $e(g_1, g_2)^{\psi z_2 t_2}$. To remove this extra term, we can add in the key $g_1^{-\boldsymbol{d}_2 t_2}$ and in the ciphertext $g_2^{\boldsymbol{d}_2^* z_2}$.
But as we add an element to normal keys and one to normal ciphertexts, we have to modify the semi-functional keys and ciphertext by adding them $g_1^{-\boldsymbol{d}_2 t_2 - \boldsymbol{d}_4 t_4}$ and $g_2^{\boldsymbol{d}_2^* z_2 + \boldsymbol{d}_4^* z_4}$ respectively. Notice that we keep the same randoms for coefficients of $\boldsymbol{d}_4$ and $\boldsymbol{d}_4^*$ in both parts of the semi-functional key and ciphertext (as the assumption's challenge gives us only one coefficient for $\boldsymbol{d}_4, \boldsymbol{d}_4^*$ and if we randomized it for the second part of SF keys and ciphertext, again we will not be able to remove the extra term). But doing so we obtain that a semi-functional key always decrypt a semi-functional ciphertext, which should not be possible.

To fix this issue, we can define normal keys and ciphertexts as follows:

$$\mathsf{sk}_\Upsilon = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i + (\boldsymbol{d}_1 - \boldsymbol{d}_2)} \quad , \quad \mathsf{aux}_2 = g_2^{r\boldsymbol{d}_1^* + (\boldsymbol{d}_1^* + \boldsymbol{d}_2^*)}.$$

With this definition, we obtain in the accumulator verification $e(g_1, g_2)^{\psi\gamma} \cdot e(g_1, g_2)^{-\psi\gamma}$, as we wanted. But we also obtain $e(g_1, g_2)^{\psi\gamma \sum_{i=0}^{q} a_i s^i}$, an extra term we cannot remove.

At this point, our idea is to increase the dimension of the used DPVS of the accumulator to 3 (and thus 6 for the ABE to have semi-functional spaces). Then, we define normal

and semi-functional keys and ciphertexts as:

$$\mathsf{sk}_\Upsilon = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i + (\boldsymbol{d}_2 - \boldsymbol{d}_3)} \quad , \quad \mathsf{aux}_2 = g_2^{r\boldsymbol{d}_1^* + (\boldsymbol{d}_2^* + \boldsymbol{d}_3^*)}$$
$$\mathsf{sk}_\Upsilon^{(SF)} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i + (\boldsymbol{d}_2 - \boldsymbol{d}_3) + z_5\boldsymbol{d}_5 + z_6\boldsymbol{d}_6} \quad , \quad \mathsf{aux}_2^{(SF)} = g_2^{r\boldsymbol{d}_1^* + (\boldsymbol{d}_2^* + \boldsymbol{d}_3^*) + t_5\boldsymbol{d}_5^* + t_6\boldsymbol{d}_6^*}$$

where $z_5, z_6, t_5, t_6 \leftarrow \mathbb{Z}_p$. Decryption of a normal ciphertext by a normal or SK key will work as no extra term will be in the result and decryption of a SF ciphertext by a normal key will also work. However, decryption of a SF ciphertext by a SK key will not work as it has an extra term: $e(g_1, g_2)^{\psi(t_5 z_5 + t_6 z_6)}$[6].

Though there is one problem when defining keys and ciphertexts like this. In the security proof, we use the challenge of DS2 assumption t_2, t_3 to build the challenge ciphertext. (t_2, t_3) are either equals to $(g_2^{\tau_1 \boldsymbol{d}_2^*}, g_2^{\tau_1 \boldsymbol{d}_3^*})$ or to $(g_2^{\tau_1 \boldsymbol{d}_2^* + \tau_2 \boldsymbol{d}_5^*}, g_2^{\tau_1 \boldsymbol{d}_3^* + \tau_2 \boldsymbol{d}_6^*})$. We set $\mathsf{aux}_2 = g_2^{-\alpha_1\alpha_2 \boldsymbol{d}_1^*} \cdot t_2 \cdot t_3$. In the both case, we have that $\boldsymbol{d}_2^*, \boldsymbol{d}_3^*$ are randomized by τ_1. Thus we need to define $\mathsf{aux}_2 = g_2^{r\boldsymbol{d}_1^* + z(\boldsymbol{d}_2^* + \boldsymbol{d}_3^*)}$ for $z \leftarrow \mathbb{Z}_p$. As the same goes when using the challenge of DS1 assumption to build the challenge key, we have to define $\mathsf{sk}_\Upsilon = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i + r(\boldsymbol{d}_2 - \boldsymbol{d}_3)}$ for $r \leftarrow \mathbb{Z}_p$. We carry these modifications in $\mathsf{sk}_\Upsilon^{(SF)}$ and $\mathsf{aux}_2^{(SF)}$.

But notice that with way of building the challenge ciphertext, when t_2, t_3 are equals to $g_2^{\tau_1 \boldsymbol{d}_2^* + \tau_2 \boldsymbol{d}_5^*}, g_2^{\tau_1 \boldsymbol{d}_3^* + \tau_2 \boldsymbol{d}_6^*}$ we have that $t_5 = t_6 = \tau_2$ (and the same goes for the challenge key where $z_5 = z_6 = \tau_2$). Thus, we do not obtain an SF ciphertext (or SF key). To solve this issue we actually randomized $\boldsymbol{d}_3$ in the keys and $\boldsymbol{d}_2^*$ in the ciphertext, with the same random γ. Therefore, we define normal and SF keys and ciphertext as follows:

$$\mathsf{sk}_\Upsilon = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i + (\boldsymbol{d}_2 - \gamma\boldsymbol{d}_3)} \quad , \quad \mathsf{aux}_2 = g_2^{r\boldsymbol{d}_1^* + (\gamma\boldsymbol{d}_2^* + \boldsymbol{d}_3^*)}$$
$$sk_\Upsilon^{(SF)} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i + (\boldsymbol{d}_2 - \gamma\boldsymbol{d}_3) + z_5\boldsymbol{d}_5 + z_6\boldsymbol{d}_6} \quad , \quad \mathsf{aux}_2^{(SF)} = g_2^{r\boldsymbol{d}_1^* + (\gamma\boldsymbol{d}_2^* + \boldsymbol{d}_3^*) + t_5\boldsymbol{d}_5^* + t_6\boldsymbol{d}_6^*}$$

This gives us our second intermediate accumulator, presented in Figure 6.17. We highlight in green the differences with our first intermediate accumulator scheme, presented in Figure 6.16. Notice that we do not include $g_2^{r(\boldsymbol{d}_2^* - \gamma\boldsymbol{d}_3^*)}$ in the publicly computed accumulator as we do not need a semi-functional form of it.

Finally, to conclude our security proof, we will do a change of bases from $(\mathbb{D}, \mathbb{D}^*)$ to $(\mathbb{F}, \mathbb{F}^*)$. By the way we define it, we obtain $\boldsymbol{f}_1 = \boldsymbol{d}_1 - \eta\boldsymbol{d}_5$ where $\eta \leftarrow \mathbb{Z}_p$. It means that each part of the ciphertext that uses $\boldsymbol{d}_1$ will have a semi-functional part in bases $\mathbb{F}$,

[6] This idea is inspired by the IBE of [51].

- Gen(λ, q): run a bilinear group generation algorithm to get $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$. Then choose randoms $s, \gamma \leftarrow \mathbb{Z}_p^*$, and run Dual$(\mathbb{Z}_p^3)$ to get $\mathbb{D} = (\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3), \mathbb{D}^* = (\boldsymbol{d}_1^*, \boldsymbol{d}_2^*, \boldsymbol{d}_3^*)$. Let $\psi \in \mathbb{Z}_p^*$ be the random such that $\boldsymbol{d}_1 \cdot \boldsymbol{d}_1^* = \boldsymbol{d}_2 \cdot \boldsymbol{d}_2^* = \boldsymbol{d}_3 \cdot \boldsymbol{d}_3^* = \psi$. Set $\mathsf{sk}_{\mathsf{acc}} = (s, \gamma, \mathbb{D}, \mathbb{D}^*)$, and

$$\mathsf{pk}_{\mathsf{acc}} = \left(\begin{array}{c} \Gamma, g_1^{\boldsymbol{d}_1}, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 s}, \cdots, g_1^{\boldsymbol{d}_2 s^q}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_1^* s}, \\ \cdots, g_2^{\boldsymbol{d}_1^* s^q}, g_2^{\boldsymbol{d}_2^* \gamma}, g_2^{\boldsymbol{d}_2^* s}, \cdots, g_2^{\boldsymbol{d}_2^* s^q}, g_2^{\boldsymbol{d}_3^*} \end{array} \right).$$

  Return $\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}$.

- Eval$(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$: compute the coefficients $\{a_i\}_{i=0,\cdots,q}$ of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (Z + x)$. Then pick $r \leftarrow \mathbb{Z}_p$ and compute $\mathsf{acc}_{\mathcal{X}} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i + r(\boldsymbol{d}_2 - \gamma \boldsymbol{d}_3)}$, and return $\mathsf{acc}_{\mathcal{X}}$.

- PublicEval$(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$: compute the coefficients $\{a_i\}_{i=0,\cdots,q}$ of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (Z + x)$. Then compute $\mathsf{accp}_{\mathcal{X}} = g_2^{\boldsymbol{d}_1^* \sum_{i=0}^{q} a_i s^i}$, and return $\mathsf{accp}_{\mathcal{X}}$.

- WitCreate$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}} \backslash \mathsf{accp}_{\mathcal{X}}, \mathcal{X}, \mathcal{I})$: let $\{b_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{X}\setminus\mathcal{I}}[Z] = \prod_{x \in \mathcal{X}\setminus\mathcal{I}} (x + Z)$. Compute $W_1 = g_1^{\boldsymbol{d}_1 b_0}$ and $W_2 = g_2^{\boldsymbol{d}_2^* \sum_{i=1}^{q} b_i s^i}$, and return $\mathsf{wit}_{\mathcal{I}} = (W_1, W_2)$.

- Verify$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{wit}_{\mathcal{I}}, \mathcal{I})$: let $\{c_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{I}}[Z] = \prod_{x \in \mathcal{I}} (x + Z)$ and return 1 if $e(\mathsf{acc}_{\mathcal{X}}, g_2^{\boldsymbol{d}_1^*} \cdot g_2^{\boldsymbol{d}_2^* \gamma} \cdot g_2^{\boldsymbol{d}_3^*}) = e(W_1, g_2^{\boldsymbol{d}_1^* \sum_{i=0}^{q} c_i s^i}) \cdot e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, W_2)$, 0 otherwise.

- PublicVerify$(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}}, \mathsf{wit}_{\mathcal{I}}, \mathcal{I})$: let $\{c_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{I}}[Z] = \prod_{x \in \mathcal{I}} (x + Z)$ and return 1 if $e(g_1^{\boldsymbol{d}_1}, \mathsf{accp}_{\mathcal{X}}) = e(W_1, g_2^{\boldsymbol{d}_1^* \sum_{i=0}^{q} c_i s^i}) \cdot e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, W_2)$, 0 otherwise.

Figure 6.17: The second intermediate accumulator scheme.

and our ciphertext will no longer be a correct SF ciphertext. Indeed, we defined (and we need for the other parts of the proof) a SF ciphertext as being a normal ciphertext with only element aux_2 having a semi-functional part. Therefore, we need to replace $\boldsymbol{d}_1$

by $\boldsymbol{d}_3$ in ciphertexts to avoid this issue. As in our CP-ABE the anticipation of the first element of the membership witness, aux_1, uses $\boldsymbol{d}_1$, we modify our accumulator so that W_1 has in exponent $\boldsymbol{d}_3$. Plus, as in the CP-ABE ciphertext the mask of the message is $e(g_1^{\boldsymbol{d}_1}, \text{accp}_{\mathcal{Y}})$, we have to change the way to publicly computed accumulators: we now use $\{\boldsymbol{b}_3^* s^i\}_{i=0}^q$ instead of $\{\boldsymbol{b}_1^* s^i\}_{i=0}^q$. We also change elements ele_1, ele_2 in our CP-ABE ciphertext: we replace $\boldsymbol{d}_1^*$ by $\boldsymbol{d}_3^*$ to keep correctness. That gives us our last accumulator scheme (the one that we will use in our CP-ABE scheme), presented in Figure 6.18. We highlight in purple the differences with the intermediate accumulator of Figure 6.17.

Note 6.2.2 *To prove security of our attribute-based encryption scheme we will use DS1 and DS2 with parameter $k = 3$ and $n = 2k = 6$, and so DPVS of dimension 6.*

6.2.3 Our CP-ABE Scheme From Dually Computable Accumulator

The resulting CP-ABE is fully given in Figure 6.19. As said above, it permits to manage access policies expressed as disjunctions of conjunctions without “NO” gates. For sake of clarity, we use the same color notations than in the accumulator of Figure 6.18 and we highlight in red the elements used for anticipation of the witnesses and the intersection of both sets.

Theorem 6.2.1 *Our ciphertext policy attribute-based encryption scheme is correct.*

Proof 6.2.1 *We have that*

$$
\begin{aligned}
& e(\textsf{aux}_1^{\delta\delta'}, ele_1 \cdot ele_2^{\xi}) \\
= \; & e((g_1^{\alpha_2 \boldsymbol{d}_3 (x_0+y_0)})^{\delta\delta'}, g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0 s + s^2)} \cdot (g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0+s)})^{\xi}) \\
= \; & e(g_1^{\alpha_2 \boldsymbol{d}_3 \delta\delta' (x_0+y_0)}, g_2^{\alpha \boldsymbol{d}_3^* (s^2 + s(z_0+\xi) + z_0\xi)}) \\
= \; & e(g_1, g_2)^{\psi\alpha_1\alpha_2 (s^2 + s(z_0+\xi) + z_0\xi) c_0 \delta'} \cdot e(g_1, g_2)^{\psi\alpha_1\alpha_2 (s^2 + s(z_0+\xi) + z_0\xi) t_0 \delta}
\end{aligned}
$$

and

$$
\begin{aligned}
& e(ele_3 \cdot ele_4^{\xi}, W_2^{\delta'} \cdot W_2'^{\delta}) \\
= \; & e(g_1^{\alpha_1\alpha_2 \boldsymbol{d}_2 (z_0 s + s^2)} \cdot (g_1^{\alpha_1\alpha_2 \boldsymbol{d}_2 (z_0+s)})^{\xi}, (g_2^{\boldsymbol{d}_2^* \sum_{i=1}^{Q} c_i s^i})^{\delta'} \cdot (g_2^{\boldsymbol{d}_2^* \sum_{i=1}^{Q} t_i s^i})^{\delta}) \\
= \; & e(g_1^{\alpha_1\alpha_2 \boldsymbol{d}_2 (s^2 + s(z_0+\xi) + z_0\xi)}, g_2^{\boldsymbol{d}_2^* \delta' \sum_{i=1}^{Q} c_i s^i + \boldsymbol{d}_2^* \delta \sum_{i=1}^{Q} t_i s^i}) \\
= \; & e(g_1, g_2)^{\psi\alpha_1\alpha_2 (s^2 + s(z_0+\xi) + z_0\xi) \delta' \sum_{i=1}^{Q} c_i s^i}
\end{aligned}
$$

- $\mathsf{Gen}(\lambda, q)$: run a bilinear group generation algorithm to get $\Gamma = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$. Then choose randoms $s, \gamma \leftarrow \mathbb{Z}_p^*$, and run $\mathsf{Dual}(\mathbb{Z}_p^3)$ to get $\mathbb{D} = (\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3), \mathbb{D}^* = (\boldsymbol{d}_1^*, \boldsymbol{d}_2^*, \boldsymbol{d}_3^*)$. Let $\psi \in \mathbb{Z}_p^*$ be the random such that $\boldsymbol{d}_1 \cdot \boldsymbol{d}_1^* = \boldsymbol{d}_2 \cdot \boldsymbol{d}_2^* = \boldsymbol{d}_3 \cdot \boldsymbol{d}_3^* = \psi$. Set $\mathsf{sk}_{\mathsf{acc}} = (s, \gamma, \mathbb{D}, \mathbb{D}^*)$, and

$$\mathsf{pk}_{\mathsf{acc}} = \left(\begin{array}{c} \Gamma, g_1^{\boldsymbol{d}_3}, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 s}, \cdots, g_1^{\boldsymbol{d}_2 s^q}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_1^* s}, \cdots, g_2^{\boldsymbol{d}_1^* s^q}, \\ g_2^{\boldsymbol{d}_2^* \gamma}, g_2^{\boldsymbol{d}_2^* s}, \cdots, g_2^{\boldsymbol{d}_2^* s^q}, g_2^{\boldsymbol{d}_3^*}, g_2^{\boldsymbol{d}_3^* s}, \cdots, g_2^{\boldsymbol{d}_3^* s^q} \end{array} \right),$$

  and return $\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}$.

- $\mathsf{Eval}(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$: compute the coefficients $\{a_i\}_{i=0,\cdots,q}$ of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (Z + x)$. Then pick $r \leftarrow \mathbb{Z}_p$ and compute $\mathsf{acc}_{\mathcal{X}} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{q} a_i s^i + r(\boldsymbol{d}_2 - \gamma \boldsymbol{d}_3)}$, and return $\mathsf{acc}_{\mathcal{X}}$.

- $\mathsf{PublicEval}(\mathsf{pk}_{\mathsf{acc}}, \mathcal{X})$: compute the coefficients $\{a_i\}_{i=0,\cdots,q}$ of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z] = \prod_{x \in \mathcal{X}} (Z + x)$. Then compute $\mathsf{accp}_{\mathcal{X}} = g_2^{\boldsymbol{d}_3^* \sum_{i=0}^{q} a_i s^i}$, and return $\mathsf{accp}_{\mathcal{X}}$.

- $\mathsf{WitCreate}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}} \backslash \mathsf{accp}_{\mathcal{X}}, \mathcal{X}, \mathcal{I})$: let $\{b_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{X}\setminus\mathcal{I}}[Z] = \prod_{x \in \mathcal{X}\setminus\mathcal{I}} (x + Z)$. Compute $W_1 = g_1^{\boldsymbol{d}_3 b_0}$ and $W_2 = g_2^{\boldsymbol{d}_2^* \sum_{i=1}^{q} b_i s^i}$, and return $\mathsf{wit}_{\mathcal{I}} = (W_1, W_2)$.

- $\mathsf{Verify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{wit}_{\mathcal{I}}, \mathcal{I})$: let $\{c_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{I}}[Z] = \prod_{x \in \mathcal{I}} (x + Z)$ and return 1 if $e(\mathsf{acc}_{\mathcal{X}}, g_2^{\boldsymbol{d}_1^*} \cdot g_2^{\boldsymbol{d}_2^* \gamma} \cdot g_2^{\boldsymbol{d}_3^*}) = e(W_1, g_2^{\boldsymbol{d}_3^* \sum_{i=0}^{q} c_i s^i}) \cdot e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, W_2)$, 0 otherwise.

- $\mathsf{PublicVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{accp}_{\mathcal{X}}, \mathsf{wit}_{\mathcal{I}}, \mathcal{I})$: let $\{c_i\}_{i=0,\cdots,q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{I}}[Z] = \prod_{x \in \mathcal{I}} (x + Z)$ and return 1 if $e(g_1^{\boldsymbol{d}_3}, \mathsf{accp}_{\mathcal{X}}) = e(W_1, g_2^{\boldsymbol{d}_3^* \sum_{i=0}^{q} c_i s^i}) \cdot e(g_1^{\boldsymbol{d}_2 \sum_{i=0}^{q} c_i s^i}, W_2)$, 0 otherwise.

Figure 6.18: The dually computable accumulator used in our CP-ABE scheme.

$$\cdot e(g_1, g_2)^{\psi \alpha_1 \alpha_2 (s^2 + s(z_0 + \xi) + z_0 \xi) \delta \sum_{i=1}^{Q} t_i s^i}.$$

- Setup$(\lambda, 1^q)$: generate bilinear group $\Gamma = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g_1, g_2)$, dual pairing vector spaces $(\mathbb{D}, \mathbb{D}^*) \leftarrow \text{Dual}(\mathbb{Z}_p^6)$ such that $\mathbb{D} = (\boldsymbol{d}_1, \cdots, \boldsymbol{d}_6)$, $\mathbb{D}^* = (\boldsymbol{d}_1^*, \cdots, \boldsymbol{d}_6^*)$ and $\boldsymbol{d}_i \cdot \boldsymbol{d}_i^* = \psi$, for $i = 1, \cdots, 6$ and $\psi \in \mathbb{Z}_p^*$. Also choose $\gamma, s, x_0, y_0, z_0 \leftarrow \mathbb{Z}_p$ and a hash function $\mathcal{H}$ that takes as input an attributes set and outputs an element of $\mathbb{Z}_p \setminus \{\gamma, s, x_0, y_0, z_0\}$. Set $Q = 2^q - 1$, $\mathsf{msk} = \left(\gamma, s, g_2^{\boldsymbol{d}_2}, \left\{g_1^{\boldsymbol{d}_1 s^i}\right\}_{i=0}^{Q}, \left\{g_1^{\boldsymbol{d}_3 s^i}\right\}_{i=1}^{Q}\right)$ and

$$\mathsf{pk} = \left(\begin{array}{c} \Gamma, g_1^{\boldsymbol{d}_3}, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 s}, \cdots, g_1^{\boldsymbol{d}_2 s^Q}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_1^* s}, \cdots, g_2^{\boldsymbol{d}_1^* s^Q}, g_2^{\boldsymbol{d}_2^* \gamma}, \\ g_2^{\boldsymbol{d}_2^* s}, \cdots, g_2^{\boldsymbol{d}_2^* s^Q}, g_2^{\boldsymbol{d}_3^*}, g_2^{\boldsymbol{d}_3^* s}, \cdots, g_2^{\boldsymbol{d}_3^* s^Q}, \mathcal{H}, x_0, y_0, z_0 \end{array} \right).$$

  Return msk, pk.
- KeyGen$(\mathsf{pk}, \mathsf{msk}, \Upsilon)$: let $k \in \mathbb{N}$ be the number of attributes in Υ. Compute $p_1, \cdots, p_{2^k-1}$ all the non-empty subsets of Υ and set $\mathcal{X} = \{\mathcal{H}(p_i)\}_{i=1}^{2^k-1} \cup \{x_0, z_0\}$. Compute $\{a_i\}_{i=0,\cdots,Q}$ the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z] = (x_0 + Z) \cdot (z_0 + Z) \cdot \prod_{i=1}^{2^k-1} (\mathcal{H}(p_i) + Z)$. Pick $r \leftarrow \mathbb{Z}_p$ and set $\mathsf{sk}_\Upsilon = \mathsf{acc}_{\mathcal{X}} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{Q} a_i s^i + r(\boldsymbol{d}_2 - \gamma \boldsymbol{d}_3)}$.
- Encrypt$(\mathsf{pk}, \Pi, \mathsf{m})$: let $\Pi = \pi_1 \vee \pi_2 \vee \cdots \vee \pi_l$ be the access policy, where $l \in \mathbb{N}$ is the number of clauses in the policy, and π_i for $i = 1, \cdots, l$ is a conjunction of attributes. Define $\mathcal{Y}_i$ for $i = 1, \cdots, l$ as the set of attributes associated to clause π_i and $\mathcal{Y} = \cup_{i=1}^{l} \mathcal{H}(\mathcal{Y}_i) \cup \{y_0, z_0\}$. Let $\{m_i\}_{i=0}^{Q}$ be the coefficients of polynomial $\mathsf{Ch}_{\mathcal{Y}}[Z]$. Choose $z, \alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p$ and do
  - *Mask computation:* define $\mathsf{accp}_{\mathcal{Y}} = g_2^{\boldsymbol{d}_3^* \sum_{i=0}^{Q} m_i s^i}$ and $\boldsymbol{H} = e(g_1^{\boldsymbol{d}_3}, \mathsf{accp}_{\mathcal{Y}})^{\alpha_1 \alpha_2}$.
  - *Anticipation of the witnesses and auxiliary information computation:* set $\mathsf{aux}_1 = g_1^{\alpha_2 \boldsymbol{d}_3 (x_0 + y_0)}$ and $\mathsf{aux}_2 = g_2^{-\boldsymbol{d}_1^* \alpha_1 \alpha_2 + z(\gamma \boldsymbol{d}_2^* + \boldsymbol{d}_3^*)}$.
  - *Anticipation of the intersection:* set $ele_1 = g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0 s + s^2)}$, $ele_2 = g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0 + s)}$, $ele_3 = g_1^{\alpha_1 \alpha_2 \boldsymbol{d}_2 (z_0 s + s^2)}$, and $ele_4 = g_1^{\alpha_1 \alpha_2 \boldsymbol{d}_2 (z_0 + s)}$.

  Set $\mathsf{ct}_\Pi = (ele_1, ele_2, ele_3, ele_4, \mathsf{aux}_1, \mathsf{aux}_2, \mathsf{m} \cdot \boldsymbol{H})$ and return ct_Π.
- Decrypt$(\mathsf{pk}, \mathsf{sk}_\Upsilon, \Upsilon, \mathsf{ct}_\Pi, \Pi)$: find p_{j^*} (for $j^* \in \{1, \cdots, 2^k - 1\}$) the non-empty subset of Υ that satisfies Π It means that there exist $j \in [1, \cdots, l]$ such that $\mathcal{Y}_j = p_{j^*}$ and $\mathcal{H}(\mathcal{Y}_j) = \mathcal{H}(p_{j^*}) = \xi$. Let $\{c_i\}_{i=0}^{Q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{X}}[Z]/((z_0 + Z)(\xi + Z))$. Let $\{t_i\}_{i=0}^{Q}$ be the coefficients of the polynomial $\mathsf{Ch}_{\mathcal{Y}}[Z]/((z_0 + Z)(\xi + Z))$. Find $\delta, \delta' \in \mathbb{Z}_p$ such that $c_0 = x_0 \delta$ and $t_0 = y_0 \delta'$. Set $W_2 = g_2^{\boldsymbol{d}_2^* \sum_{i=1}^{Q} c_i s^i}$, $W_2' = g_2^{\boldsymbol{d}_2^* \sum_{i=1}^{Q} t_i s^i}$ and compute

$$\frac{\mathsf{m} \cdot \boldsymbol{H}}{\left(e(\mathsf{aux}_1^{\delta \delta'}, ele_1 \cdot ele_2^{\xi}) \cdot e(ele_3 \cdot ele_4^{\xi}, W_2^{\delta'} \cdot W_2'^{\delta}) \cdot e(\mathsf{acc}_{\mathcal{X}}, \mathsf{aux}_2)^{\delta'}\right)^{\delta^{-1}}}$$

Figure 6.19: Our adaptively secure ciphertext policy attribute-based encryption scheme with constant size ciphertexts and secret keys.

Therefore

$$\begin{aligned} & e(\mathbf{aux}_1^{\delta\delta'}, ele_1 \cdot ele_2^{\xi}) \cdot e(ele_3 \cdot ele_4^{\xi}, W_2^{\delta'} \cdot W_2'^{\delta}) \\ = \quad & e(g_1, g_2)^{\psi\alpha_1\alpha_2(s^2+s(z_0+\xi)+z_0\xi)\delta' \sum_{i=0}^{Q} c_i s^i} \\ & \cdot e(g_1, g_2)^{\psi\alpha_1\alpha_2(s^2+s(z_0+\xi)+z_0\xi)\delta \sum_{i=0}^{Q} t_i s^i} \end{aligned}$$

If ξ belongs to $\mathcal{X}$ and ξ belongs to $\mathcal{Y}$, then

$$\begin{aligned} & e(\mathbf{aux}_1^{\delta\delta'}, ele_1 \cdot ele_2^{\xi}) \cdot e(ele_3 \cdot ele_4^{\xi}, W_2^{\delta'} \cdot W_2'^{\delta}) \\ = \quad & e(g_1, g_2)^{\psi\alpha_1\alpha_2\delta' \sum_{i=0}^{Q} a_i s^i} \cdot e(g_1, g_2)^{\psi\alpha_1\alpha_2\delta \sum_{i=0}^{Q} m_i s^i} \end{aligned}$$

The last pairing is equal to

$$\begin{aligned} & e(\mathbf{acc}_{\mathcal{X}}, \mathbf{aux}_2)^{\delta'} \\ = \quad & e(g_1^{\boldsymbol{d}_1 \sum_{i=0}^{Q} a_i s^i + r(\boldsymbol{d}_2 - \gamma \boldsymbol{d}_3)}, g_2^{-\boldsymbol{d}_1^* \alpha_1\alpha_2 + z(\gamma \boldsymbol{d}_2^* + \boldsymbol{d}_3^*)})^{\delta'} \\ = \quad & e(g_1, g_2)^{-\alpha_1\alpha_2\psi \sum_{i=0}^{Q} a_i s^i \delta'} \cdot e(g_1, g_2)^{rz\gamma\psi} \cdot e(g_1, g_2)^{-rz\gamma\psi} \\ = \quad & e(g_1, g_2)^{-\alpha_1\alpha_2\psi \sum_{i=0}^{Q} a_i s^i \delta'} \end{aligned}$$

so multiplying it with $e(\mathbf{aux}_1^{\delta\delta'}, ele_1 \cdot ele_2^{\xi}) \cdot e(ele_3 \cdot ele_4^{\xi}, W_2^{\delta'} \cdot W_2'^{\delta})$ gives $e(g_1, g_2)^{\psi\alpha_1\alpha_2\delta \sum_{i=0}^{Q} m_i s^i}$. As δ is publicly known, one can from that recover $\boldsymbol{H}$ and then m. Therefore, the scheme is correct. □

Theorem 6.2.2 *Our ciphertext attribute-based encryption scheme satisfies adaptive indistinguishability under SXDH problem.*

To prove the security of our scheme, we prove that the encryption of challenge message is indistinguishable from the encryption of a random message. Let $N_q \in \mathbb{N}$ be the number of secret keys that the adversary is allowed to query.[3] To prove security of our scheme, we use a sequence $N_q + 3$ of games (our proof is inspired of Chen *et al.* [51]'s IBE security proof) and Water's dual system encryption framework (see Section 3.5).

- Game$_{Real}$ is the original security game, as presented in Figure 6.15.

[3] As the number of attributes in the scheme is bounded, so is the number of keys that an adversary can query.

- Game_0 is the same as Game_{Real} except that the challenge ciphertext is a *semi-functional* ciphertext.
- Game_i for $i = 1, \cdots, N_q$ is the same as Game_0 except that the first i keys are semi-functional.
- Game_{Final} is the same as Game_{N_q} except that the challenge ciphertext is an encryption of a random message.

Now we define semi-functional keys and ciphertexts. Let $t_5, t_6, z_5, z_6 \leftarrow \mathbb{Z}_p$.

- a semi-functional key for Υ, $\mathsf{sk}_\Upsilon^{(SF)}$, is computed from normal key sk_Υ as $\mathsf{sk}_\Upsilon^{(SF)} = \mathsf{sk}_\Upsilon \cdot g_1^{t_5 \boldsymbol{d}_5^* + t_6 \boldsymbol{d}_6^*} = g_1^{\boldsymbol{d}_1^* \sum_{i=0}^{Q} a_i s^i + r(\boldsymbol{d}_2^* - \gamma \boldsymbol{d}_3^*) + t_5 \boldsymbol{d}_5^* + t_6 \boldsymbol{d}_6^*}$
- a semi-functional ciphertext for Π, $\mathsf{ct}_\Pi^{(SF)}$, is computed as a normal ciphertext ct_Π except that $\mathsf{aux}_2^{(SF)} = \mathsf{aux}_2 \cdot g_2^{z_5 \boldsymbol{d}_5 + z_6 \boldsymbol{d}_6}$.

Notice that normal keys can decrypt SF ciphertexts, and normal ciphertexts can be decrypted by SF keys. However, decryption of a SF ciphertext by a SF key leads to an additional term: $1/e(g_1, g_2)^{(t_5 z_5 \psi + t_6 z_6 \psi)\delta^{-1}}$.

Our proof is using assumptions DS1 and DS2 (Definition 2.3.4) that hold if SXDH holds. Informally, the proof is done as follows.

- First we prove that if there exists an adversary that can distinguish Game_{Real} from Game_0 then we can build an adversary that breaks the DS2 assumption with parameters $k = 3$ and $n = 6$. To do so the main idea is to use the assumption's challenge to build the challenge ciphertext. Depending on the value of the challenge we will either obtain a normal ciphertext or a semi-functional one.
- Then we prove that if there exists an adversary that can distinguish Game_{j-1} from Game_j for $j = 1, \cdots, N_q$ we can build an adversary that breaks the DS1 assumption with $k = 3$ and $n = 6$. The idea is to use the assumption's challenge to build the j-th key. Thus, depending on the value of the challenge we will either obtain a normal key or a semi-functional one. To build the challenge ciphertext, we use the assumption's parameters to obtain a semi-functional ciphertext.
- Finally, we prove that Game_{N_q} is computationally indistinguishable from Game_{Final}, with a change of dual orthonormal bases. Doing so, we randomized the coefficient of $\boldsymbol{d}_1^*$ in the aux_2 term of the ciphertext, thereby severing its link with the blinding factor. That gives us the encryption of a random message.

Lemma 6.2.1 *If there exists a PPT algorithm $\mathcal{A}$ such that $\mathsf{Adv}_{\mathcal{A}}^{Game_{Real}} - \mathsf{Adv}_{\mathcal{A}}^{Game_0}$ is non-negligible, then there exists a PPT algorithm $\mathcal{B}$ with non-negligible advantage against assumption DS2 with $k = 3$ and $n = 6$.*

Proof 6.2.2 *INIT: $\mathcal{B}$ is given $\Delta = (\Gamma, g_1^{\boldsymbol{b}_1}, g_1^{\boldsymbol{b}_2}, g_1^{\boldsymbol{b}_3}, g_2^{\boldsymbol{b}_1^*}, g_2^{\boldsymbol{b}_2^*}, g_2^{\boldsymbol{b}_3^*}, g_2^{\boldsymbol{b}_4^*}, g_2^{\boldsymbol{b}_5^*}, g_2^{\boldsymbol{b}_6^*}, u_1, u_2, u_3, \mu_2)$ along with t_1, t_2, t_3. $\mathcal{B}$ must decide if t_1, t_2, t_3 are distributed as $g_2^{\tau_1 \boldsymbol{b}_1^*}, g_2^{\tau_1 \boldsymbol{b}_2^*}, g_2^{\tau_1 \boldsymbol{b}_3^*}$ or $g_2^{\tau_1 \boldsymbol{b}_1^* + \tau_2 \boldsymbol{b}_4^*}, g_2^{\tau_1 \boldsymbol{b}_2^* + \tau_2 \boldsymbol{b}_5^*}, g_2^{\tau_1 \boldsymbol{b}_3^* + \tau_2 \boldsymbol{b}_6^*}$.*

SETUP: $\mathcal{B}$ first chooses a random invertible matrix $\boldsymbol{A} \in \mathbb{Z}_p^{3\times 3}$. It implicitly sets dual orthonormal bases $\mathbb{D}, \mathbb{D}^*$ to: $\boldsymbol{d}_1^* = \boldsymbol{b}_1^*, \boldsymbol{d}_2^* = \boldsymbol{b}_2^*, \boldsymbol{d}_3^* = \boldsymbol{b}_3^*$, $(\boldsymbol{d}_4^*, \boldsymbol{d}_5^*, \boldsymbol{d}_6^*) = (\boldsymbol{b}_4^*, \boldsymbol{b}_5^*, \boldsymbol{b}_6^*) \cdot \boldsymbol{A}$, $\boldsymbol{d}_1 = \boldsymbol{b}_1, \boldsymbol{d}_2 = \boldsymbol{b}_2, \boldsymbol{d}_3 = \boldsymbol{b}_3$, $(\boldsymbol{d}_4, \boldsymbol{d}_5, \boldsymbol{d}_6) = (\boldsymbol{b}_4, \boldsymbol{b}_5, \boldsymbol{b}_6) \cdot (\boldsymbol{A}^{-1})^\top$.

We note that $\mathbb{D}, \mathbb{D}^*$ are properly distributed and reveal no information about $\boldsymbol{A}$. Notice also that $\mathcal{B}$ cannot produce $g_1^{\boldsymbol{d}_4}, g_1^{\boldsymbol{d}_5}, g_1^{\boldsymbol{d}_6}$, but these will not be needed to create normal keys. $\mathcal{B}$ chooses random values $\gamma, s, x_0, y_0, z_0 \in \mathbb{Z}_p$ and a hash function $\mathcal{H}$ that takes as input attributes set and outputs an element of $\mathbb{Z}_p \setminus \{\gamma, s, x_0, y_0, z_0\}$. $\mathcal{A}$ is given the public key

$$\mathsf{pk} = \left(\begin{array}{c} \Gamma, g_1^{\boldsymbol{d}_3}, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 s}, \cdots, g_1^{\boldsymbol{d}_2 s^Q}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_1^* s}, \cdots, g_2^{\boldsymbol{d}_1^* s^Q}, g_2^{\boldsymbol{d}_2^* \gamma}, \\ g_2^{\boldsymbol{d}_2^* s}, \cdots, g_2^{\boldsymbol{d}_2^* s^Q}, g_2^{\boldsymbol{d}_3^*}, g_2^{\boldsymbol{d}_3^* s}, \cdots, g_2^{\boldsymbol{d}_3^* s^Q}, \mathcal{H}, x_0, y_0, z_0 \end{array} \right)$$

The master key is $\mathsf{msk} = (\gamma, s, g_2^{\boldsymbol{d}_2^*}, \left\{ g_1^{\boldsymbol{d}_1 s^i} \right\}_{i=0}^{Q}, \left\{ g_1^{\boldsymbol{d}_3 s^i} \right\}_{i=1}^{Q})$.

KEY QUERY: msk is known to $\mathcal{B}$, which allows $\mathcal{B}$ to respond to all of $\mathcal{A}$'s key queries by calling the normal key generation algorithm.

CHALLENGE: $\mathcal{A}$ sends to $\mathcal{B}$ a challenge policy Π^* and two challenge messages m_0, m_1. $\mathcal{B}$ chooses a random bit $b \in \{0, 1\}$ and encrypts m_b under Π^* as follows: $z, \alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p$ and

$$\begin{array}{ll} \mathsf{accp}_{\mathcal{Y}} = g_2^{\boldsymbol{b}_3^* \sum_{i=0}^{Q} m_i s^i} & \boldsymbol{H} = e(g_1^{\boldsymbol{b}_3}, \mathsf{accp}_{\mathcal{Y}})^{\alpha_1 \alpha_2} \\ \mathsf{aux}_1 = g_1^{\alpha_2 \boldsymbol{b}_3 (x_0 + y_0)} & \mathsf{aux}_2 = g_2^{-\boldsymbol{b}_1^* \alpha_1 \alpha_2} \cdot t_2^{\gamma} \cdot t_3 \\ ele_1 = g_2^{\alpha_1 \boldsymbol{b}_3^* (z_0 s + s^2)} & ele_2 = g_2^{\alpha_1 \boldsymbol{b}_3^* (z_0 + s)} \\ ele_3 = g_1^{\alpha_1 \alpha_2 \boldsymbol{b}_2 (z_0 s + s^2)} & ele_4 = g_1^{\alpha_1 \alpha_2 \boldsymbol{b}_2 (z_0 + s)} \end{array}$$

where $\mathcal{Y} = \{\mathcal{H}(\mathcal{Y}_i)\}_{i=1}^{l} \cup \{y_0, z_0\}$, and $\mathcal{Y}_i$ for $i = 1, \cdots, l$ is a set that contains the elements of the clause π_i^*. It gives the ciphertext $\mathsf{ct}^* = (ele_1, ele_2, ele_3, ele_4, \mathsf{aux}_1, \mathsf{aux}_2, m \cdot \boldsymbol{H})$ to $\mathcal{A}$.

- *If* $(t_1, t_2, t_3) = (g_2^{\tau_1 \boldsymbol{b}_1^*}, g_2^{\tau_1 \boldsymbol{b}_2^*}, g_2^{\tau_1 \boldsymbol{b}_3^*})$*, we have a normal ciphertext with randomness* $z = \tau_1$*.*

$$\begin{array}{ll} \mathit{accp}_{\mathcal{Y}} = g_2^{\boldsymbol{d}_3^* \sum_{i=0}^{Q} m_i s^i} & \boldsymbol{H} = e(g_1^{\boldsymbol{d}_3}, \mathit{accp}_{\mathcal{Y}})^{\alpha_1 \alpha_2} \\ \mathit{aux}_1 = g_1^{\alpha_2 \boldsymbol{d}_3 (x_0 + y_0)} & \mathit{aux}_2 = g_2^{-\boldsymbol{d}_1^* \alpha_1 \alpha_2 + \tau_1(\gamma \boldsymbol{d}_2^* + \boldsymbol{d}_3^*)} \cdot t_2^{\gamma} \cdot t_3 \\ ele_1 = g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0 s + s^2)} & ele_2 = g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0 + s)} \\ ele_3 = g_1^{\alpha_1 \alpha_2 \boldsymbol{d}_2 (z_0 s + s^2)} & ele_4 = g_1^{\alpha_1 \alpha_2 \boldsymbol{d}_2 (z_0 + s)} \end{array}$$

  Thus $\mathcal{B}$ *has properly simulated* Game_{Real}*.*

- *If* $(t_1, t_2, t_3) = (g_2^{\tau_1 \boldsymbol{b}_1^* + \tau_2 \boldsymbol{b}_4^*}, g_2^{\tau_1 \boldsymbol{b}_2^* + \tau_2 \boldsymbol{b}_5^*}, g_2^{\tau_1 \boldsymbol{b}_3^* + \tau_2 \boldsymbol{b}_6^*})$*, then we have that* aux_2 *is equal to* $g_2^{-\boldsymbol{d}_1^* \alpha_1 \alpha_2 + \tau_1(\gamma \boldsymbol{d}_2^* + \boldsymbol{d}_3^*) + \tau_2 \gamma \boldsymbol{b}_5^* + \tau_2 \boldsymbol{b}_6^*}$*.*

This ciphertext has an additional term with coefficients in bases $\boldsymbol{b}_5^*, \boldsymbol{b}_6^*$*, which form the vector* $\tau_2(\gamma, 1)$*. To compute coefficients in the bases* $(\boldsymbol{d}_5^*, \boldsymbol{d}_6^*)$ *we multiply the matrix* $\boldsymbol{A}^{-1}$ *by the transpose of this vector. Since* $\boldsymbol{A}$ *is random, these new coefficients are uniformly random. Thus, in this case, the ciphertext is SF (with coefficients in the base* $\mathbb{D}$*) and* $\mathcal{B}$ *has properly simulated* Game_0*. This allows* $\mathcal{B}$ *to leverage* $\mathcal{A}$*'s non-negligible difference in advantage between* Game_{Real} *and* Game_0 *to achieve a non-negligible advantage against DS2.* □

Lemma 6.2.2 *If there exists a PPT algorithm* $\mathcal{A}$ *such that* $\mathsf{Adv}_{\mathcal{A}}^{\mathit{Game}_{j-1}} - \mathsf{Adv}_{\mathcal{A}}^{\mathit{Game}_j}$ *(for* $j = 1, \cdots, N_q$*) is non-negligible, then there exists a PPT algorithm* $\mathcal{B}$ *with non-negligible advantage against assumption DS1 with* $k = 3$ *and* $n = 6$*.*

Proof 6.2.3 *INIT: $\mathcal{B}$ is given* $\Delta = (\Gamma, g_2^{\boldsymbol{b}_1^*}, g_2^{\boldsymbol{b}_2^*}, g_2^{\boldsymbol{b}_3^*}, g_1^{\boldsymbol{b}_1}, g_1^{\boldsymbol{b}_2}, g_1^{\boldsymbol{b}_3}, g_1^{\boldsymbol{b}_4}, g_1^{\boldsymbol{b}_5}, g_1^{\boldsymbol{b}_5}, u_1, u_2, u_3, \mu_2)$ *along with* t_1, t_2, t_3*, distributed either as* $g_1^{\tau_1 \boldsymbol{b}_1}, g_1^{\tau_1 \boldsymbol{b}_2}, g_1^{\tau_1 \boldsymbol{b}_3}$ *or* $g_1^{\tau_1 \boldsymbol{b}_1 + \tau_2 \boldsymbol{b}_3}, g_1^{\tau_1 \boldsymbol{b}_2 + \tau_2 \boldsymbol{b}_4}, g_1^{\tau_1 \boldsymbol{b}_3 + \tau_2 \boldsymbol{b}_6}$*.*

SETUP: $\mathcal{B}$ chooses a random invertible matrix $\boldsymbol{A} \in \mathbb{Z}_q^{3 \times 3}$*. Then it implicitly sets dual orthonormal bases* $\mathbb{D}, \mathbb{D}^*$ *to:* $\boldsymbol{d}_1^* = \boldsymbol{b}_1^*$*,* $\boldsymbol{d}_2^* = \boldsymbol{b}_2^*$*,* $\boldsymbol{d}_3^* = \boldsymbol{b}_3^*$ $(\boldsymbol{d}_4^*, \boldsymbol{d}_5^*, \boldsymbol{d}_6^*) = (\boldsymbol{b}_4^*, \boldsymbol{b}_5^*, \boldsymbol{b}_6^*) \cdot \boldsymbol{A}$*,* $\boldsymbol{d}_1 = \boldsymbol{b}_1^*$*,* $\boldsymbol{d}_2^* = \boldsymbol{b}_2^*$*,* $\boldsymbol{d}_3^* = \boldsymbol{b}_3^*$*,* $(\boldsymbol{d}_4, \boldsymbol{d}_5, \boldsymbol{d}_6) = (\boldsymbol{b}_4, \boldsymbol{b}_5, \boldsymbol{b}_6) \cdot (\boldsymbol{A}^{-1})^{\top}$*.*
We note that $\mathbb{D}, \mathbb{D}^*$ *are properly distributed and reveal no information about* $\boldsymbol{A}$*.* $\mathcal{B}$ *chooses random values* $\gamma, s, x_0, y_0, z_0 \in \mathbb{Z}_p$ *and a hash function* $\mathcal{H}$ *that takes as input attributes set and outputs an element of* $\mathbb{Z}_p \setminus \{\gamma, s, x_0, y_0, z_0\}$*.* $\mathcal{A}$ *is given the public key*

$$\mathit{pk} = \left(\begin{array}{c} \Gamma, g_1^{\boldsymbol{d}_3}, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 s}, \cdots, g_1^{\boldsymbol{d}_2 s^Q}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_1^* s}, \cdots, g_2^{\boldsymbol{d}_1^* s^Q}, g_2^{\boldsymbol{d}_2^* \gamma}, \\ g_2^{\boldsymbol{d}_2^* s}, \cdots, g_2^{\boldsymbol{d}_2^* s^Q}, g_2^{\boldsymbol{d}_3^*}, g_2^{\boldsymbol{d}_3^* s}, \cdots, g_2^{\boldsymbol{d}_3^* s^Q}, \mathcal{H}, x_0, y_0, z_0 \end{array} \right)$$

The master key is msk $= (\gamma, s, g_2^{\boldsymbol{d}_2^*}, \left\{g_1^{\boldsymbol{d}_1 s^i}\right\}_{i=0}^{Q}, \left\{g_1^{\boldsymbol{d}_3 s^i}\right\}_{i=1}^{Q})$.

KEY QUERY: $\mathcal{B}$ knows msk and $g_1^{\boldsymbol{d}_5}, g_1^{\boldsymbol{d}_6}$, thus can easily call the key generation algorithm or produce semi-functional keys. It allows $\mathcal{B}$ to answer to all $\mathcal{A}$'s key queries.

- *To answer the first j-1 key queries that $\mathcal{A}$ makes, $\mathcal{B}$ runs the semi-functional key generation algorithm to produce semi-functional keys.*
- *To answer to the j-th key query for Υ^j, $\mathcal{B}$ responds with:*

$$sk_{\Upsilon^j} = g_1^{\boldsymbol{b}_1 \sum_{i=0}^{Q} a_i s^i} \cdot t_2 \cdot t_3^{-\gamma}$$

where $\{a_i\}_{i=0}^{Q}$ are the coefficients of polynomial $Ch_{\mathcal{X}}[Z]$ and $\mathcal{X} = \left\{\mathcal{H}(p_i^j)\right\}_{i=1}^{2^k-1} \cup \{x_0, z_0\}$, $k \in \mathbb{N}$ is the size of Υ^j and $\left\{p_i^j\right\}_{i=1}^{2^k-1}$ are all the non-empty parties of Υ^j.

  - *If $t_1, t_2, t_3 = g_1^{\tau_1 \boldsymbol{b}_1}, g_1^{\tau_1 \boldsymbol{b}_2}, g_1^{\tau_1 \boldsymbol{b}_3}$, then sk_{Υ^j} is a normal key with randomness $r = \tau_1$: $sk_{\Upsilon^j} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{Q} a_i s^i + \tau_1(\boldsymbol{d}_2 - \gamma \boldsymbol{d}_3)}$. Thus $\mathcal{B}$ has properly simulated $Game_{j-1}$.*
  - *If $t_1, t_2, t_3 = g_1^{\tau_1 \boldsymbol{b}_1 + \tau_2 \boldsymbol{b}_3}, g_1^{\tau_1 \boldsymbol{b}_2 + \tau_2 \boldsymbol{b}_4}, g_1^{\tau_1 \boldsymbol{b}_3 + \tau_2 \boldsymbol{b}_6}$, then: $sk_{\Upsilon^j} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{Q} a_i s^i + \tau_1(\boldsymbol{d}_2 - \gamma \boldsymbol{d}_3) + \tau_2(\boldsymbol{b}_4 - \gamma \boldsymbol{b}_6)}$.*

- *For the remaining key queries, $\mathcal{B}$ runs the normal key generation algorithm.*

CHALLENGE: At some point, $\mathcal{A}$ sends to $\mathcal{B}$ two challenge messages m_0, m_1 and a challenge policy $\Pi^* = \pi_1^* \vee \cdots \vee \pi_l^*$. $\mathcal{B}$ chooses a random bit $b \in \{0, 1\}$ and encrypts m_b under Π^* as follows: $z, \alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p$ and

$$\begin{aligned}
accp_{\mathcal{Y}} &= g_2^{\boldsymbol{b}_3^* \sum_{i=0}^{Q} m_i s^i} & \boldsymbol{H} &= e(g_1^{\boldsymbol{b}_3}, accp_{\mathcal{Y}})^{\alpha_1 \alpha_2} \\
aux_1 &= g_1^{\alpha_2 \boldsymbol{b}_3 (x_0 + y_0)} & aux_2 &= g_2^{-\boldsymbol{b}_1^* \alpha_1 \alpha_2} \cdot u_2^{\gamma} \cdot u_3 \\
ele_1 &= g_2^{\alpha_1 \boldsymbol{b}_3^* (z_0 s + s^2)} & ele_2 &= g_2^{\alpha_1 \boldsymbol{b}_3^* (z_0 + s)} \\
ele_3 &= g_1^{\alpha_1 \alpha_2 \boldsymbol{b}_2 (z_0 s + s^2)} & ele_4 &= g_1^{\alpha_1 \alpha_2 \boldsymbol{b}_2 (z_0 + s)}
\end{aligned}$$

which is equal to

$$\begin{aligned}
accp_{\mathcal{Y}} &= g_2^{\boldsymbol{d}_3^* \sum_{i=0}^{Q} m_i s^i} & \boldsymbol{H} &= e(g_1^{\boldsymbol{d}_3}, accp_{\mathcal{Y}})^{\alpha_1 \alpha_2} \\
aux_1 &= g_1^{\alpha_2 \boldsymbol{d}_3 (x_0 + y_0)} & aux_2 &= g_2^{-\boldsymbol{d}_1^* \alpha_1 \alpha_2} \cdot u_2^{\gamma} \cdot u_3 \\
ele_1 &= g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0 s + s^2)} & ele_2 &= g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0 + s)} \\
ele_3 &= g_1^{\alpha_1 \alpha_2 \boldsymbol{d}_2 (z_0 s + s^2)} & ele_4 &= g_1^{\alpha_1 \alpha_2 \boldsymbol{d}_2 (z_0 + s)}
\end{aligned}$$

where $\mathcal{Y} = \{\mathcal{H}(\mathcal{Y}_i)\}_{i=1}^{l} \cup \{y_0, z_0\}$, and $\mathcal{Y}_i$ for $i = 1, \cdots, l$ is a set that contains the ele-

ments of the clause π_i^*.

Suppose that $\mathcal{B}$ decides not to be honest, and find the nature of the j-th key by herself. To do so, she creates a SF ciphertext for a policy Π such that Υ^j satisfies Π. She tries to decrypt it with sk_{Υ^j} to learn if sk_{Υ^j} is a normal or a SF key (a normal key will decrypt correctly while a SF key will with high probability fail to decrypt). Let's see that by construction even if sk_{Υ^j} is SF it will decrypt correctly.

Suppose that $t_1, t_2, t_3 = g_1^{\tau_1 \boldsymbol{b}_1 + \tau_2 \boldsymbol{b}_3}, g_1^{\tau_1 \boldsymbol{b}_2 + \tau_2 \boldsymbol{b}_4}, g_1^{\tau_1 \boldsymbol{b}_3 + \tau_2 \boldsymbol{b}_6}$. During decryption, $\mathcal{B}$ computes $e(sk_{\Upsilon^j}, aux_2)$ which is equal to

$$e(g_1^{\boldsymbol{b}_1 \sum_{i=0}^{Q} a_i s^i + \tau_1 \boldsymbol{b}_2 + \tau_2 \boldsymbol{d}_5 + \gamma(-\tau_1 \boldsymbol{b}_3 - \tau_2 \boldsymbol{d}_6)}, g_2^{-\boldsymbol{b}_1^* \alpha\alpha_2 + \gamma(\mu_1 \boldsymbol{b}_2^* + \mu_2 \boldsymbol{b}_5^*) + \mu_1 \boldsymbol{b}_3^* + \mu_2 \boldsymbol{b}_6^*})$$

This can be decomposed as

$$\begin{aligned} e(g_1^{\boldsymbol{b}_1 \sum_{i=0}^{Q} a_i s^i}, g_2^{-\boldsymbol{b}_1^* \alpha\alpha_2}) \cdot e(g_1^{\tau_1 \boldsymbol{b}_2}, g_2^{\gamma\mu_1 \boldsymbol{b}_2^*}) \cdot e(g_1^{\tau_2 \boldsymbol{b}_5}, g_2^{\gamma\mu_2 \boldsymbol{b}_5^*}) \\ \cdot e(g_1^{-\gamma\tau_1 \boldsymbol{b}_3}, g_2^{\mu_1 \boldsymbol{b}_3^*}) \cdot e(g_1^{-\gamma\tau_2 \boldsymbol{b}_6}, g_2^{\mu_2 \boldsymbol{b}_6^*}) \end{aligned}$$

thanks to dual pairing vector spaces properties.

As Π is satisfied by Υ^j, the first pairing will cancel itself with the rest of the verification equation. And by construction, the four others cancel with each other. Thus, it will decrypt, and $\mathcal{B}$ will have no information about the j-th key's nature.

Note 6.2.3 *Notice that in order to create an SF ciphertext, $\mathcal{B}$ must use elements u_2 and u_3 of the assumption, as she does not know $g_2^{\boldsymbol{d}_5^*}$ and $g_2^{\boldsymbol{d}_6^*}$.*

In the authorized case, Υ^j does not satisfy Π^*. Let us see that when $t_1, t_2, t_3 = g_1^{\tau_1 \boldsymbol{b}_1 + \tau_2 \boldsymbol{b}_3}$, $g_1^{\tau_1 \boldsymbol{b}_2 + \tau_2 \boldsymbol{b}_4}, g_1^{\tau_1 \boldsymbol{b}_3 + \tau_2 \boldsymbol{b}_6}$, the extra coefficients in bases $(\boldsymbol{b}_5^*, \boldsymbol{b}_6^*)$ of the ciphertext and the extra coefficients in bases $(\boldsymbol{b}_5, \boldsymbol{b}_6)$ of the key are distributed as random vectors in the spans of $(\boldsymbol{d}_5^*, \boldsymbol{d}_6^*)$ and $(\boldsymbol{d}_5, \boldsymbol{d}_6)$ respectively. To express them in bases $(\boldsymbol{d}_5^*, \boldsymbol{d}_6^*)$ and $(\boldsymbol{d}_5, \boldsymbol{d}_6)$ respectively, we multiply them by $\boldsymbol{A}^{-1}$ and $\boldsymbol{A}^\top$ respectively. Since the distribution of everything given to $\mathcal{A}$ except for the j-th key and the challenge ciphertext is independent of the random matrix $\boldsymbol{A}$ and Υ^j does not satisfy Π^*, we can conclude that these coefficients are uniformly random. Thus, $\mathcal{B}$ has properly simulated $Game_j$ in this case.

If $t_1, t_2, t_3 = g_1^{\tau_1 \boldsymbol{b}_1}, g_1^{\tau_1 \boldsymbol{b}_2}, g_1^{\tau_1 \boldsymbol{b}_3}$ then the coefficients of the semi-functional part of the ciphertext are uniformly random. Thus, $\mathcal{B}$ has properly simulated $Game_{j-1}$ in this case.

Therefore, $\mathcal{B}$ can leverage $\mathcal{A}$'s non-negligible difference in advantage between these games to obtain a non-negligible advantage against DS1. □

Lemma 6.2.3 *For any PPT adversary $\mathcal{A}$,* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Game}_{Final}} \leq \mathsf{Adv}_{\mathcal{A}}^{\mathsf{Game}_{N_q}}$.

We prove this lemma, by randomizing the coefficient of $\boldsymbol{d}_1^*$ in the aux_2 term of the ciphertext, thereby severing its link with the blinding factor.

Proof 6.2.4 *We pick $\eta \in \mathbb{Z}_p$ and define new dual orthonormal bases $\mathbb{F} = (\boldsymbol{f}_1, \cdots, \boldsymbol{f}_6)$ and $\mathbb{F}^* = (\boldsymbol{f}_1^*, \cdots, \boldsymbol{f}_6^*)$ as follows:*

$$\begin{array}{ccccccc} \boldsymbol{f}_1^* = \boldsymbol{d}_1^*, & \boldsymbol{f}_2^* = \boldsymbol{d}_2^*, & \boldsymbol{f}_3^* = \boldsymbol{d}_3^*, & \boldsymbol{f}_4^* = \boldsymbol{d}_4^*, & \boldsymbol{f}_5^* = \eta\boldsymbol{d}_1^* + \boldsymbol{d}_5^*, & \boldsymbol{f}_6^* = \boldsymbol{d}_6^* \\ \boldsymbol{f}_1 = \boldsymbol{d}_1 - \eta\boldsymbol{d}_5, & \boldsymbol{f}_2 = \boldsymbol{d}_2, & \boldsymbol{f}_3 = \boldsymbol{d}_3, & \boldsymbol{f}_4 = \boldsymbol{d}_4, & \boldsymbol{f}_5 = \boldsymbol{d}_5, & \boldsymbol{f}_6 = \boldsymbol{d}_6 \end{array}$$

It is easy to see that $\mathbb{F}$ and $\mathbb{F}^*$ are also dual orthonormal, and are distributed the same as $\mathbb{D}$ and $\mathbb{D}^*$.

Then, the public key, challenge ciphertext, and queried secret keys in Game_{N^q} are expressed over bases $\mathbb{D}$ and $\mathbb{D}^*$:

$$\mathsf{pk} = \left(\begin{array}{c} \Gamma, g_1^{\boldsymbol{d}_3}, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 s}, \cdots, g_1^{\boldsymbol{d}_2 s^Q}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_1^* s}, \cdots, g_2^{\boldsymbol{d}_1^* s^Q}, g_2^{\boldsymbol{d}_2^* \gamma}, \\ g_2^{\boldsymbol{d}_2^* s}, \cdots, g_2^{\boldsymbol{d}_2^* s^Q}, g_2^{\boldsymbol{d}_3^*}, g_2^{\boldsymbol{d}_3^* s}, \cdots, g_2^{\boldsymbol{d}_3^* s^Q}, \mathcal{H}, x_0, y_0, z_0 \end{array} \right)$$

$$\mathsf{ct}_{\Pi} = \left(\begin{array}{ll} \mathsf{accp}_{\mathcal{Y}} = g_2^{\boldsymbol{d}_3^* \sum_{i=0}^{Q} m_i s^i} & \boldsymbol{H} = e(g_1^{\boldsymbol{d}_3}, g_2^{\boldsymbol{d}_3^* \sum_{i=0}^{Q} m_i s^i})^{\alpha_1 \alpha_2} \\ \mathsf{aux}_1 = g_1^{\alpha_2 \boldsymbol{d}_3 (x_0 + y_0)} & \mathsf{aux}_2 = g_2^{-\boldsymbol{d}_1^* \alpha_1 \alpha_2 + z(\gamma \boldsymbol{d}_2^* + \boldsymbol{d}_3^*) + z_5 \boldsymbol{d}_5^* + z_6 \boldsymbol{d}_6^*} \\ ele_1 = g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0 s + s^2)} & ele_2 = g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0 + s)} \\ ele_3 = g_1^{\alpha_1 \alpha_2 \boldsymbol{d}_2 (z_0 s + s^2)} & ele_4 = g_1^{\alpha_1 \alpha_2 \boldsymbol{d}_2 (z_0 + s)} \end{array} \right)$$

$$\{\mathsf{sk}_{\Upsilon^j}\}_{j \in [N_q]} = \left\{ \mathsf{acc}_{\mathcal{X}} \cdot g_1^{t_5^j \boldsymbol{d}_5 + t_6^j \boldsymbol{d}_6} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{Q} a_i^j s^i + r^j (\boldsymbol{d}_2 - \gamma \boldsymbol{d}_3) + t_5^j \boldsymbol{d}_5 + t_6^j \boldsymbol{d}_6} \right\}_{j \in [N_q]}$$

Then we can express them over bases $\mathbb{F}, \mathbb{F}^*$ as:

$$\mathsf{pk} = \left(\begin{array}{c} \Gamma, g_1^{\boldsymbol{f}_3}, g_1^{\boldsymbol{f}_2}, g_1^{\boldsymbol{f}_2 s}, \cdots, g_1^{\boldsymbol{f}_2 s^Q}, g_2^{\boldsymbol{f}_1^*}, g_2^{\boldsymbol{f}_1^* s}, \cdots, g_2^{\boldsymbol{f}_1^* s^Q}, g_2^{\boldsymbol{f}_2^* \gamma}, \\ g_2^{\boldsymbol{f}_2^* s}, \cdots, g_2^{\boldsymbol{f}_2^* s^Q}, g_2^{\boldsymbol{f}_3^*}, g_2^{\boldsymbol{f}_3^* s}, \cdots, g_2^{\boldsymbol{f}_3^* s^Q}, \mathcal{H}, x_0, y_0, z_0 \end{array} \right)$$

$$ct_{\Pi} = \begin{pmatrix} accp_{\mathcal{Y}} = g_2^{\boldsymbol{f}_3^* \sum_{i=0}^{Q} m_i s^i} & \boldsymbol{H} = e(g_1^{\boldsymbol{f}_3}, g_2^{\boldsymbol{f}_3^* \sum_{i=0}^{Q} m_i s^i})^{\alpha_1\alpha_2} \\ aux_1 = g_1^{\alpha_2 \boldsymbol{f}_3(x_0+y_0)} & aux_2 = g_2^{-\boldsymbol{f}_1^*\alpha' + z(\gamma \boldsymbol{f}_2^* + \boldsymbol{f}_3^*) + z_5 \boldsymbol{f}_5^* + z_6 \boldsymbol{f}_6^*} \\ ele_1 = g_2^{\alpha_1 \boldsymbol{f}_3^*(z_0 s + s^2)} & ele_2 = g_2^{\alpha_1 \boldsymbol{f}_3^*(z_0+s)} \\ ele_3 = g_1^{\alpha_1\alpha_2 \boldsymbol{f}_2(z_0 s + s^2)} & ele_4 = g_1^{\alpha_1\alpha_2 \boldsymbol{f}_2(z_0+s)} \end{pmatrix}$$

$$\{sk_{\Upsilon^j}\}_{j\in[N_q]} = \left\{ acc_{\mathcal{X}} \cdot g_1^{t_5^{j'} \boldsymbol{f}_5 + t_6^j \boldsymbol{f}_6} = g_1^{\boldsymbol{f}_1 \sum_{i=0}^{Q} a_i^j s^i + r^j(\boldsymbol{d}_2 - \gamma \boldsymbol{d}_3) + t_5^{j'} \boldsymbol{f}_5 + t_6^j \boldsymbol{f}_6} \right\}_{j\in[N_q]}$$

where

$$\alpha' = \alpha_1\alpha_2 - z_5\eta$$
$$\left\{ t_5'^j = t_5^j + \eta \sum_{i=0}^{Q} a_i^j s^i \right\}_{j\in[N_q]},$$

which are all uniformly distributed.

In other words, the coefficient $\alpha_1\alpha_2$ of $\boldsymbol{d}_1^*$ in the aux_2 term of the challenge ciphertext is changed to random coefficient $\alpha' \in \mathbb{Z}_p$ of $\boldsymbol{f}_1^*$, thus the challenge ciphertext can be viewed as a semi-functional encryption of a random message in $\mathbb{G}_T$. Moreover, the coefficients $\left\{t_5^{j'}\right\}_{j\in[N_q]}$ of $\boldsymbol{f}_5$ in the $\left\{sk_{\Upsilon^j}^{(SF)}\right\}_{j\in[N_q]}$ are uniformly distributed since $\left\{t_5^j\right\}$ of $\boldsymbol{d}_5$ are all independent random values. Thus $(pk, ct_{\Pi}^{(SF)}, \left\{sk_{\Upsilon^j}^{(SF)}\right\}_{i\in[N_q]})$ expressed over bases $\mathbb{F}$ and $\mathbb{F}^*$ is properly distributed as $(pk, ct_{\Pi_R}^{(SF)}, \left\{sk_{\Upsilon^j}^{(SF)}\right\}_{i\in[N_q]})$ in $Game_{Final}$.
In the adversary's view, both $(\mathbb{D}, \mathbb{D}^*)$ and $(\mathbb{F}, \mathbb{F}^*)$ are consistent with the same public parameters. Therefore, the challenge ciphertext and queried secret keys above can be expressed as keys and ciphertext in two ways, in $Game_{N_q}$ over bases $(\mathbb{D}, \mathbb{D}^*)$ and in $Game_{Final}$ over bases $(\mathbb{F}, \mathbb{F}^*)$. Thus, $Game_Q$ and $Game_{Final}$ are statistically indistinguishable. □

Lemma 6.2.4 *For any adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{Game_{Final}}(\lambda) = 0$.*

Proof 6.2.5 *The value of β is independent of the adversary's view in $Game_{Final}$. Hence, $\mathsf{Adv}_{\mathcal{A}}^{Game_{Final}}(\lambda) = 0$.* □

Comparison. It is known that monotone boolean formulas can be put under DNF form, where the latter represents the minterm of the formula, *i.e.* a minimal set of variables which, if assigned the value 1, forces the formula to take the value 1 regardless of the values assigned to the remaining variables [61]. It is also known that the circuit

complexity class monotone NC^1 is captured by monotone boolean formulas of log-depth and fan-in two [87]. Therefore, our CP-ABE can directly deal with monotone NC^1 circuits. We present in Table 6.4 a comparison of (bounded) CP-ABE scheme for monotone NC^1 circuits, based on pairings[4]. All schemes in this table overpass the one-use restriction on attributes, which imposes that each attribute is only present once in the access policy. All schemes are single authority, and secure in the standard model.

Table 6.4: Comparison of CP-ABE schemes for monotone NC^1 circuits, based on pairings. Here q is the bound on the number of attributes in the scheme, and l is the number of rows in the access matrix when the policy is expressed with LSSS matrix.

Schemes	|pk|	|ct|	|sk|	Adaptive Security	Assumption	Group Order	Pairing
[136]	$O(q)$	$O(l)$	$O(q)$	$\times$	Non Static	Prime	Symmetric
[91]	$O(q)$	$O(l)$	$O(q)$	$\surd$	Static	Composite	Symmetric
[94]	$O(q)$	$O(l)$	$O(q)$	$\surd$	Non Static	Prime	Symmetric
[87]	$O(q)$	$O(q)$	$O(l)$	$\surd$	Static	Prime	Asymmetric
Our	$O(2^q)$	$O(1)$	$O(1)$	$\surd$	Static	Prime	Asymmetric

As we can see our scheme is the first one to obtain constant size for both ciphertexts and secret keys. However, this is done at the cost of the public key size, which become exponential. This drawback comes from the fact that for accumulating user's attributes set we are running the hash function $\mathcal{H}$ on each non-empty subset of this set. Doing so we obtain an easy way to check if an attributes set verifies an access policy: if it does, one of non-empty subsets of the set is equal to one clause of the access policy. We argue that the size of the public key is less important than the size of the other parameters, as it can easily be stored on-line. Additionally, while the sets (and access policies) representation might be scary at first glance, this is not an issue in practice as (i) it is not necessary to keep all elements in memory and (ii) for each decryption, only the useful part will have to be computed again. Finding another way to accumulate attributes sets and access policies in order to have efficient membership verification may lead to a more efficient CP-ABE, with shorter public key size. We leave it as an open problem. We also leave as an open problem the case of unbounded ABE schemes

[4]Some works are expressing their monotone boolean formula through Linear Secret Sharing Scheme (LSSS) matrix, see [93] for more details on this transformation.

[12, 50], and the case of non-monotonic access formulas [115, 116], even if we give some intuitions about it in the note below.

Note 6.2.4 *To improve our CP-ABE scheme so that it deals with "NO" gates, we might need to use* universal *accumulators. A universal accumulator scheme provides both membership and non-membership proofs. We might use non-membership proofs to deal with "NO" gates. The dually computable feature can easily be defined for universal accumulator schemes. However, we were not able to construct such schemes. Our accumulator of Figure 5.8 can be made universal, following [73]'s idea for non-membership proofs: the use of Bezout's coefficients. Using Extended Euclidean algorithm, compute polynomials* $q_1[Z], q_2[Z]$ *such that* $Ch_{\mathcal{X}}[Z]q_1[Z] + Ch_{\mathcal{I}}[Z]q_2[Z] = 1$ *(at the condition that* $\mathcal{I} \cap \mathcal{X} = \varnothing$ *otherwise the gcd of their associate polynomials is not equal to* 1*). Then, set* $W_1 = g_2^{\boldsymbol{d}_1 q_1(s)}$ *and* $W_2 = g_2^{\boldsymbol{d}_2 q_2(s)}$*. However, when universal, our accumulator is no longer* dually computable*: in the non-membership verification, we have* $e(acc_{\mathcal{X}}, W_1)$*. Therefore, as* $acc_{\mathcal{X}}$ *is replaced by* $accp_{\mathcal{X}}$ *which is composed of two elements of* $\mathbb{G}_2$*, the pairing with* W_1 *cannot work. To keep it working, we would have to modify the witness, and thus we would no longer satisfies correctness of duality. Plus, the modification requires the use of private elements.*

6.2.4 Our KP-ABE Scheme From Dually Computable Accumulator

In this section we present a key policy attribute-based encryption scheme, which is built as our ciphertext policy attribute-based encryption scheme of Section 6.2.3, and we compare it to existing schemes. Our KP-ABE is presented in Figure 6.20. We use the same color notations than in Figure 6.19.

Theorem 6.2.3 *Our scheme is correct and satisfies adaptive indistinguishability under SXDH.*

Correctness and security proofs of our KP-ABE can be done as for our CP-ABE.

In Table 6.5 we compare our KP-ABE with other KP-ABE schemes. All schemes are for single authority, secure in the standard model, bounded and in the pairing settings. As we notice for our CP-ABE, there exist schemes that are unbounded or deal with non-monotonic access policies. We leave as an open problem to modify our KP-ABE to achieve such properties.

- Setup$(\lambda, 1^q)$: generate bilinear group $\Gamma = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g_1, g_2)$, dual pairing vector spaces $(\mathbb{D}, \mathbb{D}^*) \leftarrow \text{Dual}(\mathbb{Z}_p^6)$ such that $\mathbb{D} = (\boldsymbol{d}_1, \cdots, \boldsymbol{d}_6)$, $\mathbb{D}^* = (\boldsymbol{d}_1^*, \cdots, \boldsymbol{d}_6^*)$ and $\boldsymbol{d}_i \cdot \boldsymbol{d}_i^* = \psi$, for $i = 1, \cdots, 6$ and $\psi \in \mathbb{Z}_p^*$. Also choose $\gamma, s, x_0, y_0, z_0 \leftarrow \mathbb{Z}_p$ and a hash function $\mathcal{H}$ that takes as input an attributes set and outputs an element of $\mathbb{Z}_p \setminus \{\gamma, s, x_0, y_0, z_0\}$. Set $Q = 2^q - 1$, $\text{msk} = \left(\gamma, s, g_2^{\boldsymbol{d}_2^*}, \left\{g_1^{\boldsymbol{d}_1 s^i}\right\}_{i=0}^{Q}, \left\{g_1^{\boldsymbol{d}_3 s^i}\right\}_{i=1}^{Q}\right)$ and

$$\text{pk} = \left(\begin{array}{c} \Gamma, g_1^{\boldsymbol{d}_3}, g_1^{\boldsymbol{d}_2}, g_1^{\boldsymbol{d}_2 s}, \cdots, g_1^{\boldsymbol{d}_2 s^Q}, g_2^{\boldsymbol{d}_1^*}, g_2^{\boldsymbol{d}_1^* s}, \cdots, g_2^{\boldsymbol{d}_1^* s^Q}, g_2^{\boldsymbol{d}_2^* \gamma}, \\ g_2^{\boldsymbol{d}_2^* s}, \cdots, g_2^{\boldsymbol{d}_2^* s^Q}, g_2^{\boldsymbol{d}_3^*}, g_2^{\boldsymbol{d}_3^* s}, \cdots, g_2^{\boldsymbol{d}_3^* s^Q}, \mathcal{H}, x_0, y_0, z_0 \end{array} \right).$$

  Return msk, pk.
- KeyGen(pk, msk, Π): let $\Pi = \pi_1 \vee \pi_2 \vee \cdots \vee \pi_l$ be the access policy, where $l \in \mathbb{N}$ is the number of clauses in the policy, and π_i for $i = 1, \cdots, l$ is a conjunction of attributes. Define $\mathcal{Y}_i$ for $i = 1, \cdots, l$ as the set of attributes associated to clause π_i and $\mathcal{Y} = \cup_{i=1}^{l} \mathcal{H}(\mathcal{Y}_i) \cup \{y_0, z_0\}$. Let $\{m_i\}_{i=0}^{Q}$ be the coefficients of polynomial $\text{Ch}_{\mathcal{Y}}[Z]$. Pick $r \leftarrow \mathbb{Z}_p$ and set $\text{sk}_\Pi = \text{acc}_{\mathcal{Y}} = g_1^{\boldsymbol{d}_1 \sum_{i=0}^{Q} m_i s^i + r(\boldsymbol{d}_2 - \gamma \boldsymbol{d}_3)}$
- Encrypt(pk, Υ, m): let $k \in \mathbb{N}$ be the number of attributes in Υ. Compute $p_1, \cdots, p_{2^k-1}$ all the non-empty parties of Υ and set $\mathcal{X} = \{\mathcal{H}(p_i)\}_{i=1}^{2^k-1} \cup \{x_0, z_0\}$. Compute $\{a_i\}_{i=0,\cdots,Q}$ the coefficients of the polynomial $\text{Ch}_{\mathcal{X}}[Z] = (x_0 + Z) \cdot (z_0 + Z) \cdot \prod_{i=1}^{2^k-1} (\mathcal{H}(p_i) + Z)$. Choose $z, \alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p$ and do
  - *Mask computation:* define $\text{accp}_{\mathcal{X}} = g_2^{\boldsymbol{d}_3^* \sum_{i=0}^{Q} a_i s^i}$ and $\boldsymbol{H} = e(g_1^{\boldsymbol{d}_3}, \text{accp}_{\mathcal{X}})^{\alpha_1 \alpha_2}$.
  - *Anticipation for the witnesses and auxiliary information computation:* set $\text{aux}_1 = g_1^{\alpha_2 \boldsymbol{d}_3 (x_0 + y_0)}$ and $\text{aux}_2 = g_2^{-\boldsymbol{d}_1^* \alpha_1 \alpha_2 + z(\gamma \boldsymbol{d}_2^* + \boldsymbol{d}_3^*)}$.
  - *Anticipation of the element computation:* set $ele_1 = g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0 s + s^2)}$, $ele_2 = g_2^{\alpha_1 \boldsymbol{d}_3^* (z_0 + s)}$, $ele_3 = g_1^{\alpha_1 \alpha_2 \boldsymbol{d}_2 (z_0 s + s^2)}$ and $ele_4 = g_1^{\alpha_1 \alpha_2 \boldsymbol{d}_2 (z_0 + s)}$

  Set $\text{ct}_\Upsilon = (ele_1, ele_2, ele_3, ele_4, \text{aux}_1, \text{aux}_2, \text{m} \cdot \boldsymbol{H})$ and return ct_Υ.
- Decrypt(pk, sk_Π, Π, ct_Υ, Υ): Find p_{j^*} (for $j^* \in \{1, \cdots, 2^k - 1\}$) such that Υ satisfies Π It means that there exist $j \in [1, \cdots, l]$ such that $p_{j^*} = \mathcal{Y}_j$ and $\mathcal{H}(p_{j^*}) = \mathcal{H}(\mathcal{Y}_j) = \zeta$. Let $\{c_i\}_{i=0}^{Q}$ be the coefficients of the polynomial $\text{Ch}_{\mathcal{X}}[Z]/((z_0 + Z)(\zeta + Z))$. Let $\{t_i\}_{i=0}^{Q}$ be the coefficients of the polynomial $\text{Ch}_{\mathcal{Y}}[Z]/((z_0 + Z)(\zeta + Z))$. Find $\delta, \delta' \in \mathbb{Z}_p$ such that $c_0 = x_0 \delta$ and $t_0 = y_0 \delta'$. Set $W_2 = g_2^{\boldsymbol{d}_2^* \sum_{i=1}^{Q} c_i s^i}$, $W_2' = g_2^{\boldsymbol{d}_2^* \sum_{i=1}^{Q} t_i s^i}$ and compute

$$\frac{\text{m} \cdot \boldsymbol{H}}{\left(e(\text{aux}_1^{\delta \delta'}, ele_1 \cdot ele_3^{\zeta}) \cdot e(ele_2 \cdot ele_4^{\zeta}, W_2^{\delta'} \cdot W_2'^{\delta}) \cdot e(\text{acc}_{\mathcal{X}}, \text{aux}_2)^{\delta'}\right)^{\delta^{-1}}}$$

Figure 6.20: Our adaptively secure key policy attribute-based encryption scheme with constant size ciphertexts and secret keys.

Table 6.5: Comparison of KP-ABE schemes for monotone NC^1 circuits, based on pairings. Here q is the bound on the number of attributes in the scheme, and l is the number of rows in the access matrix when the policy is expressed with LSSS matrix.

Schemes	|pk|	|ct|	|sk|	Adaptive Security	Assumption	One-Use	Group Order	Pairing
[78]	$O(q)$	$O(l)$	$O(q)$	$\times$	Static	Yes	Prime	Symmetric
[91]	$O(q)$	$O(q)$	$O(l)$	$\surd$	Static	No	Composite	Symmetric
[94]	$O(q)$	$O(q)$	$O(l)$	$\surd$	Non Static	No	Prime	Symmetric
[87]	$O(n)$	$O(n)$	$O(l)$	$\surd$	Static	No	Prime	Asymmetric
Our	$O(2^q)$	$O(1)$	$O(1)$	$\surd$	Static	No	Prime	Asymmetric

6.3 Use Case

In this section we present a specific use case requiring data sharing schemes. We here focus on connected objects and the data they produce. Connected objects have an important role in our daily life: from connected clocks to connected watches, they have different functions to play but all of them require security. First, these objects are configured to fit user's specific needs so it must avoid another to change (maliciously on purpose or by mistake) the settings. For example, no one would enjoy being woke up in the middle of the night because someone hacked their connected clock and changed the alarm. Furthermore, these objects deal with user personal data and even sensitive confidential information. Therefore the privacy level guaranteed by the devices must be really high. As an example, the data coming from a connected tensiometer or glucometer can reveal a lot of information about the individual attached to those objects. It also happens that resource owners want to share to others (called users in the sequel, and denoted U) some data collected or produced by their connected devices (called a resource in the sequel): we can imagine that one would like to share with his personal fitness trainer the number of footsteps collected by the pedometer of his smart watch, or that one neighbor wants to give to the all neighborhood access to his connected weather station. For this purpose, a very appealing setting is to store the resource coming from connected objects into a central server, since it permits a larger and more flexible sharing.

Here we hence consider such case, and introduce a *Central Server* (CS) that will centralize the data coming from a set of connected objects. But owners of these must

be able to have full control on this sharing capacity: granting access only to authorized users and being able to stop sharing whenever they want. One natural way to give and remove access to a connected device is to associate the access to a policy, which defines the conditions to satisfy in order to obtain access. Doing so, only users having the right attributes (which can be names, addresses, ages, ...) satisfying the defined access policy will access the data of the devices. For this purpose, we also consider an *Authorization Server* (AS), who will help to check whether the access policy is verified by the attributes of the user accessing the data store in CS.

In the following, we present an innovative way to use identity-based encryption with wildcard schemes to do access control in the above scenario, through a sharing platform called COPP ("Connected Objects Preserving Privacy"). We first present the context, the actors and the security requirements of the platform in Section 6.3.1. Then in Section 6.3.2 we present our generic solution using identity-based encryption with wildcard scheme and the required properties for the latter. Finally, in Section 4.3.3 we present a WIBE scheme satisfying all desired features. Our work present an innovative way to use identity-based encryption with wildcard schemes to do access control in the above scenario and could also lead to new constructions of cryptographic-based access control since this is the first time, as far as we know, that a WIBE is used for such kind of access control.

6.3.1 Presentation

From the above context, our purpose is to find a way to prevent any non authorized user to access a resource, but also to protect the privacy of both (i) the owner of a resource who wants to share it with some users, and (ii) the user who is requesting an access to a resource. For this to succeed, we need to manage two issues at the same time.

- Access policies can themselves leak personal information: imagine a healthcare organization that stores patient medical records electronically and implements an access policy that restricts access to medical records based on the patient's medical condition. For example, the policy that might allow only authorized healthcare professionals to access the medical records of patients with certain sensitive conditions, such as HIV. If an outsider learn this access policy, she could infer that patients likely have HIV, as this condition is explicitly mentioned in the policy as criteria for restricted access. It means that in terms of privacy, access policies should be protected to anyone but the owner of the resource that defines it, and the AS that will manage access to the resource according to the policy.

- The identity of the user can also leak information about the resource owner. Indeed, taking the same example as above and considering the condition having heart issue, if one learn that a cardiologist is trying to access a resource from one user, then it also learn that the resource owner has heart issues. Hence, user's attributes should also be protected to anyone (including AS) but the user himself. But to prevent a user to cheat on the attribute she has (to access a resource she would not normally has the right to), we need to introduce a new entity, namely the *Identity provider* (IdP), whose role is to certify the attributes of the requesting users.

Based on that, we now consider the following work-flow. At first, a resource owner can store his resources to CS and define an access policy that is given to AS. In parallel, any user can obtain from IdP a certification of his attributes. Eventually, a user can request AS to obtain access to a specific resource. If the attributes of the user verify the access policy attached to the requested resource, the user obtains a token that can be used with CS to obtain the resource.

Basic solutions. To manage the above, one basic solution could be for the user to send his attributes to AS, so that the latter can verify if they match the access policy. But this is not satisfactory as AS will accumulate too much information on users, as explained in the above second item. Another solution could be to add some trust between AS and IdP. In this case, AS could send the access policy to IdP who can check the validity of the user's attributes, hence validating this access to the resource. In this case, the user's attributes are no more available to AS, and the user does not obtain any information about the access policy. But this is still not satisfactory, as IdP is in this case too powerful. She can easily decide whether a user can or cannot access to a resource, independently on his attributes and the access policy. Additionally, we do not verify the above first security item for obvious reasons.

In this thesis, we propose a cryptographic solution verifying all the above security items: protection of the access to a resource from non-authorized users, protection of the access policy, and protection of user's attributes. In the sequel, we will focus on access policies expressed as disjunctions of conjunctions, i.e. boolean formulas composed of **OR** of **AND**. For simplicity, we also suppose that each resource is only associated to one access policy. Our protocol still works if several policies are associated to each object, but it becomes less easy to read and less efficient.

Related work. Based on this setting, there are multiple ways to treat our problem using different security tools.

- **Role-Based Access Control** [64, 127]: in such a system, an entity (in our case AS) authenticates a user and verifies if it has the right attributes according to a given access policy. If it has the advantage of preventing the user of knowing the access policy attached to a resource, the main issue of such technique in our use case is that such entity should know everything about the users' attributes (to verify their validity according to the access policy). Hence, it is not fully relevant.
- **Anonymous Credential Systems** [48]: this primitive permits a user to prove to third parties that she has some certified attributes, without revealing who she is among the set of users having the same attributes. Such system could be used to manage some access policies, the IdP being the entity certifying the attributes of users. But such solution does not provide attribute confidentiality (except by adding some complex zero-knowledge proofs of knowledge), nor access policy secrecy as the user needs to know it to properly choose his attributes.
- **Attribute-Based Signatures** [104](ABS): in such a scheme, a signing key is related to some attributes and the signature of a message is generated thanks to an access policy. It results that if the attributes embedded in the key that has been used to generate the signature verify the access policy used to sign the message, then the verification process will outputs "true". The signing key is generated thanks to a master secret key (e.g., managed by IdP) and a set of attributes. In our context, the user can hence generate a signature to obtain the authorization to access a resource, then protecting his attributes. But the concept of "policy-hiding" ABS does not exist yet, as far as we know, and seems to be hard to obtain.
- **Attribute-Based Encryption** [126](ABE): this primitive, which has already been detailed Section 6.2, is the equivalent of ABS but for encryption. In our context, such cryptographic tool can be used by playing the usual authentication system based on encryption: AS generates a random, encrypt it and if the user can send back the initially chosen random value, it means that she has correctly decrypted the received ciphertext [65]. To fit our use case, we need to modify such primitive by two means: (i) having a key generation which does not permit the issuer to obtain any information about the user attributes; (ii) having a ciphertext which does not reveal the access policy that has been used. In this case, we talk about "policy-hiding" ABE. As far as we know, it does not exist an ABE with both properties in the literature. The policy-hiding property has been extensively studied, and many papers can be found in the literature [138, 110, 17, 107]. Regarding the privacy-preserving key generation, we can cite [131, 5].

Our idea. Our idea is to create a new kind of ABE by using an identity-based encryption with wildcard (WIBE) scheme. As explained previously, and detailed below, from such basic cryptographic concept, we need to add three functionalities: (i) the way to treat attributes and an access policy with a WIBE; (ii) the way to hide the access policy during decryption; (iii) the way to obtain a decryption key in a blind manner. As for the ABE, such tool can quite naturally be used in our setting, as we will see below. Before that, we investigate the required properties for the identity-based encryption with wildcard scheme.

Actors, architecture and requirements. We consider a set of connected objects that belong to several different owners (a Data Owner is denoted DO). Each connected object regularly generates some data that we call *resources*. A resource is denoted R_i and the set of resources is $\mathcal{R}$. We also denote $k = |\mathcal{R}|$. A resource is considered as sensitive and must be protected to non authorized entities. But as it is most of the time done in our modern world, we assume that the resource owner cannot directly manage the storage and the access to his resources. Hence, the storage of such sensitive data is delegated to a so called Central Server CS. But for the confidentiality of the resources, the latter only provides access to a resource if the requestor (a user U) provides a valid token. Such token is generated by an Authorization Server (AS), which is responsible for the management of the access to the resources. More precisely, we consider that the access to a resource R_i is conditioned by an access policy π_i. The latter is defined for each resource by the data owner DO of the said resource.

An access policy is most of the time defined over a set of attributes, and this is the case in our setting. It follows that each user U should be associated to a set of attributes $\mathscr{A}_{\mathsf{U}}$ (names, addresses, emails...), related to the access policies. As we do not want each user to freely manage his own attributes, as she can try to cheat to obtain more resources, those are managed by an identity provider (IdP), which is considered as an independent entity. Then the COPP platform allows a user U to access a resource R_i if and only if his attributes satisfy the access policy π_i protecting R_i. This verification that the access policy is verified is mandatory and permits the user to obtain a valid token, thanks to AS, that is eventually sent to CS to obtain the resource. Notice that this does not mean that AS could know whether a user U has or has not access to a requested resource. In our solution, this will not be the case. As a user is considered as an individual in our study, his attributes should be protected against non-authorized entities. This should include AS and CS.

Finally, we also consider that the access policy is something sensitive. As explain previously, some policies could reveal information about the data owner (“access is

provided to users having the attribute cardiologist" means that this data owner certainly has heart problems) and the user ("access is provided to users having the attribute cardiologist" means that this user is a cardiologist, "access is provided to Mr. Smith" means that the name of the user is Smith). We give more details about this requirement just below. From this general overview, we obtain Table 6.6 which summarizes what information is known by whom. We consider that the introduction of all those actors is the best way to protect the privacy of all individuals, namely data owners and users.

Table 6.6: Summary of the knowledge of each actor.

Actor	Resource	Access policy	User attributes	User has access	Requested resource
DO	x	x			
CS	x			x	x
AS		x			x
U	x		x	x	x
IdP			x		

Access policy secrecy. As shown above, it is important to provide the secrecy of the access policy, *w.r.t.* the Central Server and the Identity Provider. But what about the user? Does she need to obtain the information about the access policy? Through the following two examples we show that users and even authorized users should not learn the whole access policy.

Example 1. Imagine a financial institution that grants employees varying levels of access to customer financial data based on their job roles. For example, the access policy can states that only employees with the "higher" roles can access the accounts of the most valuable customers, and all other employees are restricted from doing so. A (malicious) employee with "low" role will not be able to access the resource (as wanted), but she might infer that the customers with this level of protection are likely valuable individuals. This could lead to targeted unauthorized access or potential data breaches targeting these customers. In this scenario, the access policy inadvertently discloses sensitive financial information by implying the wealth status of certain customers, which could have significant consequences for their privacy and security.

Example 2. Imagine a company that holds a portfolio of valuable patents and trade secrets. To protect their intellectual property, they restricts access to specific projects and proprietary information with the use of an access policy: only individuals directly involved in a project have access to its related documents. There are three reasons why even authorized users must not know the access policy:

- By not disclosing the specificities of the access policy to authorized users, the company minimizes the risk of attackers gaining insight into the company's intellectual property protection measures.
- Even trusted employees may inadvertently leak information or fall victim to social engineering attacks. If employees were fully aware of the access policy and its intricacies, it could make it easier for them to exploit or bypass the policy, potentially leading to data breaches.
- By keeping authorized users unaware of the access policy, the company helps maintain the integrity of their intellectual property and ensures that access is granted only to those who genuinely require it.

In this scenario, the company prioritizes the protection of their intellectual property by limiting knowledge of the access policy among authorized users. This approach helps mitigate risks associated with both external threats and unintentional internal breaches, safeguarding their valuable assets.

Note 6.3.1 *To access a resource, a user must grant access to some of his personal attributes. By hiding the access policy from the user, the user remains unaware of which attributes she is consenting to provide access to. Following the GDPR, we need to find a suitable lawful basis to permit such treatment of such personal attributes. If we consider the "consent" lawful basis, it seems hard to prevent the user from knowing the access policy. Indeed, in such case, the service provider must precisely informed an individual about the way his personal data are used. This seems to mean that the user should know which of his attributes are used, and how. If we can certainly inform the user about which of his attributes are used, it is infeasible to hide the access policy and explain the user how his attributes are used! Another option could be to use the "contract" lawful basis, which states that the processing is necessary for a contract the service provider has with the individual. We could easily consider that the user has signed a contract to access the resources, hence permitting some specified actors to use his personal data. Again, we can provide the user which of his attributes are used. As our purpose in this thesis is not to discuss about lawful issues, we defer such discussions to legal experts and jurists.*

Procedures. Based on the above, we consider the four following steps for our system:

1. a Setup phase which is executed by the AS and the IdP, which permits them to generate all the needed parameters and keys. In particular, each of them obtains a private key, and a global public key is also output;
2. a StoreResource step, in which DO stores a new resource R_i on CS and send the related access policy π_i to AS;

3. a KeyQuery protocol that is executed between a user (with a set of attributes), the IdP and the AS. The IdP can identify the user and verify his attributes. At the end of the protocol, the user obtains a secret key which is related to his attributes;
4. a ResourceQuery protocol between a user (having played the previous step), AS and CS, which permits the former to get access to some resources. AS permits a user with a set of attributes verifying the access policy to get of token that is eventually given to CS to access the resource.

A summary of the actors and architecture is given in Figure 6.21.

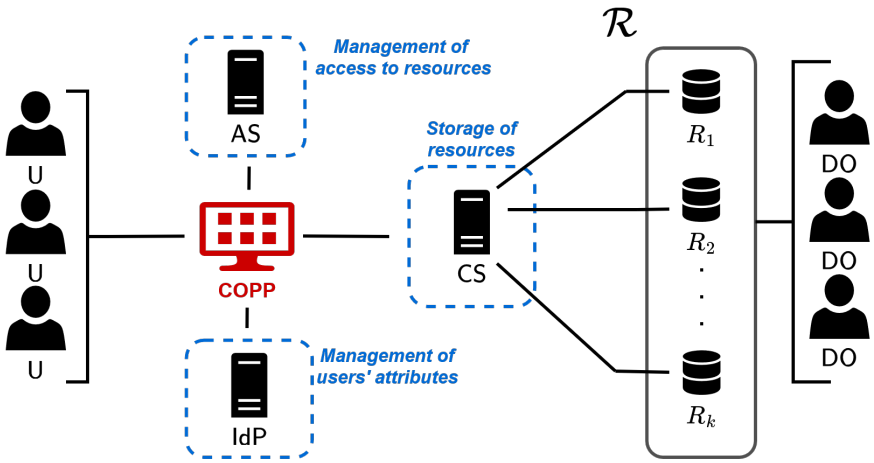


Figure 6.21: Actors and Architecture.

Security requirements. As a summary regarding security, we require the following requirements.

- For a given resource R_i, a set $\{\mathsf{U}_j\}_j$ of users, each U_j having a set of attributes $\mathscr{A}_{\mathsf{U}_j}$ that does not verify the access policy π_i defined by AS for R_i, must not access R_i.
- For a given resource, the access policy must only be known by AS and DO.
- For a given user U_j, the set of attributes $\mathscr{A}_{\mathsf{U}_j}$ verifying the access policy π_i attached to a resource R_i should not be revealed to AS nor CS.

We additionally assume the following, additionally to the fact that we consider no coalitions between the different actors:

- DO is honest;
- AS is honest but curious, meaning that it will correctly define the access policy attached to a resource, but can try to obtain some information about users' attributes;

- CS is honest but curious, trying to obtain information about the user's attributes and the access policy;
- IdP is honest but curious, and can want to obtain some information about the access policy, and the resource;
- users U are dishonest and can try to cheat to obtain access to more resources, and to obtain more information about access policies;

Requested features for WIBE. Before entering into the formal presentation of our solution with identity-based encryption with wildcard scheme, we first sketch how we can add the three above functionalities to a WIBE scheme, informally speaking.

Policies, attributes and patterns. We here consider that an access policy π is represented as a disjunction of conjunctions. For simplicity of the reading, we consider policies with only one clause even if our scheme will be working with policies considering several clauses (but be less efficient). Notice that a clause can be seen as a conjunction, thus it can be transformed into a pattern $\mathbf{P}'$ of space $\{0,1\}^L$, where for $l = 1, \cdots, L$, $P'_l = 1$ if attribute l is in the clause, and $P'_l = 0$ otherwise. It follows that we can associate to each user a pattern $\mathbf{P}$ of space $\{0,\star\}^L$, where for $l = 1, \cdots, L$, $P_l = \star$ if user has attribute l, $P_l = 0$ otherwise. Thus, we have that the key decrypts the ciphertext only if for $l = 1, \cdots, L$, $P_l = P'_l$ or $P_l = \star$, which will be denoted by $\mathbf{P}' \in_\star \mathbf{P}$. Based on that, it is easy to see that a "basic" WIBE is enough to manage such feature. What we need is to add two different procedures:

- Att2Pattern which takes as input a set $\mathscr{A}_\mathsf{U}$ of attributes and which outputs a pattern $\mathbf{P}_\mathsf{U}$ using the above transformation;
- AP2Pattern which takes as input an access policy π and which outputs a pattern $\mathbf{P}'$ using the above transformation;

Hiding the access policy. The next step is to hide the access policy during the decryption phase. As we now manage an access policy as described just above, what we need is a WIBE scheme in which the pattern is hidden with the sole knowledge of the ciphertext. Such concept already exists and known as an *anonymous* WIBE (see Definition 4.1.5).

Note 6.3.2 *Here anonymous security is enough for the WIBE, there is no need to consider our stronger security property of pattern-hiding.*

Attribute-protecting key generation. To obtain such feature, we can use a privacy-preserving key generation WIBE scheme, as presented in Chapter 4 (Section 4.2 and Definition 4.2.1). Doing so, we will have a scheme in which the user does not have to send its $\mathbf{P} = (P_1, \cdots, P_L)$ in the clear to obtain the associated decryption keys.

6.3.2 Our Generic Solution

In this section, we show how a privacy-preserving key generation identity-based encryption with wildcard scheme (Definition 4.2.2) with policy hiding can be used within the COPP platform. Our protocol is also using signature schemes (Definition 3.4.1).

Basic idea. Based on the architecture given in Section 6.3.1, we give a possible instanciation based on WIBE. For simplicity, we present the protocol with only one user U with attributes set $\mathscr{A}$ and one resource R associated to policy π, but the protocol easily works when there are several users and resources. The basic idea is as follows:

- during the Setup phase, the AS, playing the role of the KGC, generates the WIBE master secret key. The IdP generates a key pair for a digital signature. The IdP will also play the role of the PAC for the WIBE scheme. This is given by operations circled in blue in Figure 6.22;
- the StoreResource phase simply consists for the Data Owner DO to send, through two different secure channels (using e.g., TLS), (i) a new resource R to CS and (ii) the accompanied access policy π to AS. We consider that a unique identifier permits to make the link between the two. This step is represented by red circles in Figure 6.22. Compare to the others, this phase is no more detailed in the sequel, as we do not really have anything to add;
- the KeyQuery protocol corresponds to the privacy-preserving key generation phase in which the user obtains a secret key based on her attributes. The IdP verifies the attributes (i.e., the pattern in the WIBE sense), and the AS generates the user's secret key, in a blind manner, as described by operations circled in green in Figure 6.22;
- in the ResourceQuery phase, when receiving a query to access a resource R by a user U, the CS chooses a random challenge t and sends it to U, who forwards it to AS. The latter signs it and encrypts both t and the signature σ based on the underlying access policy π. It finally sends the ciphertext to the user. Based on the secret key obtains in the KeyQuery step, the user can decrypt it to retrieve t and σ iff her attributes verifies the access policy (otherwise, the resulting value has no sense w.r.t. t and the signature and CS will reject the query). This is represented by operations circled in purple in Figure 6.22.

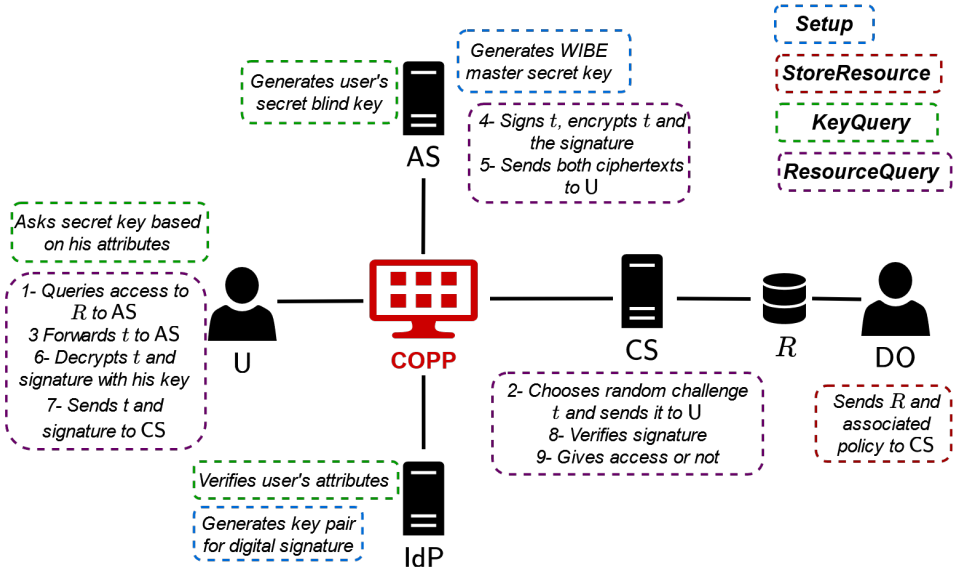


Figure 6.22: Our basic protocol.

Details. Let $\Pi = (\text{Setup, UserTemKeyGen, BlindTokenGen, BlindKeyGen, KeyExtract, Encrypt, Decrypt})$ be a WIBE scheme as given in Definition 4.2.1. We also need a signature scheme $\Sigma = (\text{KeyGen}, \text{Sign}, \text{Verify})$ which is existentially unforgeable against chosen message attacks (see Definition 3.4.3). Again, we present the protocol with only one user U with attributes set $\mathscr{A}$ and one resource R associated to policy π.

- Setup:
  1. AS runs the WIBE Π.Setup to obtain pk and msk. It also runs the Σ.Setup algorithm to obtain a signature key pair $(\text{sk}_{\text{AS}}, \text{vk}_{\text{AS}})$.
  2. IdP executes the signature Σ.Setup algorithm to obtain its key pair (sk, vk).

  Refer to operations circled in blue in Figure 6.23.
- KeyQuery: this step is executed by each user U having a set $\mathscr{A}$ of attributes, in collaboration with IdP and AS.
  1. U makes a request to IdP, by authenticating himself using the COPP interface, and by executing the Π.UserTemKeyGen algorithm, on input pk, to obtain the temporary key pair $(\text{tpk}_{user}, \text{tsk}_{user})$. The public part is sent to IdP.
  2. From the authentication, IdP retrieves the set $\mathscr{A}$ of attributes of this user and executes the Att2Pattern procedure on input $\mathscr{A}$ to get the corresponding pattern $\mathbf{P}$. It then answers with a blind token $\text{bt}_{\mathbf{P}}$ by executing Π.BlindTokenGen on input the pattern $\mathbf{P}$ and the user temporary public key (tpk_{user}). It eventually signs the resulting blind token $\text{bt}_{\mathbf{P}}$ using Σ.Sign on input its private key sk. It sends $\text{bt}_{\mathbf{P}}$ and the resulting signature σ to the user.
  3. U sends to AS (via COPP) the blind token $\text{bt}_{\mathbf{P}}$ and the IdP signature σ.

4. AS checks the validity of the signature σ, using Σ.Verify and the public key $\mathsf{pk}_{\mathsf{IdP}}$. If it is correct, it runs the blind key generation algorithm Π.BlindKeyGen on input the master secret key msk and the blind token $\mathsf{bt}_{\mathbf{P}}$, which outputs the blind secret key $\mathsf{bsk}_{\mathbf{P}}$. The latter is finally sent to user U.
5. U uses COPP to extracts her final secret key $\mathsf{sk}_{\mathscr{A}}$ by executing Π.KeyExtract on input the blind secret key $\mathsf{bsk}_{\mathbf{P}}$ and her temporary secret key tsk_{user}.

Refer to operations circled in green in Figure 6.23.

- ResourceQuery: we consider that U has previously played the KeyQuery protocol. She is thus in possession of a secret key $\mathsf{sk}_{\mathscr{A}}$, based on her attributes $\mathscr{A}$.
  1. U makes through COPP a request to CS for resource R, identified by Id_R.
  2. CS generates a random challenge t, sends it back to U who forwards it to AS
  3. AS first runs AP2Pattern routine to transform π into a pattern $\mathbf{P}^*$. It then signs $Id_R \| t$ using Σ.Sign and its signing key $\mathsf{sk}_{\mathsf{AS}}$, obtaining σ_t. The value $\tau = (Id_R, t, \sigma_t)$ corresponds to the access token. It then runs the WIBE encryption Π.Encrypt on input the pattern $\mathbf{P}^*$ and the message τ to get the ciphertext ct. Eventually, AS sends ct to user U.
  4. U runs the WIBE algorithm Π.Decrypt with U's secret key $\mathsf{sk}_{\mathscr{A}}$ and gets a value $\tilde{\tau}$, which is sent to the Central Server CS resource server for resource R.
  5. If U's attributes allow her to access the resource, then CS retrieves $((Id_R, t, \sigma_t)$ from $\tilde{\tau})$. If σ_t is a valid signature for $\mathsf{pk}_{\mathsf{AS}}$ on the message $Id_R \| t$, if t is similar to the one it has sent during step 1., and if Id_R exists, then the corresponding resource R is sent to U.

  Refer to operations circled in purple in Figure 6.23.

We now show that our above proposal verifies three properties.

Secrecy of the resource. If a non authorized user can access a resource for which she does not have the right attributes, it means that (i) she has corrupted IdP to obtain a secret key for attributes she does not have (but we have assume that such coalition is not possible), (ii) she has broken the indistinguishability of the WIBE scheme by being able to decrypt a message while not having a valid secret key (which is assumed to be infeasible), or (iii) she has broken the signature scheme to forge an AS signature to generate a token that will be accepted by CS (which is also infeasible).

Secrecy of the access policy. Such security property is given by the way the access policy is treated (see Table 6.6), and from the anonymity property of the WIBE scheme.

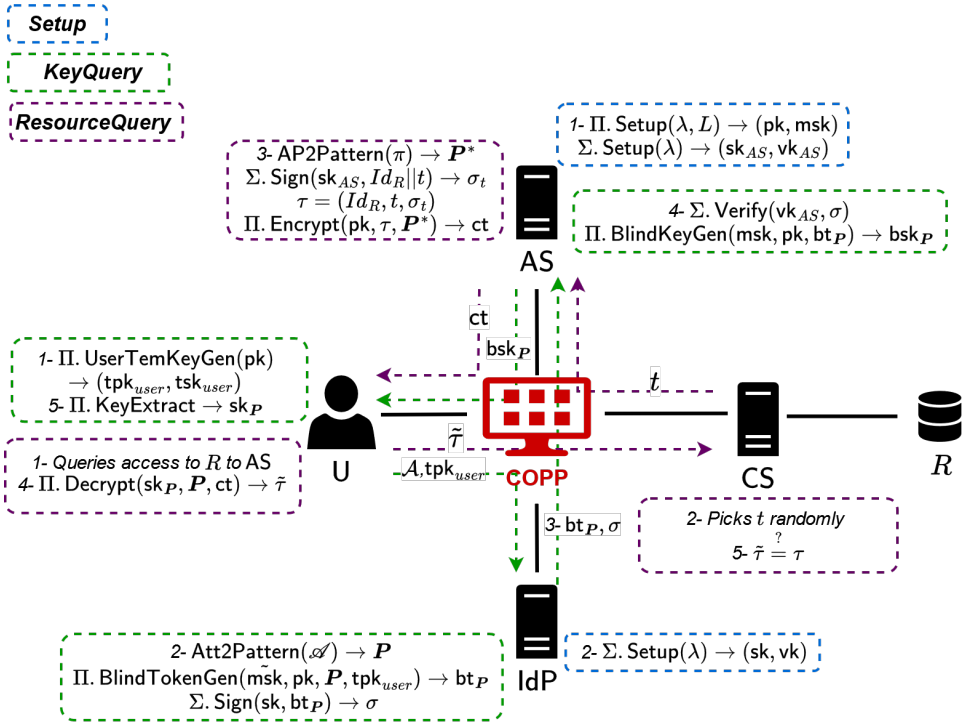


Figure 6.23: Our detailed protocol.

Secrecy of user's attributes. During the execution of a complete sequence, the only way for non-authorized to obtain any information about the user's attribute is to (i) corrupt the IdP (but such coalition is our of scope), or (ii) break the privacy-preserving key generation property of the WIBE scheme (which is assumed to be infeasible).

6.3.3 Our Concrete Solution: An Anonymous PPKG-WIBE

We now present a WIBE scheme that matches all the requirements defined above.

Core WIBE construction. The basis of our identity-based encryption with wildcard scheme is an idea given in [3] and presented in Section 4.2: the construction of an anonymous WIBE from an inner product encryption scheme. We however need to modify such construction since (i) their proposal considers a key derivation from another key, which we do not need and (ii) we have some specific restrictions regarding our patterns spaces.

Hiding the access policy. Based on the above core generic WIBE construction, our idea is then to build our own scheme on [91]'s IPE scheme, which satisfies *attribute-hiding* property (Definition 3.3.4), meaning that ciphertexts do not give information about the associated vector. Doing so, we protect the access policy almost for free. Another option would have been to use the construction by Okamoto and Takashima [114] or

by Chen *et al.* [51]. At the cost of an efficiency lost, those schemes achieve adaptive and attribute hiding security, based on a static and standard assumption, hence more secure than [91].
But with the current version of the scheme, we do not protect users attributes as in order to obtain a secret key, any user must send its attributes to the owner of the master secret key. Hence, we need to provide a privacy-preserving key generation such that the master secret key owner does not learn users attributes.

Privacy-preserving key generation. Our WIBE scheme presented in Figure 4.16, in Section 4.16 is a PPKG-WIBE scheme, based on the combination of Abdalla *et al.* [3] generic construction of anonymous WIBE from IPE and the Lewko *et al.* [91] attribute-hiding IPE scheme. Notice our scheme do not consider key derivation from another key. Combining this with the above paragraph, we have that our PPKG-WIBE scheme once adapted to our new patterns spaces is a good candidate to instantiate our generic solution for the use case.

Patterns spaces restrictions. Our PPKG-WIBE scheme (Figure 4.16) is dealing with patterns that belong to $\{0,1,\star\}^L$. For the use case, patterns spaces are equal to $\{0,1\}^L$ and $\{0,\star\}^L$ for ciphertexts and keys respectively. For algorithms ExtendingCtPattern and ExtendingKeyPatternRandomized this change is not an issue as $\{0,1\}^L \subset \{0,1,\star\}^L$ and $\{0,\star\}^L \subset \{0,1,\star\}^L$. We present in Figure 6.24 an example of the execution of both algorithms of such subsets. Therefore our PPKG-WIBE scheme can be used for the use case.

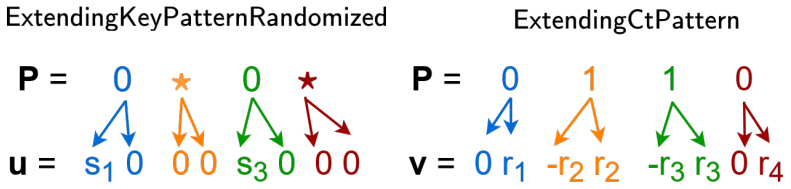


Figure 6.24: ExtendingKeyPatternRandomized and ExtendingCtPattern on a example.

6.4 Conclusion of This Chapter

This chapter introduced two data sharing schemes: *broadcast encryption* (and its variants) and *attribute-based encryption*. Our contributions regarding the first primitive is a generic construction of (augmented) broadcast encryption scheme from (pattern-hiding) identity-based encryption with wildcard, along with two new schemes: a constant

size ciphertext broadcast encryption scheme and an augmented broadcast encryption scheme, both proven to be adaptively secure in the standard model. As for attribute-based encryption, we first propose a new scheme that is the first one in the literature to have constant size ciphertext and secret keys. This ABE scheme is based on our dually computable accumulator scheme, presented in Section 5.4. We also investigate a use case centered on access control within the realm of connected devices, and propose an identity-based encryption with wildcard scheme that protects both the privacy of the access policies and users' attributes.

Regarding our ABE from dually computable accumulator, the size of its public key is a major drawback. This increase of the size is coming from the way we decided to represent sets of attributes and access policies and how we defined verification that an access policy is satisfied. Finding another way to represent them and to check if the attribute set verifies the access policy might lead to a more efficient scheme. We leave this question an open problem. Our construction relies strongly on some features of pairing-based accumulators (such as the use of the characteristic polynomial to represent a set of elements). That makes it unfortunately not generalizable. Finding a way to build attribute-based encryption from dually computable accumulators without relying on some specific features to obtain a generic construction is a challenging problem that we leave open. Another thing to improve regarding this construction is the complexity of supported access policies. Indeed currently our scheme is only dealing with disjunctions of conjunctions, which is not suitable for fine-grained access control. Building a ABE scheme from a dully computable accumulator that deals with complex access policies is a challenge for a future work.

Finally a last open question is about our data sharing schemes resistance to quantum computers. As we instantiate our identity-based encryption with wildcard schemes and accumulators schemes with pairing, our obtained broadcast encryption, augmented broadcast encryption and attribute-based encryption schemes are not quantum resistant. However, for the (augmented) broadcast encryption scheme as we provide a generic construction, building lattice-based (for example) identity-based encryption with wildcard schemes with all the required properties (and efficiently) will lead to (efficient) quantum resistant data sharing schemes. But building such identity-based encryption with wildcard schemes is quite challenging. As for accumulators, finding, for example, lattice-based accumulators that are efficient and can be dually computable might lead to an efficient quantum resistant attribute-based encryption scheme. However, currently there are only few accumulators based on lattices and they suffer from large accumulated values (or witnesses).

7

Conclusion

Contents

The conclusion of this thesis is divided into two parts: one that summarizes all of our contributions and another that addresses an open problem.

7.1 Our Results

This thesis presents two methods for data sharing: broadcast encryption and attribute-based encryption. The initial chapters, Chapters 2 and 3, provide the necessary mathematical background and cryptographic fundamentals. Chapter 4 focuses on identity-based encryption with wildcards, a primitive employed as a building block in Chapter 6 (specifically in Section 6.1) for broadcast encryption schemes. Additionally, we propose a generic construction of augmented broadcast encryption, a broadcast encryption variant, using the same kind of ideas. This construction is made possible through the introduction in Section 4.2 of the novel *pattern-hiding* security property. Leveraging our identity-based encryption with wildcards schemes from Section 4.3, including one with pattern-hiding capabilities, we create new (augmented) broadcast encryption schemes that are adaptively secure in the standard model. In Section 6.2, we present a novel attribute-based encryption scheme based on a cryptographic accumulator scheme introduced in Section 5.4. This new attribute-based encryption scheme represents the first instance in the literature with constant size ciphertexts and secret keys. The key idea behind its design is to exploit our new feature of accumulators, known as *dually computable*, as presented in Section 5.3. Finally, Section 6.3, explores

a specific use case wherein the data sharing scheme must adhere to varying levels of privacy. We propose a solution for this use case, utilizing a newly introduced feature of identity-based encryption with wildcards, termed *privacy preserving key generation* (introduced in Section 4.2).

The main focus of this thesis was to establish formal connections between various cryptographic primitives. While we were able to provide generic constructions for some of them, such as constructing broadcast encryption schemes from identity-based encryption with wildcards, there is still work to be done. For example, one open problem left unresolved by this thesis is the generic construction of attribute-based encryption schemes from dually computable accumulators. Another connection we attempted to prove is the relationship between accumulators and locally verifiable aggregate signature schemes [76]. Unfortunately, our attempts were unsuccessful, and we have presented the results of these attempts in a short article [24] at the CFail 2023 conference.

7.2 Locally Verifiable Aggregate Signatures and Accumulators

Here, we briefly describe locally verifiable aggregate signature schemes, along with another primitive called aggregate signature [37], to underscore the similarities with accumulators. In the sequel we consider the *single-signer* setting, meaning that aggregation of signatures is possible only when signatures were generated using the same verification key. We restrict our attention to this setting as there are similarities with cryptographic accumulators only when considering single-signer (locally verifiable) aggregate signature schemes.

Aggregate Signatures [37]. An aggregate signature scheme is a signature scheme (see Definition 3.4.1) that also provides two algorithms Aggregate and AggVerify, where

- Aggregate takes as input a verification key, along with a set of message-signature pairs and returns a shorter aggregate signature $\hat{\sigma}$;
- and AggVerify takes as input a verification key, a set of messages and an aggregate signature, and outputs 1 if the aggregate signature is valid with respect to the set of messages, 0 otherwise.

Aggregate signatures must satisfy *correctness* (as any signature scheme, refer to Definition 3.4.2) and must also satisfy *correctness of aggregation*, meaning that for all security parameters, all signing and verification keys generated honestly, all messages,

all signatures and all aggregate signatures generated honestly, the AggVerify algorithm outputs 1. Regarding efficiency, aggregate signature schemes have a *compactness of aggregation* requirement: the size of an aggregate signature is a fixed polynomial in the security parameter, independent of the number of aggregations. As for security, an aggregate signature scheme must satisfy *aggregated unforgeability*, which is the same definition as *unforgeability* of signature schemes (see Definition 3.4.3) except that now the adversary must produce a set of messages along with a forged aggregate signature.

Locally Verifiable Aggregate Signatures [76]. A locally verifiable aggregate signatures scheme (LVAS), is an aggregate signature scheme with two additional algorithms LocalOpen and LocalAggVerify such that

- LocalOpen takes as input a verification key, an aggregate signature, a set of $l \in \mathbb{N}$ messages and an index ind in $[l]$ and returns auxiliary information aux corresponding to the message of index ind;
- and LocalAggVerify takes as input a verification key, an aggregate signature, *one* message and auxiliary information associated, and returns 1 if the aggregate signature contains the signature of the message, 0 otherwise.

This local opening brings efficient verification: the verification algorithm takes as input *one* specific message instead of all messages, to prove that the signature of this message is indeed in the aggregate signature. A LVAS scheme must satisfy *correctness of local opening* meaning that for all security parameter, all signing and verification keys generated honestly, all messages, all honestly computed signatures and aggregate signatures, and all honestly generated auxiliary information, the LocalAggVerify algorithm returns 1. It must also satisfy *compactness of aggregation* as for aggregate signature schemes and *compactness of opening* that requires that the size of the auxiliary information is fixed polynomial in the security parameter, and is independent of the number of aggregations.
Regarding security, LVAS scheme must satisfy *aggregated unforgeability with adversarial opening*: this definition is similar to *unforgeability* of signature schemes (see Definition 3.4.3) except that this time the adversary must produce a tuple of aggregate signature-auxiliary information-message that passes the LocalAggVerify algorithm.

Similarities between signature and symmetric accumulator. We can easily build a signature scheme from an accumulator scheme with private evaluation and public generation. Indeed, set $\mathsf{sk} = \mathsf{sk}_{\mathsf{acc}}$, $\mathsf{vk} = \mathsf{pk}_{\mathsf{acc}}$, let the signature of a message m be the accumulator of the set $\{\mathsf{m}\}$ and let the signature verification be the accumulator verification algorithm. Correctness of the signature scheme comes straight from the

correctness of the accumulator scheme. The construction can be done the other way round: the accumulator scheme can be build from the signature scheme, however it results in a *bounded* accumulator scheme, with bound equals to 1 in our case. As for security, the question is more tricky: an adversary of the accumulator security will try to find another set that has the same accumulator (*i.e.* the same signature) as one given as challenge, while an adversary of the signature security will try to produce a new pair of message-signature. Thus we were not able to make the reduction from one to the other as both adversaries are requiring different inputs, and have different outputs that cannot be used to solve the other's security game.

Similarities between aggregate signatures and symmetric accumulator. Using an aggregate signature scheme to build a symmetric accumulator will solve the above problem of the bound equals to 1. Indeed, when the set to accumulate contains more than one element, the Eval algorithm runs the Sign algorithm on all elements of the set, then queries the Aggregate algorithm on all pairs of element-signature and returns the aggregate signature as the accumulator of the set. However, there is some issue when defining the verification algorithm Verify from the aggregate verification algorithm AggVerify: the latter is expecting as input a set of messages while the former is only expecting one element (i.e. one message). This difference of syntax traduces the fact that AggVerify is doing verification for all messages at the same time, while Verify is doing verification for one element (i.e. message) only. Thus the construction in this way is not working.
Now let us do the construction in the other way round. It is more complicated as we need to find a way to define Aggregate. Using an *additive* accumulator scheme [46] might be the answer, as such accumulator schemes provide an algorithm Add that takes as input the scheme (secret and) public key(s), a set of elements along with its accumulator and an element y to add, and returns an accumulator corresponding to the union of the original set and $\{y\}$. Then Aggregate is defined as follow: it runs Eval on one message of the set given as input, then runs the addition algorithm Add to obtain an accumulator of all messages, that will be output as an aggregate signature. However notice that the Aggregate algorithm takes as input only the verification key, therefore we need to use an accumulator scheme with *public* addition, meaning that Add only takes as input the accumulator public key and not the accumulator secret key. As for the aggregate verification algorithm AggVerify, we can define it such that it runs the accumulator verification algorithm Verify for all messages taken as input and outputs 0 if Verify returns 0 at least once.
Regarding security, our attempt to prove the security of the construction above through a security reduction was hindered by the differing inputs and outputs in the accumulator and aggregate signature security games, preventing us from completing the proof.

Additionally, we have observed a significant point concerning the construction mentioned above: in Theorem 5.2.1, we established that symmetric accumulator schemes result in accumulator sizes that grow linearly with the number of accumulated elements. Consequently, if we were able to build an aggregate signature scheme based on a symmetric accumulator, it would not satisfy the *compactness of aggregation* property, as an aggregate signature is treated as an accumulator within our construction.

To address the challenge posed by the varying inputs in the verification algorithms, one potential solution could involve using a locally verifiable aggregate signature scheme instead of an aggregate signature scheme. The former provides a verification algorithm for *a single* message only. However, it is important to note that this verification algorithm also requires auxiliary information as input, which symmetric accumulator schemes cannot provide. Asymmetric accumulator schemes, on the other hand, can provide this information through the use of witnesses. This leads us to a comparison between asymmetric accumulators and locally verifiable aggregate signature schemes.

Similarities between locally verifiable aggregate signatures and asymmetric accumulators. We can easily build an asymmetric accumulator scheme (with private evaluation and public key generation) from a LVAS scheme:

- the Gen algorithm runs the Setup algorithm;
- the Eval algorithm runs the Sign algorithm for each element of the set, then runs the Aggregate algorithm on all obtained signatures;
- the WitCreate algorithm runs the LocalOpen algorithm;
- and the Verify algorithm runs the LocalAggVerify algorithm.

Here all algorithm are consistent in the inputs/outputs, the correctness of the accumulator scheme directly comes from the LVAS *correctness of local opening* and the scheme has constant size accumulators and witnesses thanks to the LVAS *compactness of aggregation* and *compactness of local opening*.

The other way round, the construction is a little bit more complicated but still possible:

- Setup runs the accumulator scheme algorithm Gen;
- Sign runs the Eval algorithm on a singleton;
- Verify runs the algorithms WitCreate and Verify of the accumulator scheme;
- Aggregate runs the Eval algorithm on a set composed of one message then run the Add algorithm to include all the other messages' signatures;
- AggVerify runs the accumulator scheme algorithms WitCreate and Verify for each messages;
- LocalOpen runs the algorithm WitCreate;
- and LocalAggVerify runs the accumulator Verify algorithm.

In order for the construction to function properly, the accumulator scheme must possess private evaluation and public witness creation capabilities, and it must also be additive, with a publicly computable Add algorithm, as previously demonstrated with the symmetric accumulator. It is worth noting that if the accumulator scheme permits *subset queries* [59], meaning that witnesses can be generated for a subset of elements rather than just one element, then the efficiency of AggVerify improves. It is evident that the correctness of the LVAS is derived from the correctness of the accumulator scheme. As for the compactness of aggregation and local opening, these are assured if the accumulator scheme maintains constant-sized accumulators and witnesses.
Regarding security, the *unforgeability of local opening* property in the LVAS scheme appears somewhat analogous to the *collision resistance* security property of accumulator schemes (see Definition 5.1.3). However, there are intricacies in these reductions: while our property of *unforgeability of private evaluation* (see Definition 5.3.1) might prove useful in demonstrating the unforgeability of the aggregate signature, it lacks the inclusion of the challenge witness being forged in our property. To our knowledge, there is no security property in the cryptographic accumulator literature that guards against the forgery of both the accumulator and witnesses. And what about oracle queries? In the collision resistance security game, the adversary is granted the freedom to query the oracle for any sets without restrictions. In the aggregated unforgeability with adversarial opening security game, the adversary can query the oracle for any messages except the challenge message. When conducting security reductions from one adversary to the other, we encountered certain limitations that altered the adversary's advantage in winning the collision resistance security game, in a manner that we were unable to accurately assess.

Table 7.1: Summary of the different constructions. "A.", "S.", "$\surd$", "$\times$" and "$\approx$" respectively mean "asymmetric", "symmetric", "working", "not working", and "working under some conditions". In red we highlight the most inefficient construction, and in green the best construction we made.

From	To	Problems	Working	Security
S. accumulator	Signature	Existence	$\approx$	$\times$
Signature	S. accumulator	Bounded by 1	$\surd$	$\times$
(Additive) S. accumulator	Aggregate signature	Public Add, sizes	$\approx$	$\times$
Aggregate signature	S. accumulator	Verify/AggVerify	$\times$	$\times$
LVAS	A. accumulator	No	$\surd$	$\times$
(Additive) A. accumulator	LVAS	Public Add	$\approx$	$\times$

Conclusion. At first glance, cryptographic accumulators and locally verifiable aggregate signatures share many similarities. Table 7.1 summarizes the different constructions we made in this study. However, demonstrating that they are indeed the same is

not a straightforward task, even in the single-signer setting. Furthermore, addressing the multi-signer scenario presents significant challenges, as it would necessitate the creation of a novel class of cryptographic accumulators where accumulators computed from different keys could be “accumulated” together. This innovative development could have numerous potential applications, and we conclude this thesis by posing this intriguing open question.

8

Bibliography

Contents

References

[1] Michel Abdalla. “Brief Introduction to Provable Security”. In: 2014. URL: https://api.semanticscholar.org/CorpusID:10673744.

[2] Michel Abdalla, Angelo Caro, and Duong Phan. “Generalized Key Delegation for Wildcarded Identity-Based and Inner-Product Encryption”. In: *IEEE Transactions on Information Forensics and Security* 7 (Nov. 2012), pp. 1695–1706. DOI: 10.1109/TIFS.2012.2213594.

[3] Michel Abdalla, Angelo Caro, and Duong Phan. “Generalized Key Delegation for Wildcarded Identity-Based and Inner-Product Encryption”. In: *IEEE Transactions on Information Forensics and Security* 7 (Nov. 2012), pp. 1695–1706. DOI: 10.1109/TIFS.2012.2213594.

[4] Michel Abdalla, Dario Catalano, Alexander W. Dent, John Malone-Lee, Gregory Neven, and Nigel P. Smart. “Identity-Based Encryption Gone Wild”. In: *Automata, Languages and Programming*. Ed. by Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 300–311. ISBN: 978-3-540-35908-1.

[5] Masayuki Abe and Miguel Ambrona. “Blind Key-Generation Attribute-Based Encryption for General Predicates”. In: *Des. Codes Cryptography* 90.10 (Nov. 2022), pp. 2271–2299. ISSN: 0925-1022. DOI: 10.1007/s10623-022-01069-5. URL: https://doi.org/10.1007/s10623-022-01069-5.

[6] Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. “Structure-Preserving Signatures and Commitments to Group Elements”. In: *Advances in Cryptology – CRYPTO 2010*. Ed. by Tal Rabin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 209–236. ISBN: 978-3-642-14623-7.

[7] Tolga Acar and Lan Nguyen. *Revocation for Delegatable Anonymous Credentials*. Tech. rep. MSR-TR-2010-170. International Association for Cryptologic Research. Dec. 2010. URL: https://www.microsoft.com/en-us/

research/publication/revocation-for-delegatable-anonymous-credentials/.

[8] Tolga Acar and Lan Nguyen. “Revocation for Delegatable Anonymous Credentials”. In: *Public Key Cryptography – PKC 2011*. Ed. by Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 423–440. ISBN: 978-3-642-19379-8.

[9] Shweta Agrawal, Daniel Wichs, and Shota Yamada. “Optimal Broadcast Encryption from LWE and Pairings in the Standard Model”. In: *Theory of Cryptography*. Ed. by Rafael Pass and Krzysztof Pietrzak. Cham: Springer International Publishing, 2020, pp. 149–178. ISBN: 978-3-030-64375-1.

[10] Murat Ak, Serdar Pehlivanoğlu, and Ali Aydın Selçuk. “Anonymous trace and revoke”. In: *Journal of Computational and Applied Mathematics* 259 (2014). Recent Advances in Applied and Computational Mathematics: ICACM-IAM-METU, pp. 586–591. ISSN: 0377-0427. DOI: https://doi.org/10.1016/j.cam.2013.10.019. URL: https://www.sciencedirect.com/science/article/pii/S0377042713005633.

[11] Tomoyuki Asano. “A Revocation Scheme with Minimal Storage at Receivers”. In: *Advances in Cryptology — ASIACRYPT 2002*. Ed. by Yuliang Zheng. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 433–450. ISBN: 978-3-540-36178-7.

[12] Nuttapong Attrapadung. “Dual System Encryption Framework in Prime-Order Groups via Computational Pair Encodings”. In: *Advances in Cryptology – ASIACRYPT 2016*. Ed. by Jung Hee Cheon and Tsuyoshi Takagi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 591–623. ISBN: 978-3-662-53890-6.

[13] Nuttapong Attrapadung, Benoît Libert, and Elie de Panafieu. “Expressive Key-Policy Attribute-Based Encryption with Constant-Size Ciphertexts”. In: *Public Key Cryptography – PKC 2011*. Ed. by Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 90–108. ISBN: 978-3-642-19379-8.

[14] Man Ho Au, Patrick P. Tsang, Willy Susilo, and Yi Mu. “Dynamic Universal Accumulators for DDH Groups and Their Application to Attribute-Based Anonymous Credential Systems”. In: *Topics in Cryptology – CT-RSA 2009*. Ed. by Marc Fischlin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 295–308. ISBN: 978-3-642-00862-7.

[15] Man Ho Au, Qianhong Wu, Willy Susilo, and Yi Mu. “Compact E-Cash from Bounded Accumulator”. In: *Topics in Cryptology – CT-RSA 2007*. Ed. by Masayuki

Abe. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 178–195. ISBN: 978-3-540-69328-4.

[16] Foteini Baldimtsi, Ioanna Karantaidou, and Srinivasan Raghuraman. *Oblivious Accumulators*. Cryptology ePrint Archive, Paper 2023/1001. https://eprint.iacr.org/2023/1001. 2023. URL: https://eprint.iacr.org/2023/1001.

[17] A. Balu and Kaviya Kuppusamy. "Ciphertext Policy Attribute-based Encryption with anonymous access policy". In: *International Journal of Peer to Peer Networks* 1 (Nov. 2010). DOI: 10.5121/ijp2p.2010.1101.

[18] Niko Barić and Birgit Pfitzmann. "Collision-Free Accumulators and Fail-Stop Signature Schemes Without Trees". In: *Advances in Cryptology — EUROCRYPT '97*. Ed. by Walter Fumy. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 480–494. ISBN: 978-3-540-69053-5.

[19] Adam Barth, Dan Boneh, and Brent Waters. "Privacy in Encrypted Content Distribution Using Private Broadcast Encryption". In: *Financial Cryptography and Data Security*. Ed. by Giovanni Di Crescenzo and Avi Rubin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 52–64. ISBN: 978-3-540-46256-9.

[20] Anaïs Barthoulot, Olivier Blazy, and Sébastien Canard. "(Augmented) Broadcast Encryption from Identity-Based Encryption with Wildcard". In: *Cryptology and Network Security*. Ed. by Alastair R. Beresford, Arpita Patra, and Emanuele Bellini. Cham: Springer International Publishing, 2022, pp. 143–164. ISBN: 978-3-031-20974-1.

[21] Anaïs Barthoulot, Olivier Blazy, and Sébastien Canard. "(Augmented) Broadcast Encryption from Identity-Based Encryption with Wildcard". In: https://eprint.iacr.org/2022/1192. 2022. URL: https://eprint.iacr.org/2022/1192.

[22] Anaïs Barthoulot, Olivier Blazy, and Sébastien Canard. "Dually Computable Cryptographic Accumulators and Their Application to Attribute Based Encryption". In: *Cryptology and Network Security*. Ed. by Jing Deng, Vladimir Kolesnikov, and Alexander A. Schwarzmann. Singapore: Springer Nature Singapore, 2023, pp. 538–562. ISBN: 978-981-99-7563-1.

[23] Anaïs Barthoulot, Olivier Blazy, and Sébastien Canard. *Dually Computable Cryptographic Accumulators and Their Application to Attribute-Based Encryption*. Cryptology ePrint Archive, Paper 2023/1277. https://eprint.iacr.org/2023/1277. 2023. URL: https://eprint.iacr.org/2023/1277.

[24] Anaïs Barthoulot, Olivier Blazy, and Sébastien Canard. “Locally Verifiable Signatures and Cryptographic Accumulators: Different Names, Same Thing?” In: https://www.cfail.org/cfail2023. 2023. URL: https://www.cfail.org/cfail2023.

[25] Mihir Bellare and Phillip Rogaway. “Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols”. In: *Proceedings of the 1st ACM Conference on Computer and Communications Security*. CCS ’93. Fairfax, Virginia, USA: Association for Computing Machinery, 1993, pp. 62–73. ISBN: 0897916298. DOI: 10.1145/168588.168596. URL: https://doi.org/10.1145/168588.168596.

[26] Mihir Bellare, Brent Waters, and Scott Yilek. “Identity-Based Encryption Secure against Selective Opening Attack”. In: *Theory of Cryptography*. Ed. by Yuval Ishai. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 235–252. ISBN: 978-3-642-19571-6.

[27] Josh Benaloh and Michael de Mare. “One-Way Accumulators: A Decentralized Alternative to Digital Signatures”. In: *Advances in Cryptology — EUROCRYPT ’93*. Ed. by Tor Helleseth. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 274–285. ISBN: 978-3-540-48285-7.

[28] Alex Biryukov, Aleksei Udovenko, and Giuseppe Vitto. “Cryptanalysis of a Dynamic Universal Accumulator over Bilinear Groups”. In: *Topics in Cryptology – CT-RSA 2021*. Ed. by Kenneth G. Paterson. Cham: Springer International Publishing, 2021, pp. 276–298. ISBN: 978-3-030-75539-3.

[29] Olivier Blazy, Sayantan Mukherjee, Huyen Nguyen, Duong Hieu Phan, and Damien Stehlé. “An Anonymous Trace-and-Revoke Broadcast Encryption Scheme”. In: *Information Security and Privacy*. Ed. by Joonsang Baek and Sushmita Ruj. Cham: Springer International Publishing, 2021, pp. 214–233. ISBN: 978-3-030-90567-5.

[30] Dan Boneh. “The Decision Diffie-Hellman problem”. In: *Algorithmic Number Theory*. Ed. by Joe P. Buhler. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 48–63. ISBN: 978-3-540-69113-6.

[31] Dan Boneh and Xavier Boyen. “Short Signatures Without Random Oracles”. In: *Advances in Cryptology - EUROCRYPT 2004*. Ed. by Christian Cachin and Jan L. Camenisch. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 56–73. ISBN: 978-3-540-24676-3.

[32] Dan Boneh and Xavier Boyen. “Short Signatures Without Random Oracles and the SDH Assumption in Bilinear Groups”. In: *J. Cryptology* 21 (Apr. 2008), pp. 149–177. DOI: 10.1007/s00145-007-9005-7.

[33] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. “Hierarchical Identity Based Encryption with Constant Size Ciphertext”. In: *Advances in Cryptology – EUROCRYPT 2005*. Ed. by Ronald Cramer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 440–456. ISBN: 978-3-540-32055-5.

[34] Dan Boneh, Xavier Boyen, and Hovav Shacham. “Short Group Signatures”. In: *Advances in Cryptology – CRYPTO 2004*. Ed. by Matt Franklin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 41–55. ISBN: 978-3-540-28628-8.

[35] Dan Boneh, Benedikt Bünz, and Ben Fisch. “Batching Techniques for Accumulators with Applications to IOPs and Stateless Blockchains”. In: *Advances in Cryptology – CRYPTO 2019*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Cham: Springer International Publishing, 2019, pp. 561–586. ISBN: 978-3-030-26948-7.

[36] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. “Fully Key-Homomorphic Encryption, Arithmetic Circuit ABE and Compact Garbled Circuits”. In: *Advances in Cryptology – EUROCRYPT 2014*. Ed. by Phong Q. Nguyen and Elisabeth Oswald. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 533–556. ISBN: 978-3-642-55220-5.

[37] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. “Aggregate and Verifiably Encrypted Signatures from Bilinear Maps”. In: *Advances in Cryptology — EUROCRYPT 2003*. Ed. by Eli Biham. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 416–432. ISBN: 978-3-540-39200-2.

[38] Dan Boneh, Craig Gentry, and Brent Waters. “Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys”. In: *Advances in Cryptology – CRYPTO 2005*. Ed. by Victor Shoup. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 258–275. ISBN: 978-3-540-31870-5.

[39] Dan Boneh, Amit Sahai, and Brent Waters. “Fully Collusion Resistant Traitor Tracing with Short Ciphertexts and Private Keys”. In: *Advances in Cryptology - EUROCRYPT 2006*. Ed. by Serge Vaudenay. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 573–592. ISBN: 978-3-540-34547-3.

[40] Dan Boneh and Brent Waters. “A Fully Collusion Resistant Broadcast, Trace, and Revoke System”. In: *Proceedings of the 13th ACM Conference on Computer and Communications Security*. CCS ’06. Alexandria, Virginia, USA: Association

for Computing Machinery, 2006, pp. 211–220. ISBN: 1595935185. DOI: 10.1145/1180405.1180432. URL: https://doi.org/10.1145/1180405.1180432.

[41] Dan Boneh, Brent Waters, and Mark Zhandry. "Low Overhead Broadcast Encryption from Multilinear Maps". In: *Advances in Cryptology – CRYPTO 2014*. Ed. by Juan A. Garay and Rosario Gennaro. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 206–223. ISBN: 978-3-662-44371-2.

[42] Zvika Brakerski and Vinod Vaikuntanathan. *Lattice-Inspired Broadcast Encryption and Succinct Ciphertext-Policy ABE*. Cryptology ePrint Archive, Paper 2020/191. https://eprint.iacr.org/2020/191. 2020. URL: https://eprint.iacr.org/2020/191.

[43] Ahto Buldas, Peeter Laud, and Helger Lipmaa. "Eliminating Counterevidence with Applications to Accountable Certificate Management". In: *Journal of Computer Security* 10 (Aug. 2002), pp. 273–296. DOI: 10.3233/JCS-2002-10304.

[44] Philippe Camacho, Alejandro Hevia, Marcos Kiwi, and Roberto Opazo. "Strong Accumulators from Collision-Resistant Hashing". In: *Information Security*. Ed. by Tzong-Chen Wu, Chin-Laung Lei, Vincent Rijmen, and Der-Tsai Lee. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 471–486. ISBN: 978-3-540-85886-7.

[45] Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. "An Accumulator Based on Bilinear Maps and Efficient Revocation for Anonymous Credentials". In: *Public Key Cryptography – PKC 2009*. Ed. by Stanisław Jarecki and Gene Tsudik. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 481–500. ISBN: 978-3-642-00468-1.

[46] Jan Camenisch and Anna Lysyanskaya. "Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials". In: *Advances in Cryptology — CRYPTO 2002*. Ed. by Moti Yung. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 61–76. ISBN: 978-3-540-45708-4.

[47] David Chaum. "Blind Signatures for Untraceable Payments". In: *Advances in Cryptology*. Ed. by David Chaum, Ronald L. Rivest, and Alan T. Sherman. Boston, MA: Springer US, 1983, pp. 199–203. ISBN: 978-1-4757-0602-4.

[48] David Chaum. "Security without Identification: Transaction Systems to Make Big Brother Obsolete". In: *Commun. ACM* 28.10 (Oct. 1985), pp. 1030–1044. ISSN: 0001-0782. DOI: 10.1145/4372.4373. URL: https://doi.org/10.1145/4372.4373.

[49] Jie Chen, Romain Gay, and Hoeteck Wee. “Improved Dual System ABE in Prime-Order Groups via Predicate Encodings”. In: *Advances in Cryptology - EUROCRYPT 2015*. Ed. by Elisabeth Oswald and Marc Fischlin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 595–624. ISBN: 978-3-662-46803-6.

[50] Jie Chen, Junqing Gong, Lucas Kowalczyk, and Hoeteck Wee. “Unbounded ABE via Bilinear Entropy Expansion, Revisited”. In: *Advances in Cryptology – EUROCRYPT 2018*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Cham: Springer International Publishing, 2018, pp. 503–534. ISBN: 978-3-319-78381-9.

[51] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Hoeteck Wee. “Shorter IBE and Signatures via Asymmetric Pairings”. In: *Pairing-Based Cryptography – Pairing 2012*. Ed. by Michel Abdalla and Tanja Lange. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 122–140. ISBN: 978-3-642-36334-4.

[52] Benny Chor, Amos Fiat, and Moni Naor. “Tracing Traitors”. In: *Advances in Cryptology — CRYPTO ’94*. Ed. by Yvo G. Desmedt. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 257–270. ISBN: 978-3-540-48658-9.

[53] Geoffroy Couteau, Helger Lipmaa, Roberto Parisella, and Arne Tobias Ødegaard. “Efficient NIZKs for Algebraic Sets”. In: *Advances in Cryptology – ASIACRYPT 2021*. Ed. by Mehdi Tibouchi and Huaxiong Wang. Cham: Springer International Publishing, 2021, pp. 128–158. ISBN: 978-3-030-92078-4.

[54] Ivan Damgård and Nikos Triandopoulos. “Supporting Non-membership Proofs with Bilinear-map Accumulators”. In: *IACR Cryptol. ePrint Arch.* 2008 (2008), p. 538. URL: https://api.semanticscholar.org/CorpusID:14263646.

[55] Ivan Damgård and Nikos Triandopoulos. “Supporting Non-membership Proofs with Bilinear-map Accumulators.” In: *IACR Cryptology ePrint Archive* 2008 (Jan. 2008), p. 538.

[56] Hermann De Meer, Manuel Liedel, Henrich C. Poehls, Joachim Posegga, and Kai Samelin. *Indistinguishability of one-way accumulators*. Tech. rep. MIP-1210. Dec. 2012. URL: https://www.fim.uni-passau.de/fileadmin/dokumente/fakultaeten/fim/lehrstuhl/meer/publications/pdf/DeMeer2012a.pdf.

[57] David Derler, Christian Hanser, and Daniel Slamanig. “Revisiting Cryptographic Accumulators, Additional Properties and Relations to Other Primitives”. In: *Topics in Cryptology — CT-RSA 2015*. Ed. by Kaisa Nyberg. Cham: Springer International Publishing, 2015, pp. 127–144. ISBN: 978-3-319-16715-2.

[58] W. Diffie and M. Hellman. “New directions in cryptography”. In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654. DOI: 10.1109/TIT.1976.1055638.

[59] Yevgeniy Dodis, Aggelos Kiayias, Antonio Nicolosi, and Victor Shoup. “Anonymous Identification in Ad Hoc Groups”. In: *Advances in Cryptology - EUROCRYPT 2004*. Ed. by Christian Cachin and Jan L. Camenisch. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 609–626. ISBN: 978-3-540-24676-3.

[60] Danny Dolev, Cynthia Dwork, and Moni Naor. “Non-Malleable Cryptography”. In: *SIAM Journal of Computing* 30 (Mar. 2001). DOI: 10.1145/103418.103474.

[61] K. Elbassioni, K. Makino, and I Rauf. “On the readability of monotone Boolean formulae”. In: *Journal of Combinatorial Optimization* (2011), pp. 293–304.

[62] Prastudy Fauzi, Helger Lipmaa, and Bingsheng Zhang. “Efficient Non-Interactive Zero Knowledge Arguments for Set Operations”. In: *Financial Cryptography and Data Security*. Ed. by Nicolas Christin and Reihaneh Safavi-Naini. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 216–233. ISBN: 978-3-662-45472-5.

[63] Nelly Fazio and Antonio Nicolosi. “Cryptographic Accumulators: Definitions, Constructions and Applications”. In: 2002.

[64] David Ferraiolo and Kuhn D. “Role-Based Access Control”. In: *15th National Computer Security Conference* (1992), pp. 554–563.

[65] Anna Lisa Ferrara, Georg Fachsbauer, Bin Liu, and Bogdan Warinschi. “Policy Privacy in Cryptographic Access Control”. In: *2015 IEEE 28th Computer Security Foundations Symposium*. 2015, pp. 46–60. DOI: 10.1109/CSF.2015.11.

[66] Amos Fiat and Moni Naor. “Broadcast Encryption”. In: *Advances in Cryptology — CRYPTO’ 93*. Ed. by Douglas R. Stinson. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 480–491. ISBN: 978-3-540-48329-8.

[67] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. “Efficient Private Matching and Set Intersection”. In: *Advances in Cryptology - EUROCRYPT 2004*. Ed. by Christian Cachin and Jan L. Camenisch. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 1–19. ISBN: 978-3-540-24676-3.

[68] Steven Galbraith, Florian Hess, and Frederik Vercauteren. “Aspects of Pairing Inversion.” In: *IEEE Transactions on Information Theory* 54 (Jan. 2008), pp. 5719–5728.

[69] Steven D. Galbraith, Kenneth G. Paterson, and Nigerl P. Smart. *Pairings for Cryptographers*. Cryptology ePrint Archive, Paper 2006/165. https://eprint.

iacr.org/2006/165. 2006. URL: https://eprint.iacr.org/2006/165.

[70] Sanjam Garg, Abishek Kumarasubramanian, Amit Sahai, and Brent Waters. "Building Efficient Fully Collusion-Resilient Traitor Tracing and Revocation Schemes". In: *Proceedings of the 17th ACM Conference on Computer and Communications Security*. CCS 10. Chicago, Illinois, USA: Association for Computing Machinery, 2010, pp. 121–130. ISBN: 9781450302456. DOI: 10.1145/1866307.1866322. URL: https://doi.org/10.1145/1866307.1866322.

[71] Romain Gay, Lucas Kowalczyk, and Hoeteck Wee. "Tight Adaptively Secure Broadcast Encryption with Short Ciphertexts and Keys". In: *Security and Cryptography for Networks*. Ed. by Dario Catalano and Roberto De Prisco. Cham: Springer International Publishing, 2018, pp. 123–139. ISBN: 978-3-319-98113-0.

[72] Craig Gentry and Zulfikar Ramzan. "RSA Accumulator-Based Broadcast Encryption". In: *Information Security*. Ed. by Kan Zhang and Yuliang Zheng. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 73–86. ISBN: 978-3-540-30144-8.

[73] Esha Ghosh, Olga Ohrimenko, Dimitrios Papadopoulos, Roberto Tamassia, and Nikos Triandopoulos. "Zero-Knowledge Accumulators and Set Algebra". In: *Advances in Cryptology – ASIACRYPT 2016*. Ed. by Jung Hee Cheon and Tsuyoshi Takagi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 67–100. ISBN: 978-3-662-53890-6.

[74] Shafi Goldwasser and Silvio Micali. "Probabilistic encryption". In: *Journal of Computer and System Sciences* 28.2 (1984), pp. 270–299. ISSN: 0022-0000. DOI: https://doi.org/10.1016/0022-0000(84)90070-9. URL: https://www.sciencedirect.com/science/article/pii/0022000084900709.

[75] Rishab Goyal, Willy Quach, Brent Waters, and Daniel Wichs. "Broadcast and Trace with N^{ϵ} Ciphertext Size from Standard Assumptions". In: *Advances in Cryptology – CRYPTO 2019*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Cham: Springer International Publishing, 2019, pp. 826–855. ISBN: 978-3-030-26954-8.

[76] Rishab Goyal and Vinod Vaikuntanathan. "Locally Verifiable Signature and Key Aggregation". In: *Advances in Cryptology – CRYPTO 2022*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Cham: Springer Nature Switzerland, 2022, pp. 761–791. ISBN: 978-3-031-15979-4.

[77] Rishab Goyal, Satyanarayana Vusirikala, and Brent Waters. "Collusion Resistant Broadcast and Trace from Positional Witness Encryption". In: *Public-Key Cryp-*

tography – PKC 2019. Ed. by Dongdai Lin and Kazue Sako. Cham: Springer International Publishing, 2019, pp. 3–33. ISBN: 978-3-030-17259-6.

[78] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data". In: *Proceedings of the 13th ACM Conference on Computer and Communications Security*. CCS '06. Alexandria, Virginia, USA: Association for Computing Machinery, 2006, pp. 89–98. ISBN: 1595935185. DOI: 10.1145/1180405.1180418. URL: https://doi.org/10.1145/1180405.1180418.

[79] Jens Groth. "On the Size of Pairing-Based Non-interactive Arguments". In: *Advances in Cryptology – EUROCRYPT 2016*. Ed. by Marc Fischlin and Jean-Sébastien Coron. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 305–326. ISBN: 978-3-662-49896-5.

[80] Jens Groth. "Short Pairing-Based Non-interactive Zero-Knowledge Arguments". In: *Advances in Cryptology - ASIACRYPT 2010*. Ed. by Masayuki Abe. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 321–340. ISBN: 978-3-642-17373-8.

[81] Jens Groth and Amit Sahai. "Efficient Non-interactive Proof Systems for Bilinear Groups". In: *Advances in Cryptology – EUROCRYPT 2008*. Ed. by Nigel Smart. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 415–432. ISBN: 978-3-540-78967-3.

[82] Mahabir Prasad Jhanwar and Reihaneh Safavi-Naini. "Compact Accumulator Using Lattices". In: *Security, Privacy, and Applied Cryptography Engineering*. Ed. by Rajat Subhra Chakraborty, Peter Schwabe, and Jon Solworth. Cham: Springer International Publishing, 2015, pp. 347–358. ISBN: 978-3-319-24126-5.

[83] Hongyong Jia, Yue Chen, Julong Lan, Kaixiang Huang, and Jun Wang. "Efficient revocable hierarchical identity-based encryption using cryptographic accumulators". In: *International Journal of Information Security* (2018).

[84] Jonathan Katz, Amit Sahai, and Brent Waters. "Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products". In: *Advances in Cryptology – EUROCRYPT 2008*. Ed. by Nigel Smart. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 146–162. ISBN: 978-3-540-78967-3.

[85] Jihye Kim, Seunghwa Lee, Jiwon Lee, and Hyunok Oh. "Scalable Wildcarded Identity-Based Encryption". In: *Computer Security*. Ed. by Javier Lopez, Jianying Zhou, and Miguel Soriano. Cham: Springer International Publishing, 2018, pp. 269–287. ISBN: 978-3-319-98989-1.

[86] Jihye Kim, Seunghwa Lee, Jiwon Lee, and Hyunok Oh. “Scalable Wildcarded Identity-Based Encryption”. In: *Computer Security*. Ed. by Javier Lopez, Jianying Zhou, and Miguel Soriano. Cham: Springer International Publishing, 2018, pp. 269–287. ISBN: 978-3-319-98989-1.

[87] Lucas Kowalczyk and Hoeteck Wee. “Compact Adaptively Secure ABE for NC^1 from k-Lin”. In: *Advances in Cryptology – EUROCRYPT 2019*. Ed. by Yuval Ishai and Vincent Rijmen. Cham: Springer International Publishing, 2019, pp. 3–33. ISBN: 978-3-030-17653-2.

[88] Amrit Kumar, Pascal Lafourcade, and Cédric Lauradoux. “Performances of Cryptographic Accumulators”. In: (May 2014). DOI: 10.1109/LCN.2014.6925793.

[89] Allison Lewko. “Functional encryption : new proof techniques and advancing capabilities”. Phd Thesis. The University of Texas at Austin, May 2012. URL: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=ecb5e57fbd375f246d0414c96730d5427b332f7c.

[90] Allison Lewko. “Tools for Simulating Features of Composite Order Bilinear Groups in the Prime Order Setting”. In: *Advances in Cryptology – EUROCRYPT 2012*. Ed. by David Pointcheval and Thomas Johansson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 318–335. ISBN: 978-3-642-29011-4.

[91] Allison Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. “Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption”. In: *Advances in Cryptology – EUROCRYPT 2010*. Ed. by Henri Gilbert. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 62–91. ISBN: 978-3-642-13190-5.

[92] Allison Lewko, Amit Sahai, and Brent Waters. “Revocation Systems with Very Small Private Keys”. In: *Proceedings of the 2010 IEEE Symposium on Security and Privacy*. SP ’10. USA: IEEE Computer Society, 2010, pp. 273–285. ISBN: 9780769540351. DOI: 10.1109/SP.2010.23. URL: https://doi.org/10.1109/SP.2010.23.

[93] Allison Lewko and Brent Waters. “Decentralizing Attribute-Based Encryption”. In: *Advances in Cryptology – EUROCRYPT 2011*. Ed. by Kenneth G. Paterson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 568–588. ISBN: 978-3-642-20465-4.

[94] Allison Lewko and Brent Waters. “New Proof Methods for Attribute-Based Encryption: Achieving Full Security through Selective Techniques”. In: *Advances in Cryptology – CRYPTO 2012*. Ed. by Reihaneh Safavi-Naini and Ran Canetti.

Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 180–198. ISBN: 978-3-642-32009-5.

[95] Allison Lewko and Brent Waters. “New Techniques for Dual System Encryption and Fully Secure HIBE with Short Ciphertexts”. In: *Theory of Cryptography*. Ed. by Daniele Micciancio. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 455–479. ISBN: 978-3-642-11799-2.

[96] Allison Lewko and Brent Waters. “Unbounded HIBE and Attribute-Based Encryption”. In: *Advances in Cryptology – EUROCRYPT 2011*. Ed. by Kenneth G. Paterson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 547–567. ISBN: 978-3-642-20465-4.

[97] Fagen Li, Yupu Hu, and Chuanrong Zhang. “An Identity-Based Signcryption Scheme for Multi-domain Ad Hoc Networks”. In: *Applied Cryptography and Network Security*. Ed. by Jonathan Katz and Moti Yung. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 373–384. ISBN: 978-3-540-72738-5.

[98] Jiangtao Li, Ninghui Li, and Rui Xue. “Universal Accumulators with Efficient Nonmembership Proofs”. In: *Applied Cryptography and Network Security*. Ed. by Jonathan Katz and Moti Yung. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 253–269. ISBN: 978-3-540-72738-5.

[99] Benoît Libert, San Ling, Khoa Nguyen, and Huaxiong Wang. “Zero-Knowledge Arguments for Lattice-Based Accumulators: Logarithmic-Size Ring Signatures and Group Signatures Without Trapdoors”. In: *Advances in Cryptology – EUROCRYPT 2016*. Ed. by Marc Fischlin and Jean-Sébastien Coron. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 1–31. ISBN: 978-3-662-49896-5.

[100] Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. “Anonymous Broadcast Encryption: Adaptive Security and Efficient Constructions in the Standard Model”. In: *Public Key Cryptography – PKC 2012*. Ed. by Marc Fischlin, Johannes Buchmann, and Mark Manulis. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 206–224. ISBN: 978-3-642-30057-8.

[101] Benoît Libert, Somindu C. Ramanna, and Moti Yung. “Functional Commitment Schemes: From Polynomial Commitments to Pairing-Based Accumulators from Simple Assumptions”. In: *International Colloquium on Automata, Languages and Programming*. 2016.

[102] Helger Lipmaa. “Secure Accumulators from Euclidean Rings without Trusted Setup”. In: *Applied Cryptography and Network Security*. Ed. by Feng Bao, Pierangela Samarati, and Jianying Zhou. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 224–240. ISBN: 978-3-642-31284-7.

[103] Helger Lipmaa and Roberto Parisella. *Set (Non-)Membership NIZKs from Determinantal Accumulators*. Cryptology ePrint Archive, Paper 2022/1570. https://eprint.iacr.org/2022/1570. 2022. URL: https://eprint.iacr.org/2022/1570.

[104] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. "Attribute-Based Signatures". In: *Topics in Cryptology – CT-RSA 2011*. Ed. by Aggelos Kiayias. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 376–392. ISBN: 978-3-642-19074-2.

[105] Ueli Maurer. "Abstract Models of Computation in Cryptography". In: *Cryptography and Coding*. Ed. by Nigel P. Smart. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 1–12. ISBN: 978-3-540-32418-8.

[106] Hermann de Meer, Henrich C. Pöhls, Joachim Posegga, and Kai Samelin. "Redactable Signature Schemes for Trees with Signer-Controlled Non-Leaf-Redactions". In: *E-Business and Telecommunications*. Ed. by Mohammad S. Obaidat and Joaquim Filipe. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 155–171. ISBN: 978-3-662-44791-8.

[107] Yan Michalevsky and Marc Joye. "Decentralized Policy-Hiding ABE with Receiver Privacy". In: *Computer Security*. Ed. by Javier Lopez, Jianying Zhou, and Miguel Soriano. Cham: Springer International Publishing, 2018, pp. 548–567. ISBN: 978-3-319-98989-1.

[108] Eric Miles, Amit Sahai, and Mark Zhandry. "Annihilation Attacks for Multilinear Maps: Cryptanalysis of Indistinguishability Obfuscation over GGH13". In: Aug. 2016, pp. 629–658. ISBN: 978-3-662-53007-8. DOI: 10.1007/978-3-662-53008-5_22.

[109] Gordon E. Moore. "Cramming more components onto integrated circuits, Reprinted from Electronics, volume 38, number 8, April 19, 1965, pp.114 ff." In: *IEEE Solid-State Circuits Society Newsletter* 11.3 (2006), pp. 33–35. DOI: 10.1109/N-SSC.2006.4785860.

[110] Sascha Müller and Stefan Katzenbeisser. "Hiding the Policy in Cryptographic Access Control". In: *Security and Trust Management*. Ed. by Catherine Meadows and Carmen Fernandez-Gago. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 90–105. ISBN: 978-3-642-29963-6.

[111] M. Naor and M. Yung. "Public-Key Cryptosystems Provably Secure against Chosen Ciphertext Attacks". In: *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*. STOC '90. Baltimore, Maryland, USA: Association for Computing Machinery, 1990, pp. 427–437. ISBN: 0897913612.

DOI: 10.1145/100216.100273. URL: https://doi.org/10.1145/100216.100273.

[112] Moni Naor and Benny Pinkas. “Efficient Trace and Revoke Schemes”. In: *Proceedings of the 4th International Conference on Financial Cryptography*. FC ’00. Berlin, Heidelberg: Springer-Verlag, 2000, pp. 1–20. ISBN: 3540427007.

[113] Lan Nguyen. “Accumulators from Bilinear Pairings and Applications”. In: *Topics in Cryptology – CT-RSA 2005*. Ed. by Alfred Menezes. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 275–292. ISBN: 978-3-540-30574-3.

[114] Tatsuaki Okamoto and Katsuyuki Takashima. “Adaptively Attribute-Hiding (Hierarchical) Inner Product Encryption”. In: *Advances in Cryptology – EUROCRYPT 2012*. Ed. by David Pointcheval and Thomas Johansson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 591–608. ISBN: 978-3-642-29011-4.

[115] Tatsuaki Okamoto and Katsuyuki Takashima. “Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption”. In: *Advances in Cryptology – CRYPTO 2010*. Ed. by Tal Rabin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 191–208. ISBN: 978-3-642-14623-7.

[116] Tatsuaki Okamoto and Katsuyuki Takashima. “Fully Secure Unbounded Inner-Product and Attribute-Based Encryption”. In: *Advances in Cryptology – ASIACRYPT 2012*. Ed. by Xiaoyun Wang and Kazue Sako. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 349–366. ISBN: 978-3-642-34961-4.

[117] Tatsuaki Okamoto and Katsuyuki Takashima. “Hierarchical Predicate Encryption for Inner-Products”. In: *Advances in Cryptology – ASIACRYPT 2009*. Ed. by Mitsuru Matsui. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 214–231. ISBN: 978-3-642-10366-7.

[118] Tatsuaki Okamoto and Katsuyuki Takashima. “Homomorphic Encryption and Signatures from Vector Decomposition”. In: *Pairing-Based Cryptography – Pairing 2008*. Ed. by Steven D. Galbraith and Kenneth G. Paterson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 57–74. ISBN: 978-3-540-85538-5.

[119] Ilker Ozcelik, Sai Medury, Justin Broaddus, and Anthony Skjellum. “An Overview of Cryptographic Accumulators”. In: Jan. 2021, pp. 661–669. DOI: 10.5220/0010337806610669.

[120] Duong Hieu Phan. “Some Advances in Broadcast Encryption and Traitor Tracing”. Habilitation à diriger des recherches. Ecole normale supérieure - ENS PARIS, Nov. 2014. URL: https://tel.archives-ouvertes.fr/tel-02384086.

[121] David Pointcheval and Jacques Stern. “Security Arguments for Digital Signatures and Blind Signatures”. In: *Journal of Cryptology* 13 (Oct. 2001). DOI: 10.1007/s001450010003.

[122] Charles Rackoff and Daniel R. Simon. “Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack”. In: *Advances in Cryptology — CRYPTO ’91*. Ed. by Joan Feigenbaum. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 433–444. ISBN: 978-3-540-46766-3.

[123] Yongjun Ren, Xinyu Liu, Qiang Wu, Ling Wang, and Weijian Zhang. “Cryptographic Accumulator and Its Application: A Survey”. In: *Security and Communication Networks* 2022 (Mar. 2022), pp. 1–13. DOI: 10.1155/2022/5429195.

[124] Leonid Reyzin and Sophia Yakoubov. “Efficient Asynchronous Accumulators for Distributed PKI”. In: *Security and Cryptography for Networks*. Ed. by Vassilis Zikas and Roberto De Prisco. Cham: Springer International Publishing, 2016, pp. 292–309. ISBN: 978-3-319-44618-9.

[125] Leonid Reyzin and Sophia Yakoubov. “Efficient Asynchronous Accumulators for Distributed PKI”. In: *Security and Cryptography for Networks*. Ed. by Vassilis Zikas and Roberto De Prisco. Cham: Springer International Publishing, 2016, pp. 292–309. ISBN: 978-3-319-44618-9.

[126] Amit Sahai and Brent Waters. “Fuzzy Identity-Based Encryption”. In: *Advances in Cryptology – EUROCRYPT 2005*. Ed. by Ronald Cramer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 457–473. ISBN: 978-3-540-32055-5.

[127] R.S. Sandhu, E.J. Coyne, H.L. Feinstein, and C.E. Youman. “Role-based access control models”. In: *Computer* 29.2 (1996), pp. 38–47. DOI: 10.1109/2.485845.

[128] C. E. Shannon. “Communication theory of secrecy systems”. In: *The Bell System Technical Journal* 28.4 (1949), pp. 656–715. DOI: 10.1002/j.1538-7305.1949.tb00928.x.

[129] Peter W. Shor. “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”. In: *SIAM J. Comput.* 26.5 (Oct. 1997), pp. 1484–1509. ISSN: 0097-5397. DOI: 10.1137/S0097539795293172. URL: https://doi.org/10.1137/S0097539795293172.

[130] Victor Shoup. *Sequences of games: a tool for taming complexity in security proofs*. Cryptology ePrint Archive, Paper 2004/332. https://eprint.iacr.org/2004/332. 2004. URL: https://eprint.iacr.org/2004/332.

[131] Yujiao Song, Hao Wang, Xiaochao Wei, and Lei Wu. “Efficient Attribute-Based Encryption with Privacy-Preserving Key Generation and Its Application in In-

dustrial Cloud". In: *Security and Communication Networks* 2019 (May 2019), pp. 1–9. DOI: 10.1155/2019/3249726.

[132] Shravan Srinivasan, Ioanna Karantaidou, Foteini Baldimtsi, and Charalampos Papamanthou. "Batching, Aggregation, and Zero-Knowledge Proofs in Bilinear Accumulators". In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. CCS '22. Los Angeles, CA, USA: Association for Computing Machinery, 2022, pp. 2719–2733. ISBN: 9781450394505. DOI: 10.1145/3548606.3560676. URL: https://doi.org/10.1145/3548606.3560676.

[133] Naoki Tanaka and Taiichi Saito. "On the q-Strong Diffie-Hellman Problem". In: *IACR Cryptol. ePrint Arch.* 2010 (2010), p. 215.

[134] Peishun Wang, Huaxiong Wang, and Josef Pieprzyk. "A New Dynamic Accumulator for Batch Updates". In: *Information and Communications Security*. Ed. by Sihan Qing, Hideki Imai, and Guilin Wang. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 98–112. ISBN: 978-3-540-77048-0.

[135] Xiuhua Wang and Sherman S. M. Chow. "Cross-Domain Access Control Encryption: Arbitrary-policy, Constant-size, Efficient". In: *2021 IEEE Symposium on Security and Privacy (SP)*. 2021, pp. 748–761. DOI: 10.1109/SP40001.2021.00023.

[136] Brent Waters. "Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization". In: *Public Key Cryptography – PKC 2011*. Ed. by Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 53–70. ISBN: 978-3-642-19379-8.

[137] Brent Waters. "Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions". In: *Advances in Cryptology - CRYPTO 2009*. Ed. by Shai Halevi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 619–636. ISBN: 978-3-642-03356-8.

[138] Umesh Chandra Yadav and Syed Taqi Ali. "Ciphertext Policy-Hiding Attribute-Based Encryption". In: *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. 2015, pp. 2067–2071. DOI: 10.1109/ICACCI.2015.7275921.

[139] Mark Zhandry. "New Techniques for Traitor Tracing: Size $N^{1/3}$ and More from Pairings". In: *Advances in Cryptology – CRYPTO 2020*. Ed. by Daniele Micciancio and Thomas Ristenpart. Cham: Springer International Publishing, 2020, pp. 652–682. ISBN: 978-3-030-56784-2.

List of Publications

International Conferences with Peer Review

- Anaïs Barthoulot, Olivier Blazy, and Sébastien Canard. "(Augmented) Broadcast Encryption from Identity-Based Encryption with Wildcard". In: *Cryptology and Network Security*. Ed. by Alastair R. Beresford, Arpita Patra, and Emanuele Bellini. Cham: Springer International Publishing, 2022, pp. 143–164. ISBN: 978-3-031-20974-1.
- Anaïs Barthoulot, Olivier Blazy, and Sébastien Canard. "Dually Computable Cryptographic Accumulators and Their Application to Attribute Based Encryption". In: *Cryptology and Network Security*. Ed. by Jing Deng, Vladimir Kolesnikov, and Alexander A. Schwarzmann. Singapore: Springer Nature Singapore, 2023, pp. 538–562. ISBN: 978-981-99-7563-1.

Eprint Versions

- Anaïs Barthoulot, Olivier Blazy, and Sébastien Canard. "(Augmented) Broadcast Encryption from Identity-Based Encryption with Wildcard". In: https://eprint.iacr.org/2022/1192. 2022. URL: https://eprint.iacr.org/2022/1192.
- Anaïs Barthoulot, Olivier Blazy, and Sébastien Canard. "Dually Computable Cryptographic Accumulators and Their Application to Attribute-Based Encryption". In: https://eprint.iacr.org/2023/1277. 2023. URL: https://eprint.iacr.org/2023/1277.

International Conferences with Peer Review, without Proceedings

- Anaïs Barthoulot, Olivier Blazy, and Sébastien Canard. “Locally Verifiable Signatures and Cryptographic Accumulators: Different Names, Same Thing?” In: https://www.cfail.org/cfail2023. 2023. URL: https://www.cfail.org/cfail2023.

Chiffrement avancé pour le partage de données sensibles

Résumé : Les données, y compris les données privées, jouent aujourd'hui un rôle prépondérant dans notre quotidien. Les recherches actuelles se concentrent principalement sur le stockage de ces données, en mettant l'accent sur la possibilité de les traiter de manière sécurisée même lorsqu'elles sont chiffrées. Cependant, au-delà de leur conservation, ces données doivent également être partagées de diverses manières : soit entre un individu et un groupe d'individus, parfois unis par des caractéristiques communes qui définissent les règles de partage, soit simplement entre deux individus. À l'heure actuelle, ces différents modes de partage ne sont pas encore bien maîtrisés, que ce soit en raison de leur coût élevé en termes de performance ou de leurs fonctionnalités limitées. Cette thèse se penche sur divers schémas de chiffrement adaptés au partage de données sensibles, en proposant de nouvelles constructions. Tout d'abord, nous examinons deux primitives cryptographiques : les schémas de chiffrement basés sur l'identité avec caractère générique et les accumulateurs cryptographiques, qui serviront de point de départ pour nos nouvelles constructions. En ce qui concerne les schémas de chiffrement basés sur l'identité avec caractère générique, nous introduisons une nouvelle propriété de sécurité et proposons deux nouvelles instanciations, dont l'une satisfait cette nouvelle propriété de sécurité que nous avons définie. Pour les accumulateurs cryptographiques, nous présentons un nouveau type d'accumulateur, ainsi qu'un schéma amélioré par rapport à l'état de l'art, et un deuxième schéma illustrant notre nouvelle fonctionnalité. Nous introduisons également une nouvelle propriété de sécurité pour cette primitive et soulevons de nombreuses questions concernant différentes propriétés de cette dernière. Enfin, nous explorons la construction de schémas de chiffrement adaptés au partage de données en utilisant les deux primitives précédentes. Nous proposons une construction générique de schéma de chiffrement de groupe (y compris le chiffrement de groupe "augmenté") à partir de schémas de chiffrement basés sur l'identité avec caractère générique. Grâce à nos instantiations de la primitive, nous obtenons un nouveau schéma de chiffrement de groupe qui améliore l'état de l'art en offrant une sécurité adaptative plutôt que simplement sélective, tout en préservant l'efficacité des meilleurs schémas grâce à une taille de chiffré constante. Pour les schémas de chiffrement de groupe "augmentés", la combinaison d'une de nos instantiations de schémas de chiffrement basés sur l'identité avec caractère générique et notre construction générique nous permet d'obtenir un nouveau schéma, le premier à garantir une sécurité adaptative dans le modèle standard. Malheureusement, en termes d'efficacité, notre schéma n'est pas plus efficace qu'une solution "triviale". Cependant, grâce à nos constructions génériques, une amélioration de la primitive sous-jacente contribuera à l'amélioration des schémas de chiffrement de groupe "augmentés". Nous proposons également un schéma de chiffrement basé sur les attributs en utilisant notre nouveau type d'accumulateurs. Ce schéma est le premier à offrir une taille constante pour la clé secrète et le chiffré, indépendamment du nombre d'attributs dans le schéma, tout en garantissant

une sécurité adaptative. Cependant, cette efficacité est obtenue au détriment de la taille exponentielle de la clé publique, et notre construction, reposant sur des spécificités propres à l'instantiation de notre nouvel accumulateur avec des couplages, ne peut pas être généralisée. Enfin, à travers un cas d'usage concret, nous proposons une nouvelle approche du contrôle d'accès grâce aux schémas de chiffrement basés sur l'identité avec caractère générique.

Mots clés : cryptographie, chiffrement par attributs, chiffrement de groupe, chiffrement basé sur l'identité avec caractère générique, accumulateurs cryptographiques

Advanced Encryption for the Sharing of Sensitive Data

Abstract: Data, including private information, plays a pivotal role in our daily lives today. Current research predominantly focuses on data storage, with an emphasis on the ability to securely process data even when it is encrypted. However, beyond mere preservation, data must also be shared in various ways: either among an individual and a group of individuals, sometimes bound by common characteristics defining sharing rules, or simply between two individuals. Currently, these different modes of sharing are not yet well-mastered, either due to their high performance cost or limited functionalities. This thesis delves into various encryption schemes tailored for sharing sensitive data, proposing new constructions. Firstly, we investigate two cryptographic primitives: identity-based encryption schemes with wildcards and cryptographic accumulators, which serve as a starting point for our new constructions. Regarding identity-based encryption schemes with wildcards, we introduce a new security property and propose two new instantiations, one of which satisfies this new security property that we have defined. For cryptographic accumulators, we present a new type of accumulator, an improved scheme compared to the state of the art, and a second scheme illustrating our new functionality. We also introduce a new security property for this primitive and raise numerous questions concerning various properties of the latter. Finally, we explore the construction of encryption schemes suited for data sharing using the two aforementioned primitives. We propose a generic construction of a group encryption scheme (including "augmented" group encryption) based on identity-based encryption schemes with wildcards. With our instantiations of the primitive, we achieve a new group encryption scheme that enhances the state of the art by offering adaptive security rather than just selective, while preserving the efficiency of the best schemes due to a constant ciphertext size. For "augmented" group encryption schemes, the combination of one of our instantiations of identity-based encryption schemes with wildcards and our generic construction enables us to obtain a new scheme, the first to guarantee adaptive security in the standard model. Unfortunately, in terms of efficiency, our scheme is no more efficient than a "trivial" solution. However, thanks to our generic constructions, an enhancement of the underlying primitive

will contribute to improving “augmented” group encryption schemes. We also propose an attribute-based encryption scheme using our new type of accumulators. This scheme is the first to offer a constant size for the secret key and ciphertext, regardless of the number of attributes in the scheme, while guaranteeing adaptive security. However, this efficiency comes at the cost of an exponential size for the public key, and our construction, relying on specific features of our new accumulator instantiation with pairings, cannot be generalized. Finally, through a concrete use case, we introduce a novel approach to access control using identity-based encryption schemes with wildcards.

Keywords: cryptography, attribute-based encryption, (augmented) broadcast encryption, identity-based encryption with wildcard, cryptographic accumulators

Univ. de Limoges, CNRS, XLIM, UMR 7252, F-87000 Limoges, France