Anonymity and Time in Public-Key Encryption

Elizabeth Anne Quaglia

Thesis submitted to the University of London
for the degree of Doctor of Philosophy

Information Security Group
Department of Mathematics
Royal Holloway, University of London

2012

Declaration

These doctoral studies were conducted under the supervision of Prof. Kenneth G. Paterson.

The work presented in this thesis is the result of original research carried out by myself, in collaboration with others, whilst enrolled in the Department of Mathematics as a candidate for the degree of Doctor of Philosophy. This work has not been submitted for any other degree or award in any other university or educational establishment.

Elizabeth Anne Quaglia
April, 2012

Acknowledgements

I would like to thank Prof. Kenny Paterson for the unceasing support he provided during my PhD and for his insightful guidance through the fascinating world of cryptography. I truly admire the energy and enthusiasm with which Kenny engages in his research and his role as a supervisor, and I consider myself lucky to have been his student. I am extremely grateful to him for believing in me and giving me this great opportunity.

I am grateful to the Mathematics department at Royal Holloway which not only helped me financially but also has always been a warm and pleasant environment to work in. In particular, thanks to Dr. Christian Elsholtz, without whom none of this would have happened, and to Prof. Keith Martin, Prof. Jason Crampton and Dr. Carlos Cid for being kind, friendly and always up for an interesting discussion.

I would also like to thank Benoît Libert and Pooya Farshim for our fruitful collaborations.

The years of my PhD have been an extraordinary adventure, coloured by invaluable experiences and wonderful people.

I would like to thank the Crypto group at IBM (New York) and the Product Security team at Qualcomm (San Diego) for the productive and enjoyable time I spent as an intern. These opportunities have been key to my professional and personal growth. In particular, my warmest thanks to Hugo Krawczyk, for sharing his contagious passion for research, and all its possible amazing applications, his incredible generosity and kindness.

Looking back at these years I will remember with affection the people I have met along the way. Starting from my office colleagues and friends Qin and Wei, my Chinese family, Jean-Paul, Ciaran, Shahram and Michelle. Thank you for your wisdom. Gary, Charles and Pooya, thank you for the inspiring conversations we shared. Special thanks to the Royal Holloway girls, Anastasia and Susan, whose support, especially in the writing-up period, has been invaluable.

A te, Euge, che mi regali sempre un sorriso.

To Vale, Chiara and Kate, for your unconditional friendship and love.

To Gaven and Georg, for the *totally awesome* team moments we shared.
Gaven, thank you for being like a brother to me.
Georg, thank you for your unwavering support, your infinite patience and care, and for being such a wonderful part of this journey.

I would like to express immense gratitude to Peter and Irene, thanks to whom I have always felt at home and who made sure not too much time passed between visits to Nan's favourite curry house.

My final thanks go to my beautiful and loving family: Mum, Papá and Vic, you are my strength and this thesis is dedicated to you.

Thanks to all for these **P**retty **h**appy **D**ays.

Abstract

In a world that is increasingly relying on digital technologies, the ability to securely communicate and distribute information is of crucial importance. Cryptography plays a key role in this context and the research presented in this thesis focuses on developing cryptographic primitives whose properties address more closely the needs of users.

We start by considering the notion of *robustness* in public-key encryption, a property which models the idea that a ciphertext should not decrypt to a valid message under two different keys. In contexts where anonymity is relevant, robustness is likely to be needed as well, since a user cannot tell from the ciphertext if it is intended for him or not. We develop and study new notions of robustness, relating them to one another and showing how to achieve them.

We then consider the important issue of protecting users' privacy in broadcast encryption. Broadcast encryption (BE) is a cryptographic primitive designed to efficiently broadcast an encrypted message to a target set of users that can decrypt it. Its extensive real-life application to radio, television and web-casting renders BE an extremely interesting area. However, all the work so far has striven for efficiency, focusing in particular on solutions which achieve short ciphertexts, while very little attention has been given to anonymity. To address this issue, we formally define *anonymous broadcast encryption*, which guarantees recipient-anonymity, and we provide generic constructions to achieve it from public-key, identity-based and attribute-based encryption. Furthermore, we present techniques to improve the efficiency of our constructions.

Finally, we develop a new primitive, called *time-specific encryption* (TSE), which allows us to include the important element of time in the encryption and decryption processes. In TSE, the sender is able to specify during what time interval a ciphertext can be decrypted by a receiver. This is a relevant property since information may become useless after a certain point, sensitive data may not be released before a particular time, or we may wish to enable access to information for only a limited period. We define security models for various flavours of TSE and provide efficient instantiations for all of them.

These results represent our efforts in developing public-key encryption schemes with enhanced properties, whilst maintaining the delicate balance between security and efficiency.

Contents

Chapter 1

Introduction

Contents

1.1 A delicate balance

Claude Shannon's famous paper "*Communication Theory of Secrecy Systems*" [83] can be regarded as the pivotal work which transformed cryptography from an almost recreational brain-teasing activity to the official *science of secrets* [88]. It was in [83], in fact, that the bases of the theory of cryptography were set and that a first rigorous mathematical proof of the perfect secrecy of the Vernam cipher (also known as the one-time pad) was given.

One of the original goals of cryptography can be considered that of enabling secure communication. Therefore, having designed a perfectly secure cipher could seem to have already solved this problem. On the other hand, we know that research in this area has rapidly evolved since this result and it is still progressing at a very high pace. So, what is the catch?

It is well-known that, however secure the one-time pad is, it suffers from severe practical limitations. Indeed, the secret key to encrypt can be used only once and it has to be as long as the message, raising the fundamental real-world issue of key-management. The realization of the need of practicality in the design of cryptosystems characterizes the advances of the cryptographic research community, which has tried to find the delicate balance between security and efficiency ever since.

One may ask why efficiency is actually needed. Why isn't the mere intellectual satisfaction of achieving a perfectly secure way of communicating enough? A simple answer to these questions is because cryptography *is used*. In a world that is increasingly relying on digital technologies cryptography has become part of our daily life. Mobile phone communications, credit card payments, web browsing are only a few examples of cryptographically enabled operations that are widely used today. One does not have to look too far to see many more: e-voting, e-commerce, on-line banking, distribution of digital copyright media are further examples of this.

Given the widespread use of cryptography in a variety of applications, it is fairly natural that the related research has focused on developing systems capable of providing the required security guarantees whilst maintaining acceptable levels of practicality for real-life deployment. One could regard this quest as being the *leitmotif* behind the progress of cryptography through the years. In this respect, we view the results in this thesis as making a contribution to the theme of designing cryptographic schemes delicately balanced between security and efficiency.

1.2 Anonymity and time in public-key encryption

The title of this thesis, "*Anonymity and time in public-key encryption*", identifies the core features of our work. We are going to design and construct several *encryption* schemes in the *public-key* setting satisfying enhanced security properties. In particular, we consider the important property of *anonymity*, the related notion of robustness, and the ability to include the element of *time* in the encryption process.

Public-key encryption. Since its revolutionary introduction in 1976 [42], public-key cryptography has developed greatly, and by now it represents a fundamental area of cryptography. In particular, public-key encryption (PKE) has received a lot of research attention. Indeed the cryptographic community has obtained numerous results, by achieving various levels of security in the proposed schemes, by continuously improving on the efficiency of the constructions, and by developing a variety of additional functionalities. We recall the basic notions for PKE relevant to our work in Chapter 2. However, since an exhaustive and detailed presentation of PKE is beyond the scope of this thesis, we refer the interested reader to [57].

The importance of anonymity. Addressing the issue of protecting users' privacy is of crucial importance. This is reflected by the great attention given to *anonymity* in all the main fields of modern cryptography. In the area of PKE, anonymity is often referred to as key-privacy [8]. This notion captures the property that an eavesdropper is not able to tell under which one of several public keys a ciphertext was created. The analogous concept in the identity-based setting was studied in [1, 23]. The benefit of preserving receivers' privacy is relevant in more elaborate systems involving for example hierarchical identity-based encryption (HIBE) [22], attribute-based encryption (ABE) or predicate encryption [58], where achieving anonymity guarantees becomes increasingly challenging. Furthermore, in the context of digital signatures, a number of primitives effectively *rely* on anonymity – group signatures [30], anonymous credentials [29] and e-cash [28] are well-known examples of this. In our work, we consider anonymity in the context of public-key broadcast encryption, a primitive designed to address a dynamically changing set of receivers for the secure distribution of digital data.

The importance of time. As the amount of transmitted and stored digital content rapidly increases, concerns naturally arise regarding the accessibility of such data. In this context, the dimension of *time* has become significantly relevant. Indeed, security research aims to address not only the issue of who can access the content, but also when and for how long. In the cryptographic literature, the element of time has appeared in several contexts, as the following examples will illustrate. In the IBE setting [16], for instance, it was suggested to extend the identity of a user so as to include a decryption time, allowing therefore to encrypt to the future. A whole branch of research is dedicated to the study and development of timed-release encryption [65, 27, 26], a primitive which precisely allows a sender to specify a release-time for the encrypted message, before which encryption is not possible. Some effort has also been put in developing ways to make the data unavailable after it has passed its expiry date. The *Ephemeral* line of research, initiated by [74], and the *Vanish* system [49] are examples of this. Our work on time-specific encryption offers a cryptographic solution to access data in a specific time interval, and its efficient realisation makes it suitable for many practical applications.

1.3 Organization of the thesis

In this thesis we give a series of results in the context of public-key encryption.

We start by fixing the notation and giving some basic definitions and security models, key to the development of our work, in **Chapter 2**.

Our contributions are then presented in the chapters that follow. More specifically, in **Chapter 3** we propose new notions of *robustness*, which (informally) is the security property that deals with the issue of using the wrong private key for decryption. We justify the need for such notions and provide generic ways to achieve the strongest robustness notion we introduce. The work presented in this chapter appears in [45].

In **Chapter 4** we consider the fundamental problem of *anonymity* in the context of *broadcast encryption*. After giving formal definitions and security models for the corresponding primitive, we provide constructions to achieve it securely. These results will appear in the proceedings of the international cryptographic conference *Public-Key Cryptography 2012* as part of [62].

Finally, in **Chapter 5**, we introduce a new primitive, called *time-specific encryption*, which allows the sender to express a time interval during the encryption process, specifying the period a ciphertext can be decrypted. We present several flavours of this primitive and show how to generically and securely achieve schemes in the different settings. This work was published as [71] at the conference *Security and Cryptography for Networks 2010* and received the *Best Paper Award*.

The topics presented in this thesis are connected by the dynamics of our research, namely the study of one problem led to the consideration of another and so on. For instance, we proposed the notion of time-specific encryption (TSE) to address a practical issue arising from a shortcoming of timed-release encryption, a closely related primitive. While developing ways to achieve TSE we realized that broadcast encryption (BE) could be used for this purpose. This brought us to the study of the relevant literature, through which we discovered the limitations of the current BE security models and schemes. Our work on anonymous broadcast encryption

developed naturally from here. Since in an anonymous setting ciphertexts do not reveal the intended recipients, we had to address the issue of receivers using their private key on possibly the wrong ciphertext. Precisely this was our motivation for looking at robustness, which eventually resulted in the development of new, stronger notions, for which we provide provably secure constructions.

This brief illustration of our research flow has hopefully made the connecting line between our results more visible. In any case, for each topic, we will give (in the relevant chapter) a detailed introduction, inclusive of motivation, related work and a more in-depth description of our contributions.

Chapter 2

Preliminaries

Contents

In this chapter we introduce the basic notation that will be adopted throughout this thesis, as well as some of the fundamental concepts of provable security which have been used in our work. We then provide the formal definitions and security models for the cryptographic primitives relevant to the following chapters.

2.1 Notation

We introduce some basic notation that will be adopted in this and the following chapters.

- By ":=" we denote a definition, with the definiens on the right-hand side (RHS) and the definiendum on the left-hand side (LHS).

2.1 Notation

- $\mathbb{G}$ denotes an algebraic group, $\mathbb{Z}$ denotes the integers, $\mathbb{N}$ the non-negative integers and $\mathbb{R}$ the reals.

- p denotes a prime, $\mathbb{Z}_p$ the cyclic group of order p and $\mathbb{Z}_p^* := \mathbb{Z}_p \setminus \{0\}$, i.e. the group without the neutral element.

- MsgSp, CtSp, IdSp, SSp, KSp, VSp are the spaces for messages, ciphertexts, identities, signatures, keys and values, respectively.

- $\mathcal{O}$ denotes an oracle and $\mathcal{O}^{sk}$ denotes an oracle equipped with sk. The behaviour of an oracle will be made explicit in the relevant definitions.

- By "$\leftarrow$" we denote a random assignment where the RHS is either a finite set or a probabilistic algorithm. In the former case, it denotes the assignment to the LHS of a random value chosen uniformly from the set; in the latter it denotes choosing the algorithm's random tape uniformly and assigning the outcome to the LHS.

- If s and t are two bit strings, $s||t$ denotes the concatenation of s and t, and $s \oplus t$ denotes their exclusive or.

- $\{0,1\}^n$ denotes all bit strings of length n.

- Algorithms are assumed to be polynomial time (p.t.) or probabilistic polynomial time (p.p.t.). If algorithm Alg is run on inputs x and y, we denote it by $\mathsf{Alg}(x, y)$, and if r is the internal randomness we may make it explicit by writing $\mathsf{Alg}(x, y; r)$.

- $\lambda \in \mathbb{N}$ denotes a security parameter. By convention, the running time of an algorithm is measured as a function of the length of its input, and therefore λ will be provided to the algorithm in *unary* as 1^λ (i.e. the string of λ ones).

- A function $f : \mathbb{N} \to \mathbb{R}$ is called *negligible* if for all $c \in \mathbb{N}$ there exists $k_0 \in \mathbb{N}$ such that for all $k > k_0 : |f(k)| < \frac{1}{k^c}$.

- The symbol $\perp$ denotes a distinguished output of an algorithm intended to indicate an error.

2.2 Provable security

The aim of this thesis is to provide constructions for public-key encryption primitives with enhanced properties, maintaining a balance between security and efficiency whilst achieving the desired functionality. In order to do so, we make extensive use of the tools and techniques of *provable security*, which provides a rigorous mathematical framework for the security analysis and proofs of our schemes.
In presenting the results in this thesis our approach will typically be the following.

- As a first step, we will give a formal **definition** of the cryptographic primitive we wish to study. This is essentially the description of the algorithms (with their inputs and outputs) that constitute the primitive.
- The next step is to define an **adversarial model** which formally specifies what a *computationally* bounded adversary is allowed to do when perpetrating an attack. Such a model should capture the idea of what it means for a primitive to be *secure* with respect to a specific security goal (e.g confidentiality, anonymity, ...). The security models we present will be described as *games* between a benign entity called the challenger and an adversary, which may or may not have access to a set of *oracles* controlled by the challenger. We specify a winning condition for the game and we typically define the adversary's *advantage* as a measure of the success of its strategy over that of simply guessing.

  There are two common approaches when studying computational security: the concrete one and the asymptotic one ([57, Chapter 3] provides an interesting discussion on the topic). The asymptotic approach is the one we will follow in this thesis. Here, the running time of the adversary as well as its success probability are *functions* of the security parameter λ, as opposed to *concrete* numbers. In particular, an adversary is a probabilistic algorithm whose running time is polynomial in λ and whose success probability we would like to be negligible in λ. This leads to the following informal definition ([57]):

  A scheme is secure *if for every p.p.t. adversary $\mathcal{A}$ playing a game of some specified type, the advantage of $\mathcal{A}$ in winning the game (where the winning condition is also well-defined) is negligible.*

- The following step is to achieve the specified primitive with the desired security properties. To this goal, we present possible **constructions** for it (typically using simpler cryptographic primitives as building blocks), and prove that such constructions yield secure schemes.
- To prove a scheme is secure with respect to a specific security model, we reduce its security to that of the underlying cryptographic primitive or to the hard problem it is based on. More specifically, providing a **proof of security** (also called a *reduction*) involves converting any efficient adversary that has a non-negligible advantage in winning the specific game into another efficient algorithm that succeeds in breaking the underlying primitive or in solving a hard problem. This is nowadays a standard proof technique in cryptography.

It is within this framework that the research presented in this thesis has been conducted.

2.3 Formal definitions of cryptographic primitives

Cryptographic primitives are the basic building blocks used to construct more complex cryptographic systems. Their algorithms are designed to achieve a variety of functionalities and they are carefully crafted so as to satisfy the required security properties. While there are many such primitives, the focus of this thesis will be on primitives in the *public-key* setting. We recall the ones relevant to our work next.

2.3.1 Public-key encryption

In a public-key encryption (PKE) scheme anyone can encrypt a message with respect to a public key, but only the holder of the corresponding secret key can recover it. The two keys, i.e the *public* encryption key and the *secret* decryption key, have to be mathematically related and it should be hard to obtain the secret key simply by knowing the public key.

We present a definition for PKE. This slightly differs from the usual notation by

having a parameter generation algorithm PKE.PG as an additional algorithm. This will ease the introduction of future notions. We can recover the standard definition simply by letting PKE.PG output the security parameter.

Definition 2.1 (PKE scheme) *A* public-key encryption (PKE) scheme *is defined by four algorithms* (*PKE.PG*, *PKE.KeyGen*, *PKE.Enc*, *PKE.Dec*)*, which are as follows.*

PKE.PG: This algorithm takes as input the security parameter 1^λ and returns the public parameters $pars$. These will include a description of the message space MsgSp and the ciphertext space CtSp of the scheme. We write this as $pars \leftarrow \mathsf{PKE.PG}(1^\lambda)$.

PKE.KeyGen: This is a key generation algorithm that on input $pars$ outputs a public key pk and its corresponding secret key sk. We write this as $(pk, sk) \leftarrow \mathsf{PKE.KeyGen}(pars)$.

PKE.Enc: This is an encryption algorithm that on input $pars$, a message $M \in \mathsf{MsgSp}$ and a public key pk returns a ciphertext $C \in \mathsf{CtSp}$. We write this as $C \leftarrow \mathsf{PKE.Enc}(pars, M, pk)$.

PKE.Dec: This is a decryption algorithm that on input public parameters $pars$, a public key pk, a ciphertext C and a secret key sk returns either a message or the special symbol $\perp$ denoting failure. We write this as $\mathsf{PKE.Dec}(pars, pk, C, sk) = M$, where $M \in \mathsf{MsgSp} \cup \{\perp\}$.

These algorithms are required to satisfy the following correctness property*: For every λ, for every set of parameters $pars$ output by PKE.PG, for every message $M \in$ MsgSp and every key-pair (pk, sk) generated by PKE.KeyGen, if $C \leftarrow$ PKE.Enc$(pars, M, pk)$ then PKE.Dec$(pars, pk, C, sk) = M$.*

For simplicity, we will often omit the public parameters as input to the encryption and decryption algorithm and assume they are implicit. The same holds for the public key in the decryption algorithm.

As mentioned in the previous section, we will model security for a cryptographic primitive in terms of a game between a challenger and an adversary. In particu-

lar, we define the security notion of *indistinguishability under chosen-ciphertext attacks* (IND-CCA) [52, 77] for a PKE scheme $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ as follows.

IND-CCA security game for PKE

Setup. The challenger $\mathcal{C}$ runs $\mathsf{PKE.PG}(1^\lambda)$ to generate $pars$ and $\mathsf{PKE.KeyGen}(pars)$ to obtain a key-pair (pk, sk). $\mathcal{C}$ gives $(pars, pk)$ to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ has access to a decryption oracle $\mathcal{O}^{sk}$, to which it submits queries of the type C. The oracle returns $\mathsf{PKE.Dec}(pars, pk, C, sk)$.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0, $M_1 \in \mathsf{MsgSp}$ and passes them to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0, 1\}$ and computes $C^\star \leftarrow \mathsf{PKE.Enc}(pars, M_b, pk)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to have access to a decryption oracle $\mathcal{O}^{sk}$, with the restriction that it cannot submit the query $C^\star$ to this oracle.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage in the above game as $\mathbf{Adv}^{\mathrm{IND-CCA}}_{\mathcal{A},\Pi}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$. Intuitively, the advantage is a measure of how successful the adversary's strategy in the game is over that of simply guessing the bit b.

Definition 2.2 (IND-CCA) *A PKE scheme* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is* indistinguishable under chosen-ciphertext attacks *(or is* IND-CCA secure*) if all p.p.t. adversaries have at most negligible advantage in the above game.*

It is possible to define a weaker notion of security, namely indistinguishability under chosen-plaintext attacks (IND-CPA), by removing the adversary's access to the decryption oracle $\mathcal{O}^{sk}$ in the game described above.

Since it will be relevant in this thesis, we recall a security notion for public-key encryption which models both indistinguishability and anonymity. In the public-key setting, the latter is often referred to as *key-privacy*, and was introduced by Bellare et al. in [8]. Informally, a PKE scheme is key-private if the ciphertext does not leak under which public key it was created. The authors of [8] give two notions of security of indistinguishability of keys (IK), IK-CPA and IK-CCA, which model key-privacy under chosen-plaintext and chosen-ciphertext attacks, respectively. We will use a combined notion of security, which helps streamline our presentation and proofs. The relevant game and security notion are as follows.

IND-IK-CCA security game for PKE

Setup. The challenger $\mathcal{C}$ runs $\mathsf{PKE.PG}(1^\lambda)$ to generate $pars$ and $\mathsf{PKE.KeyGen}(pars)$ twice to obtain two key-pairs (pk_0, sk_0) and (pk_1, sk_1). $\mathcal{C}$ gives $(pars, pk_0, pk_1)$ to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ has access to a decryption oracle $\mathcal{O}^{sk_0,sk_1}$, to which it submits queries of the type (C, pk_i), $i \in \{0,1\}$. More specifically, the oracle returns $\mathsf{PKE.Dec}(pars, pk_i, C, sk_i)$, $i \in \{0,1\}$.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0, $M_1 \in \mathsf{MsgSp}$ and passes them to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0,1\}$ and computes $C^\star \leftarrow \mathsf{PKE.Enc}(pars, M_b, pk_b)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to have access to the decryption oracle, with the restriction that it cannot submit queries containing $C^\star$ to the oracle.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage in the above game as $\mathbf{Adv}^{\mathrm{IND-IK-CCA}}_{\mathcal{A},\Pi}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 2.3 (IND-IK-CCA) *A PKE scheme* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is* indistinguishable and key-private under chosen-ciphertext attacks *(or is* IND-IK-CCA secure*) if all p.p.t. adversaries have at most negligible advantage in the above game.*

Also here we can define a weaker notion of security, namely IND-IK-CPA, by removing the adversary's access to the decryption oracles.

We note that for simplicity, and in accordance with the majority of the relevant literature, we decided to model indistinguishability and anonymity by letting the challenger randomly pick *one* bit, used to select both the message and the key. An alternative and equivalent approach would be to let the challenger select two bits, one for the message and one for the key. All primitives for which we model indistinguishability and anonymity in this thesis will be analyzed and proved secure in the 1-bit setting.

Examples of IND-IK-ATK secure schemes, where ATK $\in \{\text{CPA}, \text{CCA}\}$, are the ElGamal [48] and the Cramer–Shoup [37] encryption schemes. Furthermore, in [73], the authors present a *generic* way to achieve an IND-IK-CCA secure PKE scheme from identity-based encryption, a primitive which we introduce next.

2.3.2 Identity-based encryption

Identity-based encryption (IBE) was first proposed by Shamir in 1984 [82], but was only realized in 2000 by Sakai, Ohgishi and Kasahara [80], and in 2001 in the works of Boneh and Franklin [16] and Cocks [34]. The key idea in IBE is that the public key of a user can be an arbitrary string, typically representing his identity (e.g. the user's e-mail address), rather than a string that is output by a key generation algorithm as in normal PKE. The main motivation behind the introduction of IBE is that it simplifies the problem of certificate management, distinctive to public-key cryptography. This however requires a Trusted Authority (TA) to issue the users' corresponding secret keys. The need for such an authority is an inherent feature of the identity-based setting.

We next formalize the description of the IBE primitive, introducing an additional parameter generation algorithm, IBE.PG, for ease of exposition. As in the PKE case, the standard notion is recovered simply by letting IBE.PG output the security parameter.

Definition 2.4 (IBE scheme) *An* identity-based encryption (IBE) scheme *is defined by five algorithms, which are as follows.*

IBE.PG: This algorithm takes as input the security parameter 1^λ and returns the system's parameters $pars$. These include a description of the message space MsgSp, the ciphertext space CtSp and the identity space IdSp of the scheme. We write this as $pars \leftarrow \mathsf{IBE.PG}(1^\lambda)$.

IBE.Setup: This algorithm takes as input $pars$ and returns a master public key ID-MPK and a master secret key ID-MSK. We write $(\text{ID-MPK}, \text{ID-MSK}) \leftarrow \mathsf{IBE.Setup}(pars)$.

IBE.KeyExt: This is a key extraction algorithm that on input ID-MPK, ID-MSK and an identity $id \in \mathsf{IdSp}$ outputs a secret key sk_{id}. We write this as $sk_{id} \leftarrow \mathsf{IBE.KeyExt}(\text{ID-MPK}, \text{ID-MSK}, id)$.

IBE.Enc: This is an encryption algorithm that on input ID-MPK, a message $M \in \mathsf{MsgSp}$ and an identity $id \in \mathsf{IdSp}$ returns a ciphertext $C \in \mathsf{CtSp}$. We write this as $C \leftarrow \mathsf{IBE.Enc}(\text{ID-MPK}, M, id)$.

IBE.Dec: This is a decryption algorithm that on input ID-MPK, a ciphertext C and a secret key sk_{id} returns either a message or a failure symbol $\perp$. We write this as $\mathsf{IBE.Dec}(\text{ID-MPK}, C, sk_{id}) = M$, where $M \in \mathsf{MsgSp} \cup \{\perp\}$.

These algorithms are required to satisfy the following correctness property*: For every λ, for any pars output by IBE.PG, for every ID-MPK, ID-MSK output by IBE.Setup, for every message $M \in$ MsgSp and every identity $id \in$ IdSp, if $sk_{id} \leftarrow$ IBE.KeyExt$(ID\text{-}MPK, ID\text{-}MSK, id)$ and if $C \leftarrow$ IBE.Enc$(ID\text{-}MPK, M, id)$ then IBE.Dec$(ID\text{-}MPK, C, sk_{id}) = M$.*

We define the security notion of *id-based indistinguishability under chosen-ciphertext attacks* (IND-ID-CCA) [16] for an IBE scheme $I = (\mathsf{IBE.PG}, \mathsf{IBE.Setup}, \mathsf{IBE.KeyExt}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ as follows.

IND-ID-CCA security game for IBE

Setup. The challenger $\mathcal{C}$ runs $\mathsf{IBE.PG}(1^\lambda)$ to generate $pars$ and $\mathsf{IBE.Setup}(pars)$ to obtain the master public key ID-MPK and the master secret key ID-MSK and gives $(pars, \text{ID-MPK})$ to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ has access to a secret-key-extraction oracle $\mathcal{O}^{\text{ID-MSK}}$, to obtain secret keys of any $id \in \mathsf{IdSp}$. $\mathcal{A}$ has also access to a decryption oracle $\mathcal{O}^{\text{ID-MSK}}$, to which it submits queries of the type (C, id). This oracle returns $\mathsf{IBE.Dec}(\text{ID-MPK}, C, sk_{id})$, where sk_{id} is extracted using ID-MSK.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and an $id^\star \in \mathsf{IdSp}$ with the restriction that for none of the secret-key-extraction queries in Phase 1 we have that $id = id^\star$. $\mathcal{A}$ passes $M_0, M_1, id^\star$ to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0,1\}$ and computes $C^\star \leftarrow \mathsf{IBE.Enc}(\text{ID-MPK}, M_b, id^\star)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to have access to a secret-key-extraction oracle $\mathcal{O}^{\text{ID-MSK}}$, with the same restriction we have in the Challenge phase, and to a decryption oracle $\mathcal{O}^{\text{ID-MSK}}$, with the restriction that it cannot submit the query $(C^\star, id^\star)$ to this oracle.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage in the above game as $\mathbf{Adv}^{\text{IND-ID-CCA}}_{\mathcal{A},I}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 2.5 (IND-ID-CCA) *An IBE scheme* $I = (\mathsf{IBE.PG}, \mathsf{IBE.Setup}, \mathsf{IBE.KeyExt}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ *is* indistinguishable under chosen-ciphertext attacks *(or is* IND-ID-CCA secure*) if all p.p.t. adversaries have at most negligible advantage in the above game.*

We define the notion of IND-ID-CPA security (indistinguishability under chosen-plaintext attacks) by removing the adversary's access to the decryption oracle in the game described above.

In our work we will make use of a weaker security notion for IBE called *selective-id* security, introduced in [24, 13], where an adversary has to output a challenge

identity $id^\star$ at the beginning of the game. The identity $id^\star$ will be then used by the challenger as the identity to which encrypt M_b. We give details of this game next.

sID-IND-CCA security game for IBE

Initialize. The challenger $\mathcal{C}$ runs $\mathsf{IBE.PG}(1^\lambda)$ to generate $pars$ and gives them to the adversary $\mathcal{A}$. $\mathcal{A}$ outputs $id^\star \in \mathsf{IdSp}$.

Setup. $\mathcal{C}$ runs $\mathsf{IBE.Setup}(pars)$ to obtain the master public key ID-MPK and the master secret key ID-MSK and gives ID-MPK to $\mathcal{A}$.

Phase 1. $\mathcal{A}$ has access to a secret-key-extraction oracle $\mathcal{O}^{\text{ID-MSK}}$, to obtain secret keys of any $id \neq id^\star \in \mathsf{IdSp}$. $\mathcal{A}$ has also access to a decryption oracle $\mathcal{O}^{\text{ID-MSK}}$, to which it submits queries of the type (C, id).

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$. $\mathcal{A}$ passes M_0, M_1 to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0,1\}$ and computes $C^\star \leftarrow \mathsf{IBE.Enc}(\text{ID-MPK}, M_b, id^\star)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to have access to a secret-key-extraction oracle $\mathcal{O}^{\text{ID-MSK}}$, with the same restriction we have in Phase 1, and to a decryption oracle $\mathcal{O}^{\text{ID-MSK}}$, with the restriction that it cannot submit the query $(C^\star, id^\star)$ to this oracle.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage as $\mathbf{Adv}^{\text{sID}-\text{IND}-\text{CCA}}_{\mathcal{A},I}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 2.6 (sID-IND-CCA) *An IBE scheme* $I = (\mathsf{IBE.PG}, \mathsf{IBE.Setup}, \mathsf{IBE.KeyExt}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ *is* selective-id indistinguishable under chosen-ciphertext attacks *(or* sID-IND-CCA secure*) if all p.p.t. adversaries have at most negligible advantage in the above game.*

Removing access to the decryption oracle gives rise to the notion of sID-IND-CPA security.

As for the public-key setting, it is useful at this point to give the security notion for *anonymous* IBE [1], which informally models the idea that a ciphertext does not leak the identity of the intended recipient. We present a combined notion of indistinguishability and anonymity for IBE in the following game.

ANO-IND-CCA security game for IBE

Setup. The challenger $\mathcal{C}$ runs $\mathsf{IBE.PG}(1^\lambda)$ to generate $pars$ and $\mathsf{IBE.Setup}(pars)$ to obtain the master public key ID-MPK and the master secret key ID-MSK and gives $(pars, \text{ID-MPK})$ to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ has access to a secret-key-extraction oracle $\mathcal{O}^{\text{ID-MSK}}$, to obtain secret keys of any $id \in \mathsf{IdSp}$. $\mathcal{A}$ has also access to a decryption oracle $\mathcal{O}^{\text{ID-MSK}}$, to which it submits queries of the type (C, id).

Challenge. $\mathcal{A}$ selects two equal-length messages $M_0,\ M_1 \in \mathsf{MsgSp}$ and two identities $id_0,\ id_1 \in \mathsf{IdSp}$ with the restriction that for none of the secret-key-extraction queries in Phase 1 we have that $id = id_0$ or $id = id_1$. $\mathcal{A}$ passes M_0, M_1, id_0, id_1 to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0,1\}$ and computes $C^\star \leftarrow \mathsf{IBE.Enc}(\text{ID-MPK}, M_b, id_b)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to have access to a secret-key-extraction oracle $\mathcal{O}^{\text{ID-MSK}}$, with the same restriction we have in the Challenge phase, and to a decryption oracle $\mathcal{O}^{\text{ID-MSK}}$, with the restriction that it cannot submit the queries $(C^\star, id_0)$ or $(C^\star, id_1)$, to this oracle.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage as $\mathbf{Adv}^{\text{ANO}-\text{IND}-\text{CCA}}_{\mathcal{A},I}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 2.7 (ANO-IND-CCA) *An IBE scheme* $I = (\mathsf{IBE.PG}, \mathsf{IBE.Setup}, \mathsf{IBE.KeyExt}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ *is* anonymous and indistinguishable under chosen-ciphertext attacks *(or* ANO-IND-CCA secure*) if all p.p.t. adversaries have at most negligible advantage in the above game.*

As usual, removing access to the decryption oracle defines the corresponding notion of security in a chosen-plaintext attack scenario.

Since it will be relevant to our work, we briefly recall the notion of **multi-TA IBE** [72], where multiple and independent trusted authorities coexist in the system. A typical multi-TA IBE scheme consists of five algorithms: $\mathsf{IBE.PG}$, which takes as input the security parameter and outputs the shared parameters $pars$ and a set of labels of the TAs in the system; $\mathsf{IBE.TASetup}$, which takes as input $pars$ and outputs a master public key and a master secret key (this is run independently for each TA in the system); $\mathsf{IBE.KeyExt}$, $\mathsf{IBE.Enc}$, $\mathsf{IBE.Dec}$ as for a normal IBE scheme. Security notions such as indistinguishability and recipient anonymity have been considered in this setting in [72], where in particular the notion of multi-TA *anonymity* was introduced and studied. Informally, this models the idea that a ciphertext does not leak the master public key under which it was created. It should therefore be hard for an adversary to distinguish between two ciphertexts generated under two distinct master public keys, even if for the same message and the same identity. In [72] several security models for multi-TA IBE were put forth. We recall the one relevant for our work next.

sID-TAA-IND-CPA security game for multi-authority IBE

Initialize. The challenger $\mathcal{C}$ runs $\mathsf{IBE.PG}(1^\lambda)$ to generate $pars$ and gives them to the adversary $\mathcal{A}$. $\mathcal{A}$ outputs $id^\star \in \mathsf{IdSp}$.

Setup. $\mathcal{C}$ runs $\mathsf{IBE.TASetup}(pars)$ to obtain master public keys MPK_i and master secret keys MSK_i where $i \in \{1, ..., n\}$ for all TAs in the system. $\mathcal{C}$ gives $\{\mathrm{MPK}_i\}_{i\in\{1,...,n\}}$ to $\mathcal{A}$.

Phase 1. $\mathcal{A}$ has access to a corrupt oracle $\mathcal{O}^{\{\mathrm{MSK}_i\}}$ to obtain the master secret key of TA i where $i \in \{1, ..., n\}$. It has also access to a secret-key-extraction oracle to which it can submit queries of the form (i, id), where i is a TA and $id \in \mathsf{IdSp}$, in order to obtain the secret key corresponding to identity id under the trusted authority i.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and two TAs i_0 and i_1, with the restriction that neither i_0 nor i_1 was corrupted and

none of the queries in Phase 1 was of the form $(i_j, id^\star)$, with $j \in \{0,1\}$. $\mathcal{A}$ passes M_0, M_1, i_0, i_1 to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0,1\}$ and computes $C^\star \leftarrow \mathsf{IBE.Enc}(\mathrm{MPK}_{i_b}, M_b, id^\star)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to have access to a corrupt oracle and a secret-key-extraction oracle with the same restrictions we have in the challenge phase.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage as $\mathbf{Adv}^{\mathrm{sID-TAA-IND-CPA}}_{\mathcal{A},I}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 2.8 (sID-TAA-IND-CPA) *A multi-TA IBE scheme* $I = (\mathsf{IBE.PG}, \mathsf{IBE.TASetup}, \mathsf{IBE.KeyExt}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ *is* selective-id, TA-anonymous and indistinguishable under chosen-plaintext attacks *(or* sID-TAA-IND-CPA secure*) if all p.p.t. adversaries have at most negligible advantage in the above game.*

An example of IBE scheme satisfying the notion of sID-TAA-IND-CPA security is the multi-TA version of Gentry's IBE scheme [50], as shown in [73].

We note that by having $i_0 = i_1$ we obtain the standard notion of selective-id security under chosen-plaintext attacks (sID-IND-CPA) for a multi-TA IBE scheme.

2.3.3 Attribute-based encryption

Attribute-based encryption (ABE) is a powerful cryptographic primitive first introduced by Sahai and Waters [79]. The key idea in ABE is that a user can decrypt only if he has the appropriate set of attributes. We are hence intuitively encrypting to a *set* of users, as opposed to *one* as in the standard public-key and identity-based settings. In doing so our main concern is to avoid *collusion attacks*, which would allow distinct users to combine their attributes in order to decrypt something that individually they would not have been able to. ABE historically has two main flavours: key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE). In KP-ABE [53, 70] the attributes are associated with the ciphertext while the policy is

expressed in the key. In CP-ABE [12, 32] the situation is reversed, giving the sender the control over what policy needs to be satisfied in order to decrypt. Typically, such a policy will be expressed in terms of an *access structure*, as defined in [7].

Definition 2.9 (Access structure) *Let $P = \{P_1, P_2, ..., P_n\}$ be a set of parties and let 2^P denote its power set. A collection $\mathbb{A} \subseteq 2^P$ is monotone if for every B and C, if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An* access structure *(respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of P, i.e. $P \setminus \emptyset$. The sets in $\mathbb{A}$ are called the* authorized sets*, and the sets not in $\mathbb{A}$ are called the* unauthorized sets*.*

In our work, as in most relevant literature [53, 12], we will restrict ourselves to *monotone access structures* which will be specified by *access trees*. In this model, each interior node of a tree $\mathcal{T}$ is a threshold gate and the leaves are associated with attributes. If a set of attributes S satisfies the access tree $\mathcal{T}$, we denote it as $\mathcal{T}(S) = 1$.[1] This is defined recursively for $\mathcal{T}_x$, the subtree of $\mathcal{T}$ rooted at node x. In particular, if x is a leaf node, then $\mathcal{T}_x(S)$ returns 1 if and only if the attribute associated to the leaf node x belongs to S.

This is the framework within which we provide the formal definitions and security models for CP-ABE and KP-ABE.

Definition 2.10 (CP-ABE scheme) *A* ciphertext-policy ABE (CP-ABE) scheme *is defined by five algorithms, which are as follows.*

$\mathsf{CP_{ABE}}$.PG: This algorithm takes as input the security parameter 1^λ and returns the system's parameters $pars$. These will include the message space MsgSp, the ciphertext space CtSp and the universe of attributes U associated to the scheme. We write this as $pars \leftarrow \mathsf{CP_{ABE}.PG}(1^\lambda)$.

$\mathsf{CP_{ABE}}$.Setup: This algorithm takes as input $pars$ and returns a master public key CP-MPK and a master secret key CP-MSK. We write $(\text{CP-MPK}, \text{CP-MSK}) \leftarrow \mathsf{CP_{ABE}.Setup}(pars)$.

[1] In general, if S satisfies an access structure $\mathbb{A}$, we denote it as $S \in \mathbb{A}$

$\mathsf{CP_{ABE}}$.KeyGen: This is a key generation algorithm that on input CP-MPK, CP-MSK and a set of attributes $S \in U$ outputs a secret key sk_S. We write this as $sk_S \leftarrow \mathsf{CP_{ABE}.KeyGen}(\text{CP-MPK}, \text{CP-MSK}, S)$.

$\mathsf{CP_{ABE}}$.Enc: This is an encryption algorithm that on input CP-MPK, a message $M \in \mathsf{MsgSp}$ and an access structure $\mathbb{A}$ over the universe of attributes U returns a ciphertext $C \in \mathsf{CtSp}$. We write this as $C \leftarrow \mathsf{CP_{ABE}.Enc}(\text{CP-MPK}, M, \mathbb{A})$.

$\mathsf{CP_{ABE}}$.Dec: This is a decryption algorithm that on input CP-MPK, a ciphertext C and a secret key sk_S returns either a message or a failure symbol $\perp$. We write this as $\mathsf{CP_{ABE}.Dec}(\text{CP-MPK}, C, sk_S) = M$, where $M \in \mathsf{MsgSp} \cup \{\perp\}$.

These algorithms are required to satisfy the following correctness property*: For every λ, for every pars output by $\mathsf{CP_{ABE}.PG}$, for every CP-MPK, CP-MSK output by $\mathsf{CP_{ABE}.Setup}$, for every message $M \in \mathsf{MsgSp}$ and for every access structure $\mathbb{A}$ supported by the system, if $sk_S \leftarrow \mathsf{CP_{ABE}.KeyGen}(\text{CP-MPK}, \text{CP-MSK}, S)$, if $C \leftarrow \mathsf{CP_{ABE}.Enc}(\text{CP-MPK}, M, \mathbb{A})$ and if $S \in \mathbb{A}$, then $\mathsf{CP_{ABE}.Dec}(\text{CP-MPK}, C, sk_S) = M$.*

We define *indistinguishability under chosen-ciphertext attacks* (IND-CCA) for a CP-ABE scheme $\Gamma = (\mathsf{CP_{ABE}.PG}, \mathsf{CP_{ABE}.Setup}, \mathsf{CP_{ABE}.KeyGen}, \mathsf{CP_{ABE}.Enc}, \mathsf{CP_{ABE}.Dec})$ in the following way.

IND-CCA security game for CP-ABE

Setup. The challenger $\mathcal{C}$ runs $\mathsf{CP_{ABE}.PG}(1^\lambda)$ to obtain $pars$ and $\mathsf{CP_{ABE}.Setup}(pars)$ to generate the master public key CP-MPK and the master secret key CP-MSK and gives $(pars, \text{CP-MPK})$ to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ has access to a secret-key-extraction oracle $\mathcal{O}^{\text{CP-MSK}}$, to obtain secret keys for any set of attributes $S \subseteq U$. $\mathcal{A}$ has also access to a decryption oracle $\mathcal{O}^{\text{CP-MSK}}$, to which it submits queries of the type (C, S). Such an oracle will respond with $\mathsf{CP_{ABE}.Dec}(\text{CP-MPK}, C, sk_S)$, where sk_S is extracted using CP-MSK.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and a challenge access structure $\mathbb{A}^\star$ with the restriction that none of the attribute sets

for which a key was queried in Phase 1 satisfy $\mathbb{A}^\star$. $\mathcal{A}$ passes $M_0, M_1, \mathbb{A}^\star$ to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0,1\}$ and computes $C^\star \leftarrow \mathsf{CP_{ABE}.Enc}(\text{CP-MPK}, M_b, \mathbb{A}^\star)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to have access to a secret-key-extraction oracle $\mathcal{O}^{\text{CP-MSK}}$, with the same restriction we have in the Challenge phase, and to a decryption oracle $\mathcal{O}^{\text{CP-MSK}}$, with the restriction that it cannot submit the query $(C^\star, S)$, for any S satisfying $\mathbb{A}^\star$.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage in the above game as $\mathbf{Adv}^{\text{IND}-\text{CCA}}_{\mathcal{A},\Gamma}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 2.11 (IND-CCA) *A CP-ABE scheme* $\Gamma = (\mathsf{CP_{ABE}.PG}, \mathsf{CP_{ABE}.Setup}, \mathsf{CP_{ABE}.KeyGen}, \mathsf{CP_{ABE}.Enc}, \mathsf{CP_{ABE}.Dec})$ *is* indistinguishable under chosen-ciphertext attacks *(or is* IND-CCA secure*) if all p.p.t. adversaries have at most negligible advantage in the above game.*

We next recall the other type of ABE scheme, namely KP-ABE.

Definition 2.12 (KP-ABE scheme) *A* key-policy ABE (KP-ABE) scheme *is defined by five algorithms, which are as follows.*

$\mathsf{KP_{ABE}.PG}$: This algorithm takes as input the security parameter 1^λ and returns the system's parameters *pars*. These will include a description of the message space MsgSp, the ciphertext space CtSp and the universe of attributes U associated to the scheme. We write this as $pars \leftarrow \mathsf{KP_{ABE}.PG}(1^\lambda)$.

$\mathsf{KP_{ABE}.Setup}$: This algorithm takes as input *pars* and returns a master public key KP-MPK and a master secret key KP-MSK. We write $(\text{KP-MPK}, \text{KP-MSK}) \leftarrow \mathsf{KP_{ABE}.Setup}(pars)$.

$\mathsf{KP_{ABE}.KeyGen}$: This is a key generation algorithm that on input KP-MPK, KP-MSK and an access structure $\mathbb{A}$ outputs a secret key $sk_\mathbb{A}$. We write this as $sk_\mathbb{A} \leftarrow \mathsf{KP_{ABE}.KeyGen}(\text{KP-MPK}, \text{KP-MSK}, \mathbb{A})$.

$\mathsf{KP_{ABE}}$.Enc: This is an encryption algorithm that on input KP-MPK, a message $M \in \mathsf{MsgSp}$ and a set of attributes $S \in U$ returns a ciphertext $C \in \mathsf{CtSp}$. We write this as $C \leftarrow \mathsf{KP_{ABE}.Enc}(\text{KP-MPK}, M, S)$.

$\mathsf{KP_{ABE}}$.Dec: This is a decryption algorithm that on input KP-MPK, a ciphertext C and a secret key $sk_{\mathbb{A}}$ returns either a message or a failure symbol $\perp$. We write this as $\mathsf{KP_{ABE}.Dec}(\text{KP-MPK}, C, sk_{\mathbb{A}}) = M$, where $M \in \mathsf{MsgSp} \cup \{\perp\}$.

These algorithms are required to satisfy the following correctness property*: For every λ, for every pars output by $\mathsf{KP_{ABE}.PG}$, for every KP-MPK,KP-MSK output by $\mathsf{KP_{ABE}.KeyGen}$, for every message $M \in \mathsf{MsgSp}$, for every access structure $\mathbb{A}$ supported by the system and every set of attributes $S \in U$, if $sk_{\mathbb{A}} \leftarrow \mathsf{KP_{ABE}.KeyGen}(KP\text{-}MPK, KP\text{-}MSK, \mathbb{A})$, if $C \leftarrow \mathsf{KP_{ABE}.Enc}(KP\text{-}MPK, M, S)$ and if $S \in \mathbb{A}$, then $\mathsf{KP_{ABE}.Dec}(KP\text{-}MPK, C, sk_{\mathbb{A}}) = M$.*

We define *indistinguishability under chosen-ciphertext attacks* (IND-CCA) for a KP-ABE scheme $K = (\mathsf{KP_{ABE}.PG}, \mathsf{KP_{ABE}.Setup}, \mathsf{KP_{ABE}.KeyGen}, \mathsf{KP_{ABE}.Enc}, \mathsf{KP_{ABE}.Dec})$ in the following way.

IND-CCA security game for KP-ABE

Setup. The challenger $\mathcal{C}$ runs $\mathsf{KP_{ABE}.PG}(1^\lambda)$ to obtain $pars$ and $\mathsf{KP_{ABE}.Setup}(pars)$ to generate the master public key KP-MPK and the master secret key KP-MSK and gives ($pars$, KP-MPK) to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ has access to a secret-key-extraction oracle $\mathcal{O}^{\text{KP-MSK}}$, to obtain secret keys for access structures $\mathbb{A}_i$. $\mathcal{A}$ has also access to a decryption oracle $\mathcal{O}^{\text{KP-MSK}}$, to which it submits queries of the type $(C, \mathbb{A})$, where $\mathbb{A}$ is an access structure supported by the system. Such an oracle will respond with $\mathsf{KP_{ABE}.Dec}(\text{KP-MPK}, C, sk_{\mathbb{A}})$, where $sk_{\mathbb{A}}$ is extracted using KP-MSK.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and a challenge set of attributes $S^\star \subseteq U$ with the restriction that none of the access structures for which a key was queried in Phase 1 are satisfied by $S^\star$. $\mathcal{A}$ passes $M_0, M_1, S^\star$ to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0, 1\}$ and computes

$C^\star \leftarrow \mathsf{KP_{ABE}.Enc}(\text{KP-MPK}, M_b, S^\star)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to have access to a secret-key-extraction oracle $\mathcal{O}^{\text{KP-MSK}}$, with the same restriction we have in the Challenge phase, and to a decryption oracle $\mathcal{O}^{\text{KP-MSK}}$, with the restriction that it cannot submit the query $(C^\star, \mathbb{A})$, for any access structure $\mathbb{A}$ satisfied by $S^\star$.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage in the above game as $\mathbf{Adv}^{\text{IND}-\text{CCA}}_{\mathcal{A},K}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 2.13 (IND-CCA) *A KP-ABE scheme* $K = (\mathsf{KP_{ABE}.PG}, \mathsf{KP_{ABE}.Setup}, \mathsf{KP_{ABE}.KeyGen}, \mathsf{KP_{ABE}.Enc}, \mathsf{KP_{ABE}.Dec})$ *is* indistinguishable under chosen-ciphertext attacks *(or is* IND-CCA secure*) if all p.p.t. adversaries have at most negligible advantage in the above game.*

The security models we have presented so far address adversaries that can adaptively issue queries before and after the challenge phase. Most relevant work in the area [53, 32, 12, 70], however, achieves security only in the *selective* setting, where a less powerful adversary has to select *a priori* the set (or policy) he wishes to be challenged on. In [61], Lewko et al. provide a *fully* secure ABE scheme in the more general framework of functional encryption. Follow-up work in the area (for instance, [69]) also achieves this level of security.

Anonymity in ABE. As for previously considered primitives such as PKE and IBE, also in the context of ABE the issue of anonymity arises naturally. In the case of CP-ABE we call it *policy-hiding* property, reflecting that the ciphertext does not leak under what policy (access structure) it was created. Similarly for KP-ABE we denote it *attribute-hiding* property, capturing the idea that an adversary cannot tell what set of attributes a message was encrypted for. This notion was first introduced and achieved in the context of predicate encryption [58] for the special class of predicates defined by inner products.
These security notions are modeled in the natural way: an adversary selects *two* access structures (respectively, attribute sets) in the challenge phase and he will

have to guess under what access structure (attribute set) encryption was performed. Throughout the game the adversary will have access to secret-key-extraction and decryption oracles, to which he can submit queries having the obvious restrictions that prevent him from winning trivially.

We have so far introduced several encryption primitives. We next give the basic notions for two other primitives in the public-key setting which will be relevant to our work, namely digital signatures and commitments.

2.3.4 Digital signatures

In a digital signature scheme, or simply a signature scheme, a *signer* uses a secret key to sign a message and *anyone* can verify its validity using the corresponding (public) verification key. This is formalized in the following definition.

Definition 2.14 (Signature scheme) *A* signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *is defined by three algorithms, which are as follows.*

Gen: This is a key-generation algorithm that takes as input the security parameter 1^λ and outputs the system's parameters, which include a description of the message space MsgSp, the key space KSp and the signature space SSp, and a signing-verification key pair $(sigk, vk)$. For ease of exposition, we will consider the parameters implicit and simply write $(sigk, vk) \leftarrow \mathsf{Gen}(1^\lambda)$.

Sign: This is a signing algorithm that on input a signing key $sigk$ and a message $M \in \mathsf{MsgSp}$ outputs a signature $\sigma \in \mathsf{SSp}$. We write this as $\sigma \leftarrow \mathsf{Sign}(sigk, M)$.

Ver: This is a verification algorithm that takes as input a verification key vk, a message M and a signature σ and outputs a bit $b \in \{0, 1\}$. We write this as $\mathsf{Ver}(vk, M, \sigma) = b$.

These algorithms are required to satisfy the following property: For every λ, for every message $M \in \mathsf{MsgSp}$ and every key-pair $(sigk, vk)$ generated by Gen, if $\sigma \leftarrow \mathsf{Sign}(sigk, M)$ then $\mathsf{Ver}(vk, M, \sigma) = 1$.

We next define a security notion for a signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ which will be relevant in this thesis, namely the notion of strong unforgeability under a one-time message attack (SUF-1CMA). Consider the following game.

SUF-1CMA security game for a digital signature scheme

Setup. The challenger $\mathcal{C}$ runs $\mathsf{Gen}(1^\lambda)$ to generate a signing-verification key pair $(sigk, vk)$ and gives vk to the adversary $\mathcal{A}$.

Signing Query. $\mathcal{A}$ selects a message $M \in \mathsf{MsgSp}$ and gives it to $\mathcal{C}$. $\mathcal{C}$ computes $\sigma = \mathsf{Sign}(sigk, M)$. σ is passed to $\mathcal{A}$.

Forgery. $\mathcal{A}$ outputs a pair $(M^\star, \sigma^\star)$.

$\mathcal{A}$'s advantage is defined as $\mathbf{Adv}^{\mathrm{SUF-1CMA}}_{\mathcal{A},\Sigma}(\lambda) = \Pr[\mathsf{Ver}(vk, M^\star, \sigma^\star) = 1 \wedge (M^\star, \sigma^\star) \neq (M, \sigma)]$.

Intuitively, this models the idea that an adversary cannot produce a new valid signature even on the previously signed message.

Definition 2.15 (SUF-1CMA) *A signature scheme* $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *is* SUF-1CMA secure *(or strongly one-time) if all polynomial-time adversaries have at most negligible advantage in the above game.*

2.3.5 Commitments

Definition 2.16 (Commitment scheme) *A non-interactive* commitment scheme $CMT = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Vrfy})$ *is defined by three algorithms as follows.*

CPG: This is a common parameters generation algorithm that takes as input the security parameter 1^λ and outputs the scheme's parameters $cpars$. These will include a description of the commitable value space VSp. We write this as $cpars \leftarrow \mathsf{CPG}(1^\lambda)$.

Com: This is a committing algorithm that on input the parameters $cpars$ and a value $x \in \mathsf{VSp}$ returns a commitment com to x and a decommitment key dec. We write this as $(com, dec) \leftarrow \mathsf{Com}(cpars, x)$.

Vrfy: This is a verification algorithm that takes as input the common parameters $cpars$, a value $x \in \mathsf{VSp}$, a commitment com and a decommitment key dec and outputs a bit $b \in \{0, 1\}$. We write this as $\mathsf{Vrfy}(cpars, x, com, dec) = b$.

These algorithms are required to satisfy the following correctness property*: For every λ, for every set of parameters $cpars$ generated by CPG and for every value $x \in \mathsf{VSp}$, if $(com, dec) \leftarrow \mathsf{Com}(cpars, x)$ then $\mathsf{Vrfy}(cpars, x, com, dec) = 1$.*

Let $CMT = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Vrfy})$ be a commitment scheme. We recall two standard security properties for CMT, namely *hiding* and *binding*. The former property models the idea that an adversary cannot learn information about the committed value, while the latter that an adversary cannot find two *distinct* inputs that commit to the *same* value. Consider the following game.

Hiding security game for a commitment scheme

Setup. The challenger $\mathcal{C}$ runs $\mathsf{CPG}(1^\lambda)$ to generate the common parameters $cpars$ and gives $cpars$ to the adversary $\mathcal{A}$.

Challenge. $\mathcal{A}$ selects two values x_0 and $x_1 \in \mathsf{VSp}$ and gives them to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0, 1\}$ and computes $(com, dec) = \mathsf{Com}(cpars, x_b)$. The value com is passed to $\mathcal{A}$.

Guess. The adversary outputs its guess b' for b.

$\mathcal{A}$'s advantage is defined as $\mathbf{Adv}^{\text{Hiding}}_{\mathcal{A},CMT}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 2.17 (Hiding) *A commitment scheme $CMT = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Vrfy})$ is* hiding *if all polynomial-time adversaries have at most negligible advantage in the above game.*

Binding security game for a commitment scheme

Setup. The challenger $\mathcal{C}$ runs $\mathsf{CPG}(1^\lambda)$ to generate the common parameters $cpars$ and gives $cpars$ to the adversary $\mathcal{A}$.

Collision finding. $\mathcal{A}$ outputs a tuple $(com, x_0, x_1, dec_0, dec_1)$.

$\mathcal{A}$'s advantage is defined as $\mathbf{Adv}^{\text{Binding}}_{\mathcal{A},CMT}(\lambda) = \Pr[\mathsf{Vrfy}(cpars, x_0, com, dec_0) = 1 \quad \wedge \mathsf{Vrfy}(cpars, x_1, com, dec_1) = 1 \quad \wedge x_0, x_1 \in \mathsf{VSp} \text{ such that } x_0 \neq x_1]$.

Definition 2.18 (Binding) *A commitment scheme $CMT = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Vrfy})$ is* binding *if all polynomial-time adversaries have at most negligible advantage in the above game.*

2.4 Cryptographic tools and techniques

We have introduced the definitions and security models for some cryptographic primitives, selecting the ones which will be used in this thesis as building blocks for more advanced primitives. We next recall two useful cryptographic tools which are relevant in achieving some of the results in the following chapters: the first is an important proof technique based on hybrid arguments, and the second is a powerful transformation to build IND-CCA secure PKE from sID-IND-CPA secure IBE.

2.4.1 A useful proof technique

As mentioned in Section 2.2 and seen in the sections which followed, we define security for a primitive as a game between an adversary and a challenger modelled as probabilistic algorithms interacting with each other. The winning condition for the game typically depends on some particular event E occurring, and security is defined in terms of its probability being negligibly close to a "target probability", such as 0, 1/2 or the probability of some other event in some other game (where the same adversary is interacting with a different challenger). Showing that this holds is a proof of security for the primitive.

A very popular security proof technique in modern cryptography is to consider a *sequence* of games, instead of just the single one defined in the model. Since we extensively make use of such technique in this thesis, we briefly recall the main idea behind it, referring to [85] for a more detailed exposition of the topic.

We start by describing a sequence of games G_0, G_1 up to G_n, where G_0 is the original game, as defined in a security model specific to a primitive and an adversary type. For each game G_i we define an event E_i, somehow related to E, the event on which the winning condition for game G_0 depends. The aim is to show that $\Pr[E_i]$ is negligibly close to $\Pr[E_{i+1}]$ for all $i \in \{0, ..., n-1\}$ and that $\Pr[E_n]$ is negligibly close to the target probability. To prove security we need to evaluate $|\Pr[E_i] - \Pr[E_{i+1}]|$, and this is typically done by basing the transitions between G_i and G_{i+1} on one of the following [85]: *indistinguishability*, *failure events* or a *formal change*. In the first case, the idea is that we show that if an adversary detects the change between two successive games then we can build another adversary that is able to distinguish between two distributions that are meant to be indistiguishable. In the second case, when two successive games G_i and G_{i+1} are identical up to the occurrence of a certain failure event, we show that if the probability of such event occurring is negligible then so is $|\Pr[E_i] - \Pr[E_{i+1}]|$. This is known as the Difference Lemma [85, Lemma 1]. Finally, it could be the case that simple formal changes are made between G_i and G_{i+1}, so as to make the proof easier to follow, and hence $\Pr[E_i] = \Pr[E_{i+1}]$. Examples of this proof technique being adopted can be found for instance in Sections 4.3.1 and 4.3.2 of this thesis.

2.4.2 A useful transformation

In 2004, Canetti et al. [25] introduced a very powerful transformation to achieve IND-CCA secure PKE schemes from any weakly CPA-secure IBE scheme.

Informally, the transformation works as follows: the public key and secret key of the PKE scheme are simply the master public key and master secret key of the IBE scheme, respectively. To encrypt a message, the sender first generates a signing-verification key pair $(sigk, vk)$ for a strong one-time signature scheme, and then he runs the identity-based encryption algorithm on input the message and the *identity*

vk. The resulting ciphertext c is then signed using $sigk$ to obtain a signature σ. The final ciphertext consists of the verification key vk, the IBE ciphertext c, and the signature σ. To decrypt a ciphertext (vk, c, σ), the receiver first verifies the signature on c with respect to the verification key vk. If it fails, he outputs $\perp$. Otherwise, the receiver runs the identity-based key-extraction algorithm on identity vk to obtain the corresponding secret key sk_{vk}, which he then uses to decrypt c.

We recall the details of this transformation next.

Let $I = (\mathsf{IBE.PG}, \mathsf{IBE.Setup}, \mathsf{IBE.KeyExt}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ be an IBE scheme and let Σ=(Gen,Sign,Ver) be a signature scheme. The PKE scheme $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ is constructed from I and Σ as follows.

$\mathsf{PKE.PG}(1^\lambda)$: Run $\mathsf{IBE.PG}(1^\lambda)$ and return $pars$.

$\mathsf{PKE.KeyGen}(pars)$: Run $\mathsf{IBE.Setup}(pars)$ to obtain a master public key ID-MPK and a master secret key ID-MSK. These will be the public key pk and secret key sk of the PKE scheme, respectively.

$\mathsf{PKE.Enc}(pars, M, pk)$: Run $\mathsf{Gen}(1^\lambda)$ to obtain a signing-verification key pair ($sigk$, vk). Compute $c \leftarrow \mathsf{IBE.Enc}(pk, M, vk)$, where pk acts as the master public key of the IBE scheme and vk acts as an identity. Run $\mathsf{Sign}(sigk, c)$ and let σ be the resulting signature. The final ciphertext is $C = (vk, c, \sigma)$.

$\mathsf{PKE.Dec}(pars, pk, C, sk)$: Parse C into (vk, c, σ) and check whether $\mathsf{Ver}(vk, c, \sigma) = 1$. If not, output $\perp$. Otherwise compute $sk_{vk} \leftarrow \mathsf{IBE.KeyExt}(pk, sk, vk)$, the secret key corresponding to identity vk, and output $\mathsf{IBE.Dec}(pk, c, sk_{vk})$.

In [25], the following result is shown to hold.

Theorem 2.19 *[25, Theorem 1] If I is an sID-IND-CPA secure IBE scheme and Σ is a SUF-1CMA secure signature scheme then Π is an IND-CCA secure PKE scheme.*

Not only is this transformation an extremely powerful tool, it has also the advantage of being very flexible. Indeed, we will suitably adapt the key ideas behind

it to achieve CCA security in a variety of settings, as we will explore in detail in Chapter 4 and Chapter 5.

We are now ready to present our results. We start with our contributions to the area of robust public-key encryption.

Chapter 3

Robust Public-Key Encryption

Contents

In this chapter we study the notion of robustness in public-key encryption. After recalling existing definitions, we define stronger notions and provide motivation for our work. Furthermore, we classify these new notions in terms of implications and separation results, and we show how to generically achieve the strongest one we introduce. Part of the content of this chapter appears in [45], which is joint work with Pooya Farshim, Benoît Libert and Kenneth G. Paterson.

3.1 Introduction

3.1.1 Robustness and related work

A commonly pursued goal in cryptography is message privacy, which is typically achieved by means of encryption. In recent years, the privacy of users has become an equally relevant concern. It has led the research community to strive for anonymity properties when designing cryptographic primitives. In particular, *key-privacy* was introduced by [8] in the public-key setting to capture the idea that a ciphertext does not leak any information about the public key under which it was created, making therefore the communication anonymous. In this context, Abdalla, Bellare and Neven [2] raised a fundamental question: how does a legitimate user know if an anonymous ciphertext is intended for him? Moreover, what happens if he uses his secret key on a ciphertext *not* created under his public key? To address this question, Abdalla et al. formalized a property called *robustness*, which (informally speaking) guarantees that decryption attempts fail with high probability if the "wrong" private key is used, and argued that, in all applications requiring anonymous public-key encryption, robustness is usually needed as well. These applications include auction protocols with bid privacy [81], consistency [1] in searchable encryption [15] and anonymous broadcast encryption (Chapter 4). As shown by Mohassel [66], robustness is also important in guaranteeing the anonymity of hybrid encryption schemes resulting from the combination of anonymous asymmetric and symmetric components.

Robustness ensures that a ciphertext cannot correctly decrypt under two different secret keys. This notion has (often implicitly) been present in the literature (e.g. [81, 58, 6]), but formal definitions remained lacking until the recent foundational work of Abdalla et al. [2]. In particular, the authors introduced two flavours of encryption robustness: *weak* and *strong* robustness. Weak robustness is modeled as a game in which a winning adversary outputs a valid message M and two distinct public keys pk_0 and pk_1 such that the encryption of M under pk_0 decrypts to a valid message under sk_1, the secret key corresponding to pk_1. Strong robustness allows for a more powerful adversary which gets to choose a ciphertext C (as opposed to a message which will be honestly encrypted) and outputs it together with two

distinct public keys. The adversary wins if C decrypts to a valid message under both corresponding secret keys.

Achieving robustness is not as straightforward as it might seem. As pointed out by Abdalla et al. [2], merely appending the receiver's public key to the ciphertext is not an option for providing robustness, since it destroys key-privacy properties. Abdalla et al. also showed that the seemingly natural solution of using an unkeyed redundancy function to modify the message before encryption does not achieve even weak robustness, thus demonstrating the non-triviality of the problem. The authors of [2] then gave anonymity-preserving constructions to obtain both weak and strong robustness for public-key encryption. Using a simple tweak, they also showed how to render the Cramer-Shoup cryptosystem [36] strongly robust without introducing any overhead.

More recently, Mohassel [66] studied robustness in the context of hybrid encryption [37]. He showed that weak robustness (and not only anonymity) is needed in the asymmetric part of a hybrid encryption scheme to ensure anonymity of the overall scheme. Mohassel also considered relaxations, called *collision-freeness*, of both weak and strong robustness. He showed that many constructions in the literature are natively collision-free and showed how to generically turn any weakly (resp. strongly) collision-free cryptosystem into a weakly (resp. strongly) robust one.

In [62] we proved the strong robustness of a variant of the Kurosawa-Desmedt [60] cryptosystem.

3.1.2 Our contributions

This chapter is dedicated to the development and study of new notions of robustness in the context of public-key encryption. We will first look at existing definitions and point out some of their limitations, justifying the need for stronger notions. These are obtained by progressively removing various restrictions on the capabilities of the adversary in the strong robustness security model. We then show how these notions relate to each other and thus give a more complete picture of robustness in general. We finally present ways of achieving the introduced notions.

This work wishes to provide a more in depth study and an overall better understanding of robustness. We do this by introducing stronger and simpler notions which help unify and clarify this research area. Robustness is an important property which is relevant especially in the context of anonymity. Indeed, it will be key in achieving anonymous broadcast encryption, a primitive we develop in Chapter 4.

3.2 Weak and Strong Robustness

This section recalls the definitions of robust public-key encryption given by Abdalla, Bellare and Neven [2]. We first present the notion of **weak robustness**, which informally models the idea that an adversary cannot come up with a message and two distinct public keys such that the encryption of that message under the first public key returns a valid message when decrypted with the secret key corresponding to the second public key. We formalize this next.

Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a PKE scheme. Let us consider the following game.

WROB-CCA security game

Setup. The challenger $\mathcal{C}$ runs $\mathsf{PKE.PG}(1^\lambda)$ to generate the common parameters $pars$, which are passed on to $\mathcal{A}$.

Query Phase. On a polynomial number of occasions, $\mathcal{A}$ may submit the following queries:

– **Public-key query**: $\mathcal{C}$ generates and stores a key-pair (pk, sk) and returns pk to $\mathcal{A}$. We call pk a *valid* public key.

– **Secret-key query** for pk: If pk is valid, $\mathcal{C}$ returns sk, the secret key corresponding to pk. Otherwise, it returns $\perp$.

– **Decryption query** (C, pk): If pk is valid, $\mathcal{C}$ returns $\mathsf{PKE.Dec}(pars, pk, C, sk)$. Otherwise, it returns $\perp$.

Finalize. $\mathcal{A}$ outputs (M, pk_0, pk_1).

$\mathcal{A}$ is a winning adversary if its output satisfies the following conditions:

1. both pk_0 and pk_1 are *valid* public keys (i.e. they are honestly generated public keys output by $\mathcal{C}$ in the Query Phase);
2. neither sk_0 nor sk_1, the secret keys corresponding to pk_0 and pk_1 respectively, has been queried in the query phase;
3. $pk_0 \neq pk_1$;
4. $M \neq \perp \wedge \mathsf{PKE.Dec}(\mathsf{PKE.Enc}(M, pk_0), sk_1) \neq \perp$.

We define $\mathcal{A}$'s advantage as being the probability, taken over all random coins, of outputting a tuple (M, pk_0, pk_1) satisfying all of the above conditions.

Definition 3.1 (WROB-CCA[2]) *A PKE scheme* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is* weakly robust under chosen-ciphertext attacks *(or WROB-CCA) if all polynomial-time adversaries have at most negligible advantage in the above game.*

In [2] the authors present a generic transformation, for both the public-key and the identity-based settings, conferring weak robustness to chosen-ciphertext (and chosen plaintext) secure schemes. The key idea is to append some publicly-known and keyed redundancy to the message before encryption, and to check for it upon decryption. This allows for weak robustness to be efficiently and generically achieved. The notion of weak robustness is of interest since it precisely addresses the issue of *using the wrong key* that arises in anonymity contexts (such as anonymous broadcast encryption [6, 62], for instance), but it is also useful in achieving strong robustness.

Strong robustness was proposed by Abdalla et al. [2] as a, not surprisingly, stronger notion of robustness, which allows for adversarially generated ciphertexts. Roughly speaking, a scheme is strongly robust if an adversary cannot produce a ciphertext that decrypts to a valid message under two distinct keys. We formalize this idea next.

Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a PKE scheme. Let us consider the following game.

SROB-CCA security game

Setup. The challenger $\mathcal{C}$ runs $\mathsf{PKE.PG}(1^\lambda)$ to generate the common parameters $pars$, which are passed on to $\mathcal{A}$.

Query Phase. On a polynomial number of occasions, $\mathcal{A}$ may submit the following queries:

– **Public-key query**: $\mathcal{C}$ generates and stores a key-pair (pk, sk) and returns pk to $\mathcal{A}$. We call pk a *valid* public key.

– **Secret-key query** for pk: If pk is valid, $\mathcal{C}$ returns sk, the secret key corresponding to pk. Otherwise, it returns $\perp$.

– **Decryption query** (C, pk): If pk is valid, $\mathcal{C}$ returns $\mathsf{PKE.Dec}(pars, pk, C, sk)$. Otherwise, it returns $\perp$.

Finalize. $\mathcal{A}$ outputs (C, pk_0, pk_1).

$\mathcal{A}$ is a winning adversary if its output satisfies the following conditions:

1. both pk_0 and pk_1 are *valid* public keys;
2. neither sk_0 nor sk_1, the secret keys corresponding to pk_0 and pk_1 respectively, has been queried in the query phase;
3. $pk_0 \neq pk_1$;
4. $\mathsf{PKE.Dec}(C, sk_0) \neq \perp \wedge \mathsf{PKE.Dec}(C, sk_1) \neq \perp$.

We define $\mathcal{A}$'s advantage as the probability of outputting a tuple (C, pk_0, pk_1) satisfying all of the above conditions.

Definition 3.2 (SROB-CCA[2]) *A PKE scheme* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is* strongly robust under chosen-ciphertext attacks *(or SROB-CCA) if all polynomial-time adversaries have at most negligible advantage in the above game.*

The authors of [2] show, and it is easy to see, that SROB-CCA implies WROB-CCA. Indeed, an adversary against the weak robustness of a scheme can be transformed into a strong robustness adversary simply by taking its output (M, pk_0, pk_1), encrypting M under pk_0 and outputting the resulting ciphertext together with pk_0 and pk_1.

We note that it is possible to define the chosen-plaintext analogues of weak and strong robustness (WROB-CPA and SROB-CPA, respectively) simply by disallowing decryption queries.

Abdalla et al. show that also strong robustness is achievable generically by applying a particular transformation. The main idea of the transformation is to include, as part of the ciphertext, a commitment to the public key: the decommitment key is encrypted along with the message and the commitment is appended to the ciphertext. We give details of this transformation, which we call the ABN transformation, in Section 3.5.1.

We stress that the difference between WROB-CCA and SROB-CCA is that in the former the adversary has to produce a message, while in the latter it has to output a ciphertext, which may not have been obtained as an honest encryption. In [2] the need for the strong robustness notion is motivated by scenarios where ciphertexts can be adversarially chosen. The authors of [2] give Sako's auction protocol [81] as an example of such a situation, explaining that strong robustness is required in order to prevent an attack on its fairness mounted by a cheating bidder and a colluding auctioneer.

As a first motivating step towards the development of new and stronger notions of robustness we show that strong robustness is actually not sufficient to prevent such types of attack. We next take a closer look at Sako's protocol [81] and present a new attack on its fairness.

3.2.1 An attack on the fairness of Sako's protocol

Sako's auction protocol [81] was the first practical protocol to ensure *bid privacy*, i.e. to hide the value of losing bids. The basic idea is the following[1]: Let $V = \{v_1, ..., v_N\}$ be the set of possible bid values. The auctioneer prepares N key-pairs $(pk_i, sk_i)_{i\in\{1,...,N\}}$ and publishes the N public keys. To *bid* for a value v_i a bidder encrypts a pre-determined message M under the public key pk_i. This is signed and posted by the bidder. To *open* a bid the auctioneer takes the largest value v_N in V and attempts to decrypt the encrypted bids one by one using sk_N. If at least one decrypts to M, the auctioneer publishes the winning bid v_N, a list of all the winning bidders and the secret key sk_N for the bidders to verify. If no decryption returns M, the auctioneer repeats the procedure using sk_{N-1}, and so on. For the auction to hide the values of losing bids, the underlying public-key encryption scheme is required to be key-private, in the sense of [8].

In [81], Sako provides an example of an auction protocol scheme based on the ElGamal cryptosystem, which is key-private. In [2], Abdalla et al. give an attack which allows a cheating bidder and a colluding auctioneer to break the *fairness* of the protocol. Informally, this property ensures that a cheating bidder does not have an unfair advantage over an honest one (for instance, a bidder should not be able to see another bidder's encryption and produce an encryption of a bid that is one value higher). This attack is based on the fact that ElGamal is not robust and results in the auctioneer being able to open the cheating bidder's bid to an arbitrary (winning) value. To prevent this attack, the authors of [2] suggest using any *strongly* robust scheme (strong robustness, instead of simply weak robustness, is required since the ciphertexts are generated adversarially).

We now show that strong robustness is *not* sufficient to prevent an attack of this type to the fairness of Sako's protocol. More precisely, we present an attack to the protocol when instantiated with a variant of the Cramer–Shoup encryption scheme, CS*, which is key-private and strongly robust (the latter result was proved in [2]). We first recall the CS* scheme.

[1] For the purpose of this thesis, we focus on the public-key setting.

CS* encryption scheme

The common public parameters consist of a group $\mathbb{G}$ of prime order p and the description of a family of functions $H : \mathsf{Keys}(H) \times \mathbb{G}^3 \to \mathbb{G}$.

$\mathsf{PG}(1^\lambda)$: Choose $K \leftarrow \mathsf{Keys}(H)$, $g_1 \leftarrow \mathbb{G}$ and $w \leftarrow \mathbb{Z}_p^*$. Let $g_2 = g_1^w$. Return $pars = (K, g_1, g_2)$.

$\mathsf{KeyGen}(pars)$: Choose random exponents $x_1, x_2, y_1, y_2, z_1, z_2 \leftarrow \mathbb{Z}_p$ and compute

$$e = g_1^{x_1} g_2^{x_2}, \qquad f = g_1^{y_1} g_2^{y_2}, \qquad h = g_1^{z_1} g_2^{z_2}.$$

The public key is $pk = (e, f, h)$ and the private key is $sk = (x_1, x_2, y_1, y_2, z_1, z_2)$.

$\mathsf{Enc}(pars, M, pk)$: To encrypt a message $M \in \mathbb{G}$,

1. Pick $u \leftarrow \mathbb{Z}_p^*$ and compute

$$a_1 = g_1^u, \qquad a_2 = g_2^u, \qquad b = h^u,$$

2. Let $c \leftarrow b \cdot M, \qquad v \leftarrow H_K(a_1, a_2, c), \qquad d \leftarrow e^u f^{uv}$

The ciphertext is $C = (a_1, a_2, c, d)$.

$\mathsf{Dec}(C, sk)$: Parse the ciphertext C as (a_1, a_2, c, d). Compute $v = H_K(a_1, a_2, c)$, $M = c \cdot a_1^{-z_1} a_2^{-z_2}$. If $d \neq a_1^{x_1+y_1 v} a_2^{x_2+y_2 v}$ then set $M = \perp$. If $a_1 = 1$ then set $M = \perp$. Return M.

Just as with the attack of Abdalla et al. [2], the attack we present below on the Sako protocol instantiated with the scheme CS* assumes a dishonest bidder and a colluding auctioneer, and works as follows.

Let $V = \{v_1, ..., v_N\}$ be the set of possible bid values. The auctioneer runs $\mathsf{PG}(1^\lambda)$ to obtain the public parameters (K, g_1, g_2). He chooses a fixed message $M \in \mathbb{G}$ as per Sako's protocol. He selects $u, z_1, z_2 \leftarrow \mathbb{Z}_p^*$ and computes $a_1 = g_1^u$, $a_2 = g_2^u$, $b = a_1^{z_1} a_2^{z_2}$ and $c = b \cdot M$. He then computes $v = H_K(a_1, a_2, c)$. If $v = 0$, the auctioneer re-samples and re-computes the values, until $v \neq 0$. He then considers the following system of linear equations

$$\begin{cases} x_1 + vy_1 = \alpha_1 \mod p \\ x_2 + vy_2 = \alpha_2 \mod p \end{cases}$$

for some α_1, α_2 in $\mathbb{Z}_p$, and finds N distinct solutions $(x_{1,i}, x_{2,i}, y_{1,i}, y_{2,i})$ with $i \in \{1, ..., N\}$, where all the values are in $\mathbb{Z}_p$.

The auctioneer sets sk_i to be $(x_{1,i}, x_{2,i}, y_{1,i}, y_{2,i}, z_1, z_2)$ for $i \in \{1, ..., N\}$. He passes u to the cheating bidder and publishes all the public keys $pk_i = (g_1^{x_{1,i}} g_2^{x_{2,i}}, g_1^{y_{1,i}} g_2^{y_{2,i}}, g_1^{z_1} g_2^{z_2})$ with $i \in \{1, ..., N\}$.

The cheating bidder can now bid for the value v_i by encrypting M with randomness u under the public key pk_i to get ciphertext C. Such an encrypted bid C will decrypt to M under **any** sk_j with $j \in \{1, ..., N\}$, since $x_{1,i} + vy_{1,i} = x_{j_1} + vy_{j_1}$ and $x_{2,i} + vy_{2,i} = x_{j_2} + vy_{j_2}$, by construction. This means that during the protocol, the auctioneer can first observe the highest honest bid (say $h < N$). Then, he can declare the cheating bidder as the winner (for the bid $h + 1$) by revealing the private key sk_{h+1}. This clearly gives the dishonest bidder and colluding auctioneer a cheating strategy and breaks the fairness of the protocol.

Remark 1 It may be argued that the above attack can be detected by the bidders, as the maliciously generated public keys all share the same third component. Although this is a valid point, it may be unreasonable to assume that the bidders perform such checks outside the protocol description. Indeed, one (or *the*) goal of robustness is to ensure that such checks are *already* implemented within the decryption algorithm. Let us note that the attack of Abdalla et al. on the robustness of ElGamal also falls within the category of such "traceable" attacks, as the ciphertexts in their attack are of the form $(1, C)$. To further justify the relevance of the new notions, we demonstrate an untraceable attack on the modified Kurosawa–Desmedt encryption scheme (which is proven strongly robust under chosen-ciphertext attacks [62]) in Section 3.3.4.

Evidently, this attack shows that strong robustness is not enough to guarantee fairness in Sako's auction protocol. Intuitively what is needed is a stronger notion of robustness, wherein all the public keys and ciphertexts in the system may be adversarially generated. This is precisely the notion we will develop next.

3.3 A Direct Strengthening: Full Robustness

The attack in the previous section highlights the need for stronger notions of robustness. Developing such notions will not only address the limitations of the existing ones but it will also help in providing a more complete picture of the various flavours of robustness. We start by directly strengthening strong robustness. This will lead to a very natural and simple notion which we name *full robustness*. We then consider what later can be seen as its "dual" notion, namely *key-less robustness*. In this section, we define and relate these notions, providing implications and separating examples.

3.3.1 Full robustness

Recall that an SROB-CCA adversary has to output a ciphertext C and two public keys pk_0 and pk_1 such that C decrypts to a valid message M_0 under sk_0 and to a valid message M_1 under sk_1. The notion poses three restrictions on the public keys output by the adversary:

1. the public keys are honestly generated;
2. the corresponding secret keys cannot have been queried by the adversary;
3. pk_0 and pk_1 have to be distinct.

We will next see that by removing some of these restrictions we obtain increasingly stronger notions of robustness.

We start by observing that the last condition is *inherent* to modeling the behaviour of an encryption scheme when used on different public keys, and removing it would make it trivial for an adversary to win. In fact, an adversary can always encrypt a valid message M under a valid public key pk, obtaining C. By submitting (C, pk, pk) the adversary will win with probability 1, due to the correctness of the scheme[2].

[2]We assume that the public-key schemes we consider are *perfectly* correct.

3.3 A Direct Strengthening: Full Robustness

We now look at the notion resulting from the removal of restriction 2, i.e. the adversary is now allowed to query secret keys even for the finally output public keys. We call this notion **unrestricted strong robustness** (USROB). This game therefore proceeds as the SROB-CCA game in Definition 3.2 except that condition 2 is removed. This notion is powerful enough to model scenarios where keys are honestly generated, but an adversary trying to break the robustness of the scheme may know the secret keys.

If an adversary can control the generation of keys, it may be unreasonable to assume that it can only generate the keys honestly. We therefore strengthen USROB further by removing the first restriction on the adversary. We ask, however, that the adversary return the secret keys for the public keys that it chooses. Two points deserve further attention at this point. First, returning the secret keys is to allow for a polynomial-time game definition which is not excessively strong. Second, we do not require the secret keys to be valid. Indeed, it is the responsibility of the decryption algorithm to check that the key-pair it receives is valid. Note that as a result of removing the two restrictions, the adversary has now full control over the keys, and we no longer need to provide the adversary with the oracles present in the SROB-CCA and USROB games. These modifications result in a simple, but strong, notion which we call **full robustness** (FROB), and which we formalize below.

Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a PKE scheme. Let us consider the following game.

FROB security game

Setup. The challenger $\mathcal{C}$ runs $\mathsf{PKE.PG}(1^\lambda)$ to generate the common parameters $pars$ and passes them on to $\mathcal{A}$.

Finalize. $\mathcal{A}$ outputs $(C, pk_0, pk_1, sk_0, sk_1)$.

$\mathcal{A}$ is a winning adversary if its output satisfies the following conditions:

1. $pk_0 \neq pk_1$;
2. $\mathsf{PKE.Dec}(pars, pk_0, C, sk_0) \neq \perp \wedge \mathsf{PKE.Dec}(pars, pk_1, C, sk_1) \neq \perp$.

We define $\mathcal{A}$'s advantage as the probability of outputting a tuple $(C, pk_0, pk_1, sk_0, sk_1)$ satisfying both of the above conditions.

Definition 3.3 (FROB) *A PKE scheme* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is* fully robust *(or FROB) if all polynomial-time adversaries have at most negligible advantage in the above game.*

We note that we no longer require public key, secret key and decryption oracles and therefore there is no CPA or CCA notion for full robustness. This is because a FROB adversary is not restricted by conditions 1 and 2 and, having access to $pars$, it can implement the oracles on its own.

3.3.2 Key-less robustness

The new notion of full robustness accounts for an adversary which is allowed to output self-generated public keys. However, in order for the game to test the winning condition in polynomial time, such an adversary also has to output the corresponding secret keys. Such a requirement may be inconvenient for the adversary (who prefers not to give away the secret keys) or may even be unsatisfiable (the adversary may not know them!). To address this issue, we propose an alternative definition of robustness, called **key-less robustness** (KROB), where the adversary no longer needs to return any secret keys, but instead "opens" a ciphertext by providing the random coins and the message used in the encryption. More precisely, the adversary outputs two messages, two distinct public keys and two sets of random coins, and its goal is to produce a collision in the encryption algorithm. The game for key-less robustness is described next.

KROB security game

Setup. The challenger $\mathcal{C}$ runs $\mathsf{PKE.PG}(1^\lambda)$ to generate the common parameters $pars$ and passes them on to $\mathcal{A}$.

Finalize. $\mathcal{A}$ outputs $(M_0, M_1, pk_0, pk_1, r_0, r_1)$.

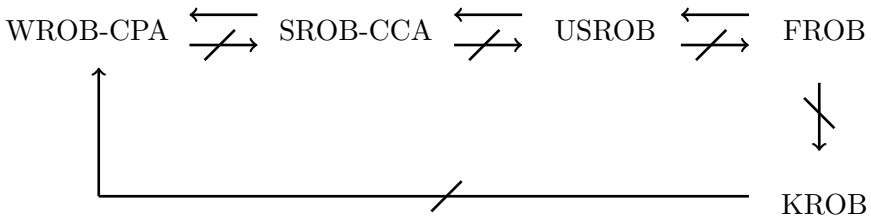

Figure 3.1: Relations among notions of robustness.

$\mathcal{A}$ is a winning adversary if its output satisfies the following conditions:

1. $pk_0 \neq pk_1$;
2. $\mathsf{PKE.Enc}(pars, M_0, pk_0; r_0) = \mathsf{PKE.Enc}(pars, M_1, pk_1; r_1)$.

We define $\mathcal{A}$'s advantage as the probability of outputting $(M_0, M_1, pk_0, pk_1, r_0, r_1)$ satisfying the above conditions.

Definition 3.4 (KROB) *A PKE scheme* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is* key-less robust *(or KROB) if all polynomial-time adversaries have at most negligible advantage in the above game.*

Intuitively this notion appears to be the strongest amongst the ones considered so far, since the adversary has the liberty to choose the public keys and does not have to reveal any secret information. Surprisingly, we will see that key-less robustness does not imply *any* of the other notions (FROB, SROB-CCA, and not even weak robustness). Furthermore, we will show that FROB does not imply KROB either.

3.3.3 Relations among notions of robustness

We now study how the various notions of robustness relate to each other. We summarize our initial findings in Figure 3.1. It is clear that FROB $\Rightarrow$ USROB $\Rightarrow$ SROB-CCA as the adversary becomes progressively more restricted in each game. For completeness, we provide proofs of these relations below.

Proposition 3.5 (FROB ⇒ USROB ⇒ SROB-CCA) *Let* Π *be a PKE scheme which is FROB (resp. USROB). Then it is also USROB (resp. SROB-CCA).*

Proof. We want to prove that if a PKE scheme $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ is FROB, then it is also USROB. Suppose there exists an adversary $\mathcal{A}$ breaking Π's USROB, then we can build an adversary $\mathcal{B}$ that interacts with $\mathcal{A}$ to break Π's FROB.

The game proceeds as follows. $\mathcal{B}$'s challenger $\mathcal{C}$ runs $\mathsf{PKE.PG}(1^\lambda)$ to obtain $pars$, which are passed to $\mathcal{B}$. $\mathcal{B}$ handles all of $\mathcal{A}$'s queries by simulating $\mathcal{A}$'s oracles (it can, since it knows $pars$). Finally $\mathcal{A}$ outputs (C, pk_0, pk_1) and $\mathcal{B}$ outputs $(C, pk_0, pk_1, sk_0, sk_1)$, where sk_0 and sk_1 are the secret keys corresponding to pk_0 and pk_1, respectively. We note that $\mathcal{B}$ knows such keys since it created them when generating the valid public keys pk_0 and pk_1 for $\mathcal{A}$. $\mathcal{B}$ provides a perfect simulation of $\mathcal{A}$'s environment and therefore has exactly the same advantage in winning the FROB game as $\mathcal{A}$ has in winning the USROB one.

For the second implication, we want to prove that if a PKE scheme $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ is USROB, then it is also SROB-CCA. Suppose there exists an adversary $\mathcal{A}$ breaking Π's SROB-CCA, then we can build an adversary $\mathcal{B}$ that interacts with $\mathcal{A}$ to break Π's USROB.

The game proceeds as follows. $\mathcal{B}$'s challenger $\mathcal{C}$ runs $\mathsf{PKE.PG}(1^\lambda)$ to obtain $pars$. $\mathcal{B}$ handles all of $\mathcal{A}$'s queries by forwarding them to $\mathcal{C}$. Finally $\mathcal{A}$ outputs (C, pk_0, pk_1) and $\mathcal{B}$ outputs the same. $\mathcal{B}$ has exactly the same advantage in winning the USROB game as $\mathcal{A}$ has in winning the SROB-CCA one. □

Next we show that USROB is strictly stronger than SROB-CCA, and that FROB is strictly stronger than USROB.

Proposition 3.6 (SROB-CCA ↛ USROB) *Let* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *be SROB-CCA. Then there is a scheme* $\Pi_1 = (\mathsf{PKE.PG}_1, \mathsf{PKE.KeyGen}_1, \mathsf{PKE.Enc}_1, \mathsf{PKE.Dec}_1)$ *which is SROB-CCA, but fails to be USROB.*

Proof. Informally, Π_1 is obtained from Π by running its algorithms, with the difference that a random string is appended to the secret key output by $\mathsf{PKE.KeyGen}$. Checking for such string will be an alternative decryption rule in case $\mathsf{PKE.Dec}$ returns $\perp$. More precisely, we define the required scheme Π_1 as follows.

$\mathsf{PKE.PG}_1(1^\lambda)$: Run $\mathsf{PKE.PG}(1^\lambda)$ to obtain $pars$. Return $pars$.

$\mathsf{PKE.KeyGen}_1(pars)$: Run $\mathsf{PKE.KeyGen}(pars)$ to obtain (pk, sk). Sample $s \leftarrow \{0,1\}^\lambda$ and return $(pk, (sk||s))$.

$\mathsf{PKE.Enc}_1(pars, M, pk)$: Run $\mathsf{PKE.Enc}(pars, M, pk)$ to obtain ciphertext C. Return C.

$\mathsf{PKE.Dec}_1(pars, pk, C, (sk||s))$: Parse C as $C||C'$. Run $\mathsf{PKE.Dec}(pars, pk, C, sk)$. If the output is a valid message M, return M. Otherwise, check if $s = C'$, and if so return a message M from the message space for pk. Else, return $\perp$.[3]

We first prove Π_1 is not USROB by constructing an adversary $\mathcal{A}$ which wins the USROB game against Π_1. Algorithm $\mathcal{A}$ queries the public-key oracle to obtain public keys pk_0 and pk_1. It then queries the secret-key-extraction oracle to receive $(sk_0||s_0)$, the secret key corresponding to pk_0. It runs $\mathsf{PKE.Enc_1}(pars, M_1, pk_1)$, where M_1 is any (valid) message, obtaining C_1. $\mathcal{A}$ then sets $C_1' := s_0$ and outputs $(C := (C_1||C_1'), pk_0, pk_1)$ as its final output. It is easy to see that this is a winning strategy for $\mathcal{A}$: C when decrypted with respect to pk_1 will return M_1 due to the correctness of the scheme. Now, if we run the decryption algorithm on $C = (C_1||C_1')$ with respect to pk_0, C_1 will not decrypt to a valid message, due to the strong robustness of Π, however $C_1' = s_0$ and therefore we obtain a valid message M.

We now prove that Π_1 is SROB-CCA. Suppose there is an adversary $\mathcal{A}$ which wins the SROB-CCA game against Π_1. We construct an adversary $\mathcal{B}$ that interacts with $\mathcal{A}$ to win the SROB-CCA game against Π. $\mathcal{A}$'s challenger generates $pars$. $\mathcal{B}$ handles $\mathcal{A}$'s queries as follows:

– **Public-key query**: $\mathcal{B}$ invokes $\mathcal{C}$ to get a valid public key pk. It then selects and stores a random bit-string s of length λ. $\mathcal{B}$ gives pk to $\mathcal{A}$.

[3] An alternative decryption rule, which checks for the equality *before* running $\mathsf{PKE.Dec}$, is possible but only guarantees overwhelming, instead of perfect, correctness.

– **Secret-key query** for pk: $\mathcal{B}$ queries $\mathcal{C}$ for the corresponding secret key sk. $\mathcal{B}$ then appends s to sk and gives $(sk||s)$ as the response to $\mathcal{A}$.

– **Decryption query** $(C||C', pk)$: $\mathcal{B}$ passes (C, pk) to its own oracle. If the answer is a valid message M, $\mathcal{B}$ forwards M to $\mathcal{A}$. If the output is $\perp$, $\mathcal{B}$ checks whether C' is equal to s (which it holds). If so, $\mathcal{B}$ outputs a valid message as a response to $\mathcal{A}$'s query. If not, $\mathcal{B}$ returns $\perp$.

Finally, when $\mathcal{A}$ outputs $(C||C', pk_0, pk_1)$, $\mathcal{B}$ outputs (C, pk_0, pk_1).

We note that $\mathcal{B}$ provides a perfect simulation of $\mathcal{A}$'s environment and that $\mathcal{B}$ wins whenever $\mathcal{A}$ wins unless $\mathcal{A}$ does so by guessing the s-component of either sk_0 or sk_1. Since the s-components are random and information theoretically hidden from $\mathcal{A}$'s view the probability of this event is at most $2 \cdot \frac{1}{2^\lambda}$. This completes the proof. □

Proposition 3.7 (USROB $\nrightarrow$ FROB) *Let* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *be USROB. Then there is a scheme* $\Pi_2 = (\mathsf{PKE.PG}_2, \mathsf{PKE.KeyGen}_2, \mathsf{PKE.Enc}_2, \mathsf{PKE.Dec}_2)$ *which is USROB, but fails to be FROB.*

Proof. Informally, the required scheme Π_2 is derived from Π simply by prepending a zero-bit to the public key output by $\mathsf{PKE.KeyGen}$. This zero-bit is then discarded by the encryption and decryption algorithms. We define Π_2 more precisely as follows.

$\mathsf{PKE.PG}_2(1^\lambda)$: Run $\mathsf{PKE.PG}(1^\lambda)$ to obtain $pars$. Return $pars$.

$\mathsf{PKE.KeyGen}_2(pars)$: Run $\mathsf{PKE.KeyGen}(pars)$ to obtain (pk, sk). Return $(0||pk, sk)$.

$\mathsf{PKE.Enc}_2(pars, M, b||pk)$: Run $\mathsf{PKE.Enc}(pars, M, pk)$ to obtain C. Return C.

$\mathsf{PKE.Dec}_2(pars, b||pk, C, sk)$: Return $\mathsf{PKE.Dec}(pars, pk, C, sk)$.

Π_2 is not FROB. Consider the adversary $\mathcal{A}$ which runs $\mathsf{PKE.KeyGen}_2(pars)$ to get a valid key-pair $(0||pk, sk)$, picks a random M and runs $\mathsf{PKE.Enc}_2(pars, M, 0||pk)$ to obtain a ciphertext C. $\mathcal{A}$ gives $(C, 0||pk, 1||pk, sk, sk)$ as its final output. It is

easy to see that $\mathcal{A}$ wins with probability 1: $0||pk \neq 1||pk$ and the decryption of C with the secret key sk returns a valid message due to correctness.

We now show Π_2 is USROB. Suppose there is an adversary $\mathcal{A}$ which wins the USROB game against Π_2. We construct an adversary $\mathcal{B}$ that interacts with $\mathcal{A}$ to win the USROB game against Π with the same probability. The challenger $\mathcal{C}$ generates $pars$. $\mathcal{B}$ handles $\mathcal{A}$'s queries by forwarding them to its own oracles prepending or removing a zero-bit to all the public keys sent and received as appropriate. Finally, when $\mathcal{A}$ outputs $(C, 0||pk_0, 0||pk_1)$ with $pk_0 \neq pk_1$, $\mathcal{B}$ also outputs (C, pk_0, pk_1). It's easy to see that $\mathcal{B}$ provides a perfect simulation of $\mathcal{A}$'s environment, and that if $\mathcal{A}$ wins so does $\mathcal{B}$. □

The next proposition shows that KROB does not even imply WROB-CPA. It follows that KROB $\not\rightarrow$ USROB and KROB $\not\rightarrow$ FROB.

Proposition 3.8 (KROB $\not\rightarrow$ WROB-CPA) *Let* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *be KROB. Then there is a scheme* $\Pi_3 = (\mathsf{PKE.PG}_3, \mathsf{PKE.KeyGen}_3, \mathsf{PKE.Enc}_3, \mathsf{PKE.Dec}_3)$ *which is KROB, but fails to be WROB-CPA.*

Proof. We define the required scheme Π_3 to be identical to Π except for its decryption algorithm, which we modify as follows:

$\mathsf{PKE.Dec}_3(pars, pk, C, sk)$: If $\mathsf{PKE.Dec}(pars, pk, C, sk)$ is a valid message M, return M. If not, return a valid fixed message M from the message space.

It is easy to see that Π_3 is not even WROB-CPA as the decryption algorithm never returns $\perp$. However, the modified scheme is still KROB as the tweaked decryption algorithm does not affect the KROB game. □

Finally, we show also that FROB does not imply KROB, separating the two seemingly stronger notions so far and completing the relations represented in Figure 3.1.

Proposition 3.9 (FROB $\nrightarrow$ KROB) *Let* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *be FROB. Then there is a scheme* $\Pi_4 = (\mathsf{PKE.PG}_4, \mathsf{PKE.KeyGen}_4, \mathsf{PKE.Enc}_4, \mathsf{PKE.Dec}_4)$ *which is FROB, but fails to be KROB.*

Proof. We define the required scheme Π_4 as follows.

$\mathsf{PKE.PG}_4(1^\lambda)$: Run $\mathsf{PKE.PG}(1^\lambda)$ to obtain $pars$. Return $pars$.

$\mathsf{PKE.KeyGen}_4(pars)$: Run $\mathsf{PKE.KeyGen}(pars)$ to obtain (pk, sk). Return $(0||pk, sk)$.

$\mathsf{PKE.Enc}_4(pars, M, b||pk)$: If $b = 1$, output a fixed ciphertext $C^\star$. If $b = 0$, run $\mathsf{PKE.Enc}(pars, M, pk)$, obtain ciphertext C and output it.

$\mathsf{PKE.Dec}_4(pars, b||pk, C, sk)$: If $b = 1$ return $\perp$. Else, output $\mathsf{PKE.Dec}(pars, pk, C, sk)$.

To see that Π_4 is not KROB, note that an adversary which outputs $1||pk_0$ and $1||pk_1$, for two valid public keys pk_1 and pk_0 wins the KROB game (for any pair of messages and any pair of random coins) as the resulting ciphertext in both cases is $C^\star$.

In order to show that Π_4 is still FROB, suppose an adversary $\mathcal{A}$ on input $pars$ outputs a winning tuple $(C, b_0||pk_0, b_1||pk_1, sk_0, sk_1)$, where $b_0||pk_0$ and $b_1||pk_1$ are two distinct public keys. Note it must be the case that $b_0 = b_1 = 0$, as otherwise $\mathcal{A}$ cannot win the FROB game. Therefore it is necessarily the case that $pk_0 \neq pk_1$, and $\mathcal{B}$ can win its FROB game against Π by outputting $(C, pk_0, pk_1, sk_0, sk_1)$. □

The separations we provided were crafted in order to systematically relate the new notions of robustness to one another. We next show an example of a more natural separating example between SROB-CCA and the stronger notions of FROB and KROB.

3.3.4 KD* is neither FROB nor KROB

We provide a separating example between the existing notion of strong robustness and the newly defined full and key-less robustness. Indeed we show that a variant of the Kurosawa-Desmedt (KD) encryption scheme [60], proved strongly robust in [62], achieves neither full nor key-less robustness. We recall the KD cryptosystem.

KD encryption scheme

The common public parameters consist of a group $\mathbb{G}$ of prime order $p > 2^\lambda$, with generators $g_1, g_2 \leftarrow \mathbb{G}$. They also include the description of a universal one-way hash function $H : \{0,1\}^* \to \mathbb{Z}_p$, a key derivation function $\mathsf{KDF} : \mathbb{G} \to \{0,1\}^k$, for some integer $k \in \mathsf{poly}(1^\lambda)$ and a symmetric authenticated encryption scheme (E, D) of key length k.

$\mathsf{KeyGen}(pars)$: Given common public parameters $pars = (\mathbb{G}, g_1, g_2, H, \mathsf{KDF}, (\mathsf{E}, \mathsf{D}))$, choose random exponents $x_1, x_2, y_1, y_2 \leftarrow \mathbb{Z}_p$ and compute

$$e = g_1^{x_1} g_2^{x_2}, \qquad f = g_1^{y_1} g_2^{y_2}$$

The public key is $pk = (e, f)$ and the private key is $sk = (x_1, x_2, y_1, y_2)$.

$\mathsf{Enc}(pars, M, pk)$: To encrypt a message $M \in \mathbb{G}$,

1. Pick $u \leftarrow \mathbb{Z}_p$ and compute

$$a_1 = g_1^u, \qquad a_2 = g_2^u, \qquad d = (e \cdot f^v)^u,$$

where $v = H(a_1, a_2) \in \mathbb{Z}_p$.

2. Compute $K = \mathsf{KDF}(d) \in \{0,1\}^k$, $c = \mathsf{E}_K(M)$.

The ciphertext is $C = (a_1, a_2, c)$.

$\mathsf{Dec}(pars, pk, C, sk)$: Parse the ciphertext C as (a_1, a_2, c). Compute $v = H(a_1, a_2)$, $d = a_1^{x_1 + v \cdot y_1} \cdot a_2^{x_2 + v \cdot y_2}$ and $K = \mathsf{KDF}(d) \in \{0,1\}^k$. Then, return $M = \mathsf{D}_K(c)$ (which may be $\perp$ if c fails to properly decrypt under the key K).

The above algorithms describe the original Kurosawa–Desmedt encryption scheme. Following [2], we denote by KD* the modified KD scheme where the encryption exponent $u = 0$ is explicitly disallowed: namely, the sender chooses $u \leftarrow \mathbb{Z}_p^*$ (instead of $u \leftarrow \mathbb{Z}_p$) during encryption and the receiver outputs $\perp$ upon receiving a ciphertext (a_1, a_2, c) such that $a_1 = 1_{\mathbb{G}}$. In [62] it is proven that KD* is strongly robust (with some conditions on the symmetric components).

We will next see that KD* is *not* fully robust. We construct an adversary $\mathcal{A}$ which gets as input $pars$, picks $M \leftarrow \mathbb{G}$ and $u, \alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p^*$. It then computes

$$a_1 = g_1^u, \qquad a_2 = g_2^u, \qquad v = H(a_1, a_2), \qquad d = a_1^{\alpha_1} a_2^{\alpha_2}.$$

If $v = 0$, $\mathcal{A}$ re-samples and re-computes the values, until $v \neq 0$. Now consider the following system of linear equations

$$\begin{cases} x_1 + v y_1 = \alpha_1 \mod p \\ x_2 + v y_2 = \alpha_2 \mod p \end{cases}$$

for some α_1, α_2 in $\mathbb{Z}_p$. Let $(x_{10}, x_{20}, y_{10}, y_{20})$ and $(x_{11}, x_{21}, y_{11}, y_{21})$ be two *distinct* integer solutions to the system.

Now $\mathcal{A}$ sets $pk = (e, f)$ and $pk' = (e', f')$ where

$$e = g_1^{x_{10}} g_2^{x_{20}}, \qquad f = g_1^{y_{10}} g_2^{y_{20}},$$

$$e' = g_1^{x_{11}} g_2^{x_{21}}, \qquad f' = g_1^{y_{11}} g_2^{y_{21}},$$

i.e., the public keys corresponding to secret keys $sk = (x_{10}, x_{20}, y_{10}, y_{20})$ and $sk' = (x_{11}, x_{21}, y_{11}, y_{21})$, respectively. $\mathcal{A}$ finally computes $d = a_1^{\alpha_1} a_2^{\alpha_2}$, $K = \mathsf{KDF}(d)$ and $c = \mathsf{E}_K(M)$. Let $C = (a_1, a_2, c)$.

$\mathcal{A}$'s output for the FROB game will be (C, pk, pk', sk, sk'). Now, $pk \neq pk'$ and by the choice of sk and sk' it is clear that C, decrypted under both secret keys, will return a valid message M with overwhelming probability. It is easy to see that this same strategy allows $\mathcal{A}$ to win also the KROB game. $\mathcal{A}$'s output will simply be

(M, M, pk, pk', u, u), where $pk \neq pk'$ and $\mathsf{Enc}(pars, M, pk; u) = \mathsf{Enc}(pars, M, pk'; u)$ by construction.

We therefore have an attack against the full and key-less robustness of KD*, providing a natural separation between these notions and that of strong robustness.

We note that in a similar way we can show that CS* is neither fully robust nor key-less robust. In fact, we can view the auctioneer in the attack in Section 3.2.1 as the adversary which is allowed to maliciously generate keys and therefore, using the same strategy, it can come up with winning outputs for both the FROB and the KROB games.

3.4 A Unified Approach: Complete Robustness

3.4.1 Complete robustness

At this point it can be asked if there are attacks which fall outside the FROB/KROB model. To answer this question, we take a somewhat different approach towards robustness and view it in terms of the behaviour of the encryption and decryption routines of a scheme with respect to each other. In fact, this is the underlying intuition behind not only the original weak robustness notion (which disappears in the SROB game because the adversary outputs ciphertexts), but also the standard correctness property for a PKE scheme (albeit for a single key). This approach leads to a new notion which we term **complete robustness** (CROB). In this notion the shared parameters of the system are passed to an adversary, which then arbitrarily interacts with the encryption and decryption routines on plaintexts, ciphertexts, keys, and even random coins of its choice. The adversary's goal is to find an "unexpected collision" in the cryptosystem (i.e., one outside the natural restrictions imposed by correctness). We formalize the CROB notion next.

Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a PKE scheme. Consider the following game.

CROB security game

Setup. The challenger $\mathcal{C}$ runs $\mathsf{PKE.PG}(1^\lambda)$ to generate the common parameters $pars$ and initializes a list L to the empty list. $\mathcal{C}$ passes $pars$ on to $\mathcal{A}$.

List building phase. $\mathcal{A}$ can make the following queries.

- **Encryption query:** $\mathcal{A}$ gives a public key pk, a message M and randomness r to $\mathcal{C}$ and obtains $C = \mathsf{PKE.Enc}(pars, M, pk; r)$. $\mathcal{C}$ adds to the list L the tuple $(pk, M, C, \perp, r)$.
- **Decryption query:** $\mathcal{A}$ gives a public key pk, a secret key sk and a ciphertext C to $\mathcal{C}$ and obtains $M = \mathsf{PKE.Dec}(pars, pk, C, sk)$. $\mathcal{C}$ adds to the list L the tuple $(pk, M, C, sk, \perp)$.

$\mathcal{A}$ is a winning adversary if there exist two tuples $(pk_0, M_0, C_0, e_{01}, e_{02}), (pk_1, M_1, C_1, e_{11}, e_{12})$ in list L, where $(e_{i1}, e_{i2}) \in \{(\perp, r_i), (sk_i, \perp)\}$ for $i \in \{0, 1\}$, such that:

1. $pk_0 \neq pk_1$;
2. $M_0 \neq \perp \ \wedge M_1 \neq \perp$; and
3. $C_0 = C_1$.

We define $\mathcal{A}$'s advantage as the probability of outputting two tuples $(pk_0, M_0, C_0, e_{01}, e_{02}), (pk_1, M_1, C_1, e_{11}, e_{12})$ satisfying all of the above conditions.

Definition 3.10 (CROB) *A PKE scheme* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is* completely robust *(or CROB) if all polynomial-time adversaries have at most negligible advantage in the above game.*

It can be seen through an easy inspection that full robustness is a sub-case of complete robustness and it can be viewed as the "decryption component" of the above definition. Key-less robustness, its *dual*, can similarly be viewed as the encryption component and therefore it is also implied by complete robustness.

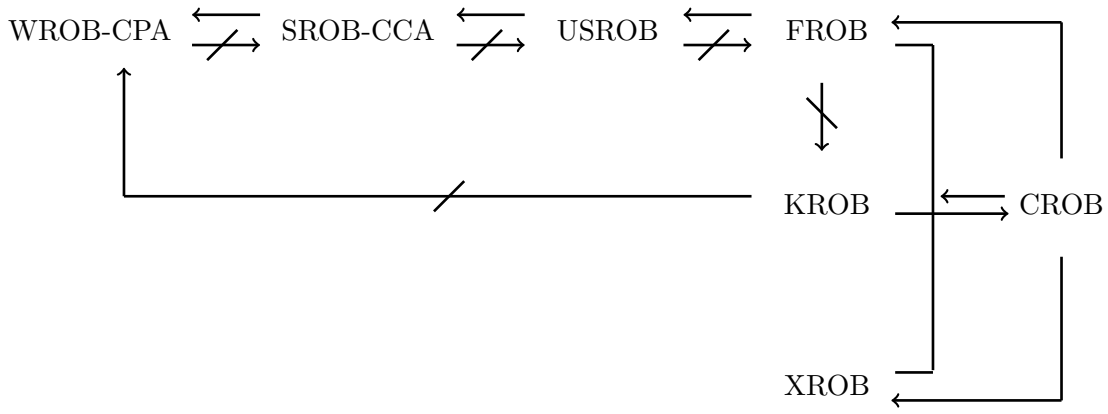

Figure 3.2: Relations among notions of robustness.

3.4.2 Relations among notions of robustness

Let us now see where CROB stands in relation to the other notions. For instance, it may be asked if FROB and KROB are strong enough together to jointly imply CROB. We show this is *not* the case. To this end, we first characterize CROB in terms of *three* notions of robustness consisting of FROB, KROB and a new *mixed* notion which we call XROB. We then show that XROB is necessary in the sense that it is not always implied by FROB and KROB put together. Figure 3.2 summarizes the main relations among notions of robustness we establish in our work.

Before proving our results we formally define the XROB game. Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a PKE scheme. Consider the following game.

XROB security game

Setup. The challenger $\mathcal{C}$ runs $\mathsf{PKE.PG}(1^\lambda)$ to generate the common parameters $pars$ and passes them on to $\mathcal{A}$.

Finalize. $\mathcal{A}$ outputs $((M_0, pk_0, r_0), (C_1, pk_1, sk_1))$.

$\mathcal{A}$ is a winning adversary if its output satisfies the following conditions:

1. $pk_0 \neq pk_1$;

2. $\mathsf{PKE.Enc}(pars, M_0, pk_0; r_0) = C_1 \wedge (M_0 \neq \perp) \wedge \mathsf{PKE.Dec}(pars, pk_1, C_1, sk_1) \neq \perp$.

We define $\mathcal{A}$'s advantage as the probability of outputting $((M_0, pk_0, r_0), (C_1, pk_1, sk_1))$ satisfying all of the above conditions.

Definition 3.11 (XROB) *A PKE scheme* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is XROB if all polynomial-time adversaries have at most negligible advantage in the above game.*

The following results hold.

Proposition 3.12 (CROB $\Leftrightarrow$ FROB $\wedge$ KROB $\wedge$ XROB) *A PKE scheme is CROB if and only if it is simultaneously FROB, KROB and XROB.*

Proof. A pair of winning tuples can arise in one of three possible ways:

- Both tuples have a secret key as their fourth entry (and therefore a $\perp$ as their last entry), meaning they were added to list L as decryption queries. These can be translated into a winning output for a FROB adversary.
- Both tuples have the encryption randomness as their last entry (and therefore a $\perp$ as their fourth entry), meaning they were added to list L as encryption queries. These can be translated into a winning output for a KROB adversary.
- One tuple has $\perp$ as its fourth entry and the other has it as its last. These tuples can be translated into a winning output for an XROB adversary.

□

Proposition 3.13 (FROB $\wedge$ KROB $\nrightarrow$ CROB) *Let* $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *be FROB and KROB. Then there is a scheme* $\Pi_5 = (\mathsf{PKE.PG}_5, \mathsf{PKE.KeyGen}_5, \mathsf{PKE.Enc}_5, \mathsf{PKE.Dec}_5)$ *which is FROB and KROB but fails to be CROB.*

Proof. We define the required scheme Π_5 as follows.

$\mathsf{PKE.PG}_5(1^\lambda)$: Run $\mathsf{PKE.PG}(1^\lambda)$ to obtain $pars'$. Run $\mathsf{PKE.KeyGen}(pars')$ to obtain $(pk^\star, sk^\star)$. Let $M^\star$ and $C^\star$ be, respectively, a fixed message and a fixed ciphertext corresponding to $pk^\star$. Return $pars := (pars', pk^\star, C^\star, M^\star)$.

$\mathsf{PKE.KeyGen}_5(pars)$: Run $\mathsf{PKE.KeyGen}(pars')$ to obtain (pk, sk). Return $(0||pk, sk)$.

$\mathsf{PKE.Enc}_5(pars, M, b||pk; r)$: If $b = 0$ return $0||\mathsf{PKE.Enc}(pars', M, pk; r)$. If $b = 1$ and $pk = pk^\star$, output $1||C^\star$. Else return $\perp$.

$\mathsf{PKE.Dec}_5(pars, b||pk, c||C, sk)$: If $b \neq c$ return $\perp$. If $b = 0$ return $\mathsf{PKE.Dec}(pars', C, sk)$. If $b = 1$, $pk = pk^\star \oplus 1$ and $C = C^\star$ return $M^\star$. Else return $\perp$.

We first show that Π_5 is not XROB (and hence it is also not CROB). We construct an XROB adversary $\mathcal{A}$ as follows. Algorithm $\mathcal{A}$ obtains $pars$ and sets $r = sk = 0$. It then returns the tuples $(M^\star, 1||pk^\star; r)$ and $(1||C^\star, 1||(pk^\star \oplus 1), sk)$. Now

$$\mathsf{PKE.Enc}_5(pars, M^\star, 1||pk^\star; r) = 1||C^\star$$

and

$$\mathsf{PKE.Dec}_5(pars, 1||(pk^\star \oplus 1), sk, 1||C^\star) = M^\star.$$

Furthermore, $1||pk^\star \neq 1||(pk^\star \oplus 1)$, $M^\star \neq \perp$, and $C^\star \neq \perp$. Therefore $\mathcal{A}$ wins the XROB game with probability 1.

We now show Π_5 is still FROB. Take any FROB adversary $\mathcal{A}$ against Π_5. We construct a FROB adversary $\mathcal{B}$ against Π as follows. $\mathcal{B}$ runs on $pars'$ and provides $\mathcal{A}$ with $pars$ based on $pars'$ as in the description of the scheme. Now if the public keys that $\mathcal{A}$ outputs begin with different bits, since $\mathsf{PKE.Dec}_5$ checks if $b \neq c$, it must be the case that the ciphertexts also begin with different bits, and hence $\mathcal{A}$ will not win the FROB game. Suppose $\mathcal{A}$ returns two public keys beginning with $b = 0$. It is clear that in this case $\mathcal{B}$ can also win its FROB game by stripping away the redundant bits. Finally, suppose both beginning bits are $b = 1$. In this case $\mathsf{PKE.Dec}_5$ will return a non-$\perp$ value only when the public keys are both $1||(pk^\star \oplus 1)$, and hence $\mathcal{A}$ cannot win its game in this case.

It remains to show Π_5 is KROB. Take any KROB adversary $\mathcal{A}$ against Π_5. We construct a KROB adversary $\mathcal{B}$ against Π as follows. $\mathcal{B}$ receives $pars'$ and generates $pars$ for Π_5 as in description of $\mathsf{PKE.PG_5}$ above. Now if the public keys that $\mathcal{A}$ outputs begin with different bits, the ciphertexts will not be colliding as the encryption algorithm attaches the first bit of the public key to the ciphertext. If $\mathcal{A}$ outputs a winning pair with pk_0 and pk_1 starting with $b = 0$, it is clear that $\mathcal{B}$ can break the KROB property of the underlying scheme by removing the redundant bits from the public keys. If both public keys start with $b = 1$ (and the ciphertexts are not $\perp$), and $\mathcal{A}$ is winning, then it must be the case that the public keys are equal, and so this case cannot lead to $\mathcal{A}$ winning the KROB game. □

Having defined complete robustness, CROB, and its weaker relatives, we now want to prove that it is achievable using generic constructions. In particular, we show that the construction for strong robustness presented in [2] (which we call the ABN transformation for short) is actually so powerful as to also achieve CROB.

3.5 Generic Constructions for Complete Robustness

3.5.1 The ABN transformation

In [2] the authors give a generic transformation which takes a scheme Π that satisfies IND-IK-ATK security (where ATK $\in \{\text{CPA}, \text{CCA}\}$), and outputs a scheme $\overline{\Pi}$ that preserves IND-IK-ATK security but is also strongly robust.

Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a PKE scheme and let $CMT = (\mathsf{CPG}, \mathsf{Com}, \mathsf{Vrfy})$ be a commitment scheme. $\overline{\Pi} = (\overline{\mathsf{PKE.PG}}, \overline{\mathsf{PKE.KeyGen}}, \overline{\mathsf{PKE.Enc}}, \overline{\mathsf{PKE.Dec}})$ is constructed as follows [2].

$\overline{\mathsf{PKE.PG}}(1^\lambda)$: Run $\mathsf{PKE.PG}$ and CPG on input 1^λ to obtain $pars$ and $cpars$. Return $pars' = (pars, cpars)$.

$\overline{\mathsf{PKE.KeyGen}}(pars')$: Run $\mathsf{PKE.KeyGen}(pars)$ to obtain (pk, sk).

$\overline{\mathsf{PKE.Enc}}(pars', M, pk)$: Generate $(com, dec) \leftarrow \mathsf{Com}(cpars, pk)$. Run $\mathsf{PKE.Enc}(pars, M||dec, pk)$ to obtain ciphertext c. Return $C = (c, com)$.

$\overline{\mathsf{PKE.Dec}}(pars', pk, C, sk)$: Parse C as (c, com). Run $\mathsf{PKE.Dec}(pars, c, sk)$ and obtain M'. If $M' = \perp$ then return $\perp$. Otherwise, parse M' as $M||dec$, for dec of the appropriate length. If $\mathsf{Vrfy}(cpars, pk, com, dec) = 1$ then return M, otherwise return $\perp$.

Abdalla et al. prove that this transformation preserves the IND-IK-ATK security of Π if Π is also WROB ([2, Theorem 4.2, part 1]), and that $\overline{\Pi}$ is also strongly robust ([2, Theorem 4.2, part 2]). We re-use the first part of this result but strengthen its second part by showing that the transformation confers *complete* robustness.

Theorem 3.14 *If CMT is binding, the ABN transformation results in a CROB scheme.*

Proof. We treat the three possible cases corresponding to FROB, KROB and XROB.

Given an FROB adversary $\mathcal{A}$ against $\overline{\Pi}$ we construct an adversary $\mathcal{B}_1$ that will interact with $\mathcal{A}$ to break the binding property of CMT. The game is as follows.

Let $\mathcal{C}$ be $\mathcal{B}_1$'s challenger. $\mathcal{C}$ runs CPG to obtain the commitment scheme's parameters $cpars$ and passes them on to $\mathcal{B}_1$. $\mathcal{B}_1$ runs $\mathsf{PKE.PG}$ to obtain $pars$, which it passes to $\mathcal{A}$ together with $cpars$.
Finally, $\mathcal{A}$ outputs $(C, pk_0, pk_1, sk_0, sk_1)$, where $C = (c, com)$ and $pk_0 \neq pk_1$. Now $\mathcal{B}_1$ runs $\mathsf{PKE.Dec}(pars, pk_0, c, sk_0)$, obtaining M_0, and $\mathsf{PKE.Dec}(pars, pk_1, c, sk_1)$, obtaining M_1. Let Succ be the event that neither M_0 nor M_1 are $\perp$. If Succ occurs then $\mathcal{B}_1$ parses M_0 and M_1 into $\tilde{M_0}||dec_0$ and $\tilde{M_1}||dec_1$, respectively. It then gives $(com, pk_0, pk_1, dec_0, dec_1)$ to $\mathcal{C}$ as its final output.
$\mathcal{B}_1$ provides a perfect simulation for $\mathcal{A}$ as well as a legal strategy for attacking the binding property of CMT, provided Succ occurs. Since this happens whenever $\mathcal{A}$ is a winning adversary against the full robustness of $\overline{\Pi}$, we have that $\mathcal{B}_1$'s advantage is the same as $\mathcal{A}$'s.

Given a KROB adversary $\mathcal{A}$ against $\overline{\Pi}$ we construct an adversary $\mathcal{B}_2$ that will interact with $\mathcal{A}$ to break the binding property of CMT. The game is as follows.

Let $\mathcal{C}$ be $\mathcal{B}_2$'s challenger. $\mathcal{C}$ runs CPG to obtain the commitment scheme's parameters $cpars$ and passes them on to $\mathcal{B}_2$. $\mathcal{B}_2$ runs $\mathsf{PKE.PG}$ to obtain $pars$, which it passes to $\mathcal{A}$ together with $cpars$.
$\mathcal{A}$ outputs $(M_0, M_1, pk_0, pk_1, r_0, r_1)$, where $pk_0 \neq pk_1$, $r_0 = (r_{0_{cmt}}, r_{0_{enc}})$ and $r_1 = (r_{1_{cmt}}, r_{1_{enc}})$. Now $\mathcal{B}_2$ runs $\mathsf{Com}(cpars, pk_0; r_{0_{cmt}})$, obtaining (com_0, dec_0), and $\mathsf{Com}(cpars, pk_1; r_{1_{cmt}})$, obtaining (com_1, dec_1). $\mathcal{B}_2$ computes $c_0 = \mathsf{PKE.Enc}(pars, M_0||dec_0, pk_0; r_{0_{enc}})$ and $c_1 = \mathsf{PKE.Enc}(pars, M_1||dec_1, pk_1; r_{1_{enc}})$. Let $C_0 = (c_0, com_0)$ and $C_1 = (c_1, com_1)$. Let Succ be the event that $C_0 = C_1$ (and therefore $c_0 = c_1 = c$ and $com_0 = com_1 = com$). If Succ occurs then $\mathcal{B}_2$ outputs $(com, pk_0, pk_1, dec_0, dec_1)$ and gives it to $\mathcal{C}$.
$\mathcal{B}_2$ provides a perfect simulation for $\mathcal{A}$ as well as a legal strategy for attacking the binding property of CMT, provided Succ occurs. Since this happens whenever $\mathcal{A}$ is a winning adversary against the key-less robustness of $\overline{\Pi}$, we have that $\mathcal{B}_2$'s advantage is the same as $\mathcal{A}$'s.

Given an XROB adversary $\mathcal{A}$ against $\overline{\Pi}$ we construct an adversary $\mathcal{B}_3$ that will interact with $\mathcal{A}$ to break the binding property of CMT. The game proceeds as follows.

Let $\mathcal{C}$ be $\mathcal{B}_3$'s challenger. $\mathcal{C}$ runs CPG to obtain the commitment schemes's parameters $cpars$ and passes them on to $\mathcal{B}_3$. $\mathcal{B}_3$ runs $\mathsf{PKE.PG}$ to obtain $pars$, which it passes to $\mathcal{A}$ together with $cpars$.
$\mathcal{A}$ outputs $(M_0, pk_0, r_0, C_1, pk_1, sk_1)$, where $pk_0 \neq pk_1$, $r_0 = (r_{0_{cmt}}, r_{0_{enc}})$ and $C_1 = (c_1, com_1)$. Now $\mathcal{B}_3$ runs $\mathsf{Com}(cpars, pk_0; r_{0_{cmt}})$, in order to obtain (com_0, dec_0), and it also runs $\mathsf{PKE.Dec}(pars, pk_1, c_1, sk_1)$, obtaining M_1. Then $\mathcal{B}_3$ computes the ciphertext $c_0 = \mathsf{PKE.Enc}(pars, M_0||dec_0, pk_0; r_{0_{enc}})$. Let $C_0 = (c_0, com_0)$. Let Succ be the event that $C_0 = C_1$ (and therefore $c_0 = c_1 = c$ and $com_0 = com_1 = com$). If Succ occurs then $\mathcal{B}_3$ outputs $(com, pk_0, pk_1, dec_0, dec_1)$ and gives it to $\mathcal{C}$.
$\mathcal{B}_3$ provides a perfect simulation for $\mathcal{A}$ as well as a legal strategy for attacking the binding property of CMT, provided Succ occurs. Since this happens whenever $\mathcal{A}$ is a winning adversary against the key-less robustness of $\overline{\Pi}$, we have that $\mathcal{B}_3$'s advantage is the same as $\mathcal{A}$'s. □

3.5.2 Further constructions

We have presented a generic construction for complete robustness. We now provide an overview of alternative ways to achieve this notion. Details of some of the following results can be found in [45].

3.5.2.1 Mohassel's transformation

Mohassel [66] provides a generic transformation in the random oracle model that converts an IND-IK-ATK encryption scheme into one which is SROB-CCA, without compromising its IND-IK-ATK security. In his construction, the hash value $H(pk, r, M)$, where r is the randomness used in the encryption, is attached to ciphertexts. This immediately rules out all forms of collisions between ciphertexts, as the hash values are unlikely to collide on two distinct public keys. It is then easy to see that this construction also achieves complete robustness.

3.5.2.2 A modified ABN transform

While the original transformation [2] *does* preserve IND-IK-CCA security and confers CROB security, the IND-IK-CCA security of the transformed scheme $\overline{\Pi}$ relies on the weak robustness of the underlying encryption scheme Π. It is possible to show that if the underlying encryption scheme supports *labels* (in which case the encryption and decryption algorithms both take an additional public string L as input), this assumption can be eliminated and we only need Π to be IND-IK-CCA secure. Although the weak robustness requirement is not too demanding in theory (since any encryption scheme can be made weakly robust by means of a keyed redundancy-based transformation [2]), this construction provides better efficiency in cases where the IND-IK-CCA encryption scheme natively supports labels (such as for the Cramer–Shoup and Kurosawa–Desmedt schemes). Details of this approach can be found in [45].

3.5.2.3 Complete robustness from IBE

In [45] the authors also answer in a positive sense a question left open in [2] as to whether the Canetti–Halevi–Katz [25] (CHK) paradigm can be leveraged so as to obtain schemes that simultaneously offer IND-IK-CCA security and robustness in a strong sense. Answering this question is non-trivial: Abdalla et al. pinpointed that applying the one-time-signature-based CHK transformation to the Boyen–Waters IBE [22], for example, does not provide SROB-CCA or even SROB-CPA security. The construction in [45] is a variant of the Boneh–Katz construction [19] for chosen-ciphertext security, and it requires the underlying IBE to only satisfy a weak level of security under chosen-plaintext attacks. Because this approach simultaneously provides complete robustness *and* IND-IK-CCA security, it results in schemes having better efficiency than what would be obtained by applying the ABN transformation of [2] to an IND-IK-CCA secure scheme obtained from the original Boneh–Katz transformation.

3.5.2.4 Concrete schemes

It is natural to ask whether it is possible to improve upon the efficiency of generic constructions with concrete schemes whose security rests on specific computational assumptions. Indeed, this is the case, and another achievement in this area of research is to directly construct a CROB and IND-IK-CCA secure scheme using, as a starting point, certain hybrid encryption systems, such as the Hofheinz–Kiltz [55] and the Kurosawa–Desmedt [60] schemes. This is developed in detail in [45].

3.5.3 Conclusions

We have developed new notions of robustness in the public-key setting, identifying complete robustness as the strongest one. We have also explored the relationship between these notions. Furthermore, we have shown how to generically achieve public-key encryption schemes which are completely robust, and provided an overview of further results in this area of research.

As a next step, it would be interesting to consider the newly introduced notions of robustness in the identity-based setting. In such a context, the natural extension of our notions would be to allow the adversary to choose the IBE master keys maliciously. It would also be interesting to explore the significance of robustness in more advanced primitives such as attribute-based encryption and predicate encryption. We leave further development of these ideas as future work.

Chapter 4

Anonymous Broadcast Encryption

Contents

In this chapter we consider anonymity in the context of broadcast encryption. We provide a security definition for anonymous broadcast encryption (ANOBE) and show that it is achievable from public-key, identity-based and attribute-based encryption, providing secure constructions from all of these primitives. Furthermore, we show how randomness re-use techniques can be deployed in the ANOBE context to reduce computational and communication costs. All of our results are in the standard

model, achieving fully collusion-resistant ANOBE schemes secure against adaptive IND-CCA adversaries. This chapter will appear as part of [62] in the proceedings of the international cryptographic conference Public-Key Cryptography 2012, as joint work with Benoît Libert and Kenneth G. Paterson.

4.1 Introduction

4.1.1 Broadcast encryption

Broadcast encryption (BE) addresses the issue of broadcasting a message to a dynamically changing privileged subset of a set of users, in a way that no user outside the privileged set can learn the message. We will call the universe of n users U and the privileged (or target) set S, where $S \subseteq U$. Since its introduction in 1993 by Fiat and Naor this area of research has received a lot of attention. This is largely due to the numerous real-world applications BE has, namely pay TV, multicast communication, Internet broadcast, audio streaming and, in general, any secure distribution of copyright-protected content. In this respect, BE represents an essential component offering solutions to several issues these applications may face: maintaining the confidentiality of the message, protecting user privacy, revoking unauthorized users, deploying traitor-tracing mechanisms, etc. Indeed, the BE schemes the cryptographic community has put forth can be seen as building blocks upon which a fully-fledged broadcast system can be developed.

Given BE's inherent practical vocation, the relevant research has progressed in achieving the desired functionalities whilst being extremely alert with respect to the efficiency of the proposed schemes. Since the very beginning, the benchmark for performance comparison has been what can be considered as the *natural* solution to the broadcast problem, that is, encrypt the message repeatedly to each user in the privileged set. This approach, however simple and efficient in terms of key-storage, results in ciphertexts whose size is linear in the size of the privileged set. In the symmetric setting, where the first BE schemes were proposed, the other natural solution is to assign a key to every possible subset of the set of users and encrypt the message with the appropriate key. This would result in constant-size ciphertexts but an exponential number of keys for the user to store (one for each subset he belongs

to). The practicality of this solution is further limited by the fact that for any new user joining the system, the existing users would have to go on-line to update their keys, creating even more key-management issues.

Finding a trade-off between key-storage requirements and ciphertext size was the main concern behind the design of the first (symmetric) BE schemes. These were predominantly presented as *revocation* schemes, allowing to address a large privileged set of receivers and, at each broadcast, to revoke a small number r of unauthorized users. The solutions of [47, 67] employ tree-based techniques to achieve sub-linearity in both the keys and the ciphertext size. To be more precise, [67] presents schemes with $O(\log n)$ key-storage requirements and ciphertexts of size $O(r \log n)$. The results of [67] go even further by providing an elegant framework called the *subset cover* framework, within which *fully collusion-resistant* broadcast encryption schemes for *stateless receivers* can efficiently be obtained. Achieving these properties is yet another advantage of this approach.

A **stateless receiver** is a user who does not need to update his private key, i.e. the key is fixed for the lifetime of the system. This clearly is more practical than having a stateful receiver, who may need a key update each time a new user joins the system or may have to change keys based on previous message transmissions. **Collusion resistance** is one of the fundamental properties of a BE scheme. This is the requirement that no coalition of users outside the privileged set can recover the message. In the literature we can find several schemes that resist collusion attacks mounted by coalitions of at most $t < n$ users, where n is the maximal number of users; only some schemes are *fully* collusion-resistant, i.e., they can tolerate attacks by coalitions of any size.

In our work, we will consider systems that allow *stateless receivers* and are *fully* collusion-resistant. These are by now standard objectives for a BE scheme. Our focus will be on developing schemes in the *public-key* setting, of which we give a brief overview next.

4.1.1.1 Public-key broadcast encryption

The subset cover framework was originally proposed in the symmetric setting. However, [67] suggests a way to extend this to the public-key environment, incurring unfortunately a very large public key. The authors of [43] improve on the efficiency of such extension by employing identity-based techniques which result in constant-size key-storage requirements and similar ciphertext length (linear in r, sublinear in n). We observe that the type of (revocation) schemes in [67, 43] are considered efficient as long as the number of revoked users in the system is small (this is the case in the context of content distribution, for example).

In the years following this work a large number of BE variants have been proposed, examples of which are *dynamic join* [40, 76] and *identity-based broadcast encryption* [39], highlighting the interest of the community in this area of research.

A major breakthrough in BE was the work by Boneh, Gentry and Waters [17] which no longer employed combinatorial techniques but adopted groups with bilinear maps to achieve very efficient public-key BE schemes. More specifically, two schemes are presented in [17]: one obtains *constant-size* private keys and ciphertexts, consisting of 1 and 2 group elements, respectively, but has a public key whose size is linear in n, the total number of users; the second construction can be parametrized so as to have again constant-size private keys, but ciphertexts and public key of size $O(\sqrt{n})$, enabling a trade-off in sizes. This work has been regarded as ground-breaking in terms of performance achieved.

Follow up work by Gentry and Waters [51] presented a BE scheme with similar efficiency, in particular yielding constant-size ciphertexts, but achieving a stronger notion of security, namely against *adaptive* adversaries, which we discuss next.

4.1.1.2 Static vs. adaptive security in BE

Amongst the many challenges BE poses, obtaining schemes which satisfy a strong security notion is one of the latest to have been overcome. In fact, the impressively efficient schemes in [17], dating from 2005, only achieve security against a *static*

adversary. This type of adversary is limited with respect to the set of users he can corrupt. More precisely, he needs to specify *a priori* the target set he wishes to be challenged on. The authors of [17] leave the construction of an efficient BE scheme secure against an *adaptive* adversary, i.e., one that can adaptively corrupt users during the game, as a major open problem. Four years later, Gentry and Waters [51] solved the problem, achieving adaptively secure BE. This stronger notion of security is arguably the correct one for BE since it captures the most general class of attackers who are allowed to see the system's parameters and can arbitrarily corrupt users before committing to a challenge set.

4.1.2 Anonymity in broadcast encryption

As discussed so far, several practical aspects need to be taken into consideration when designing a BE scheme, especially in view of its real-life applications: strength of security notions, public and private storage requirements, ciphertext length, and computational costs. The specific nature of the primitive however has led researchers to focus in particular on solutions having ciphertexts that are as short as possible. In this respect, the results of [17] and [51] are nearly optimal. However, designing BE schemes for real-life applications to broadcasting should not only involve efficiency and confidentiality issues. In particular, the privacy of users should be protected as much as possible. We believe that, to date, this aspect has not been adequately dealt with. Our study of the literature reveals that anonymity in BE has only been considered in a single paper [6], in the context of encrypted file systems. Surprisingly, almost all subsequent work on BE has ignored the issue of anonymity. Moreover, as we shall explain next, state-of-the-art BE schemes are inherently incapable of providing any kind of anonymity.

4.1.2.1 An illustrative example

To illustrate our point, we recall one of the schemes proposed by Gentry and Waters in [51, Section 3.1], which achieves adaptive security and appears to offer very short (constant-size) ciphertexts, namely 3 group elements. We call this scheme GW. As per standard BE schemes (a formal definition of which will be provided in Section

4.2), GW consists of four algorithms: BE.Setup, which on input the security parameter returns the master public key and the master secret key of the system; a key generation algorithm BE.KeyGen; an encryption algorithm BE.Enc, which on input a message and the target set returns a ciphertext, and a decryption algorithm BE.Dec, which returns the message if the secret key used belongs to a member of the target set. More precisely, GW is defined as follows.

The GW broadcast encryption scheme

Let GroupGen be an algorithm that on input a security parameter 1^λ generates $\mathbb{G}$ and $\mathbb{G}_T$, groups of prime order p equipped with a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. We define the algorithms for GW in the following way.

Setup$(1^\lambda, n)$: Run GroupGen(1^λ) to obtain $(\mathbb{G}, \mathbb{G}_T, e)$. Choose random $\alpha \leftarrow \mathbb{Z}_p$ and $g, h_1, ..., h_n \leftarrow \mathbb{G}^{n+1}$. The master public key is

$$\text{BE-MPK} = (\mathbb{G}, \mathbb{G}_T, e, g, e(g,g)^\alpha, h_1, ..., h_n)$$

and the master secret key is BE-MSK $= g^\alpha$.

KeyGen$(i, \text{BE-MSK})$: Choose random $r_i \leftarrow \mathbb{Z}_p$ and output

$$d_i = (d_{i,0}, ..., d_{i,n}) \text{ where } d_{i,0} = g^{-r_i},\ d_{i,i} = g^\alpha h_i^{r_i},\ \forall_{j \neq i}\ d_{i,j} = h_j^{r_i}.$$

Enc$(\text{BE-MPK}, M, S)$: Select $t \leftarrow \mathbb{Z}_p$ and set

$$C_1 = g^t,\ C_2 = (\prod_{j \in S} h_j)^t,\ C_3 = M \cdot e(g,g)^{\alpha \cdot t}.$$

Output $C = (C_1, C_2, C_3)$.

Dec$(\text{BE-MPK}, C, i, d_i, S)$: If $i \in S$, parse d_i as $(d_{i,0}, ..., d_{i,n})$ and C as (C_1, C_2, C_3). Compute

$$C' = e(d_{i,i} \cdot \prod_{j \in S \setminus \{i\}} d_{i,j}, C_1) \cdot e(d_{i,0}, C_2).$$

Compute $M = C_3 / C'$.

The scheme is proved correct and secure in [51].

As we would expect from any public-key BE scheme, each user in the system can obtain his private key from the BE.KeyGen algorithm, and the sender can choose an *arbitrary* target set of users S to which he wishes to broadcast a message. A closer look at the decryption algorithm, however, reveals something rather peculiar: to recover the message, a legitimate user, i.e., a user in S, has to run the decryption algorithm on input the ciphertext, his private key *and* a description of the target set S. This set S is required specifically as an input to BE.Dec in the existing definitions of BE [51, 18, 39]. We note that, as is evident from our illustrative example, this is not just a formality issue which could be solved simply by removing this requirement from the BE model. Current schemes such as GW explicitly *rely* on S as an input to BE.Dec for decryption to work. Therefore the user needs to somehow know to which set S the message was broadcast, otherwise he cannot decrypt.

This simple but crucial observation raises a fundamental question: where does S come from? In the most general usage scenario intended for BE, where S is dynamic and may be unpredictable from message to message, the ciphertexts must effectively include a description of S as part of the ciphertexts themselves. This leads to two considerations. Firstly, current BE schemes such as those in [51, 17, 39] do not account for the cost of broadcasting a description of S when calculating the size of ciphertexts. This means that the true ciphertext size in these schemes is **linear** in n rather than constant-size, as a cursory examination of the schemes might suggest[1]. However, we say that the results in [17] and [51] are nearly optimal (having ciphertexts of size n bits plus a constant number of group elements) since there is a simple counting argument showing that, for a universe of n users in which every possible subset S should be reachable by secure broadcast, ciphertexts must contain at least n bits. Indeed the ciphertext should be long enough to uniquely identify the privileged subset S. The overhead provided by these schemes is therefore impressively small, but, to repeat, the true ciphertext size is linear in n.

The second important consideration that comes from noticing S as a required input to the decryption algorithm is that this limitation in the existing BE model and schemes clearly causes serious **privacy issues**. Imagine we deploy a BE scheme, as defined above, for television broadcasting. Suppose the privileged set is the set of

[1] This does not rule the use of compact encodings of S being transmitted with ciphertexts in more restrictive usage scenarios, for example, only sending the difference in S when the set S changes only slowly from message to message.

all users who have paid a subscription to a certain channel. Each customer should have access to that channel using his private key. The problem is that, to decrypt, he will have to know who *else* has paid for the specific subscription! Not only is this requirement very inconvenient for the practical deployment of BE schemes, it is also a severe violation of the individual subscriber's privacy. Ideally, a BE scheme should protect users' privacy by guaranteeing that ciphertexts do not leak any information about the privileged set S. This is exactly what we mean for a BE scheme to be *anonymous* and we view this as a highly desirable security property for BE in practical applications.

There are many reasons why ensuring anonymity should be a top priority when designing a BE scheme. First of all, knowledge of the target set of receivers of a certain message may be more sensitive than the message itself. Furthermore, in most usage scenarios, hiding the identities of the privileged users (even from each other) is essential to guaranteeing the adequate level of security expected from the application (pay TV, Internet broadcast, etc.). It is very surprising that this problem has been so overlooked throughout these years.

4.1.2.2 Related work

The only prior work addressing the issue of anonymity in BE appears to be that of Barth et al. [6] (there, it is called *privacy*). In [6] the authors highlight the need for user privacy in certain applications of BE, such as encrypted file systems and content delivery systems. To address this they introduce the notion of private broadcast encryption and define a security model for recipient privacy. This model suffers from the limitation that only *static* adversaries are considered. Indeed, the two generic constructions for private broadcast encryption presented in [6] are proved secure only against this less powerful type of adversary. The first construction uses a key-private, IND-CCA secure PKE scheme as a base to encrypt the message to each user in the target set, and then ties together the resulting ciphertexts using a strongly secure one-time signature. This yields a private BE scheme that is secure in the standard model. A drawback of this approach, however, is that decryption is linear in the size of the target set S, as an intended recipient needs to perform an average of $|S|/2$ decryptions before recovering the message. The second construction

is a modification of the first one which allows for efficient decryption. The basic idea is to append a tag to each ciphertext component so that the legitimate user can recompute the tag with his private key and then identify his corresponding ciphertext component. The technique presented in [6] to speed-up decryption was, however, only analyzed in the random oracle model.

In [21] the authors provide a private linear broadcast encryption (PLBE) scheme to realize a fully collusion-resistant traitor-tracing scheme. A PLBE, however, is a BE system with limited capabilities (i.e. it cannot address arbitrary sets of users) and hence this work does not provide a solution to the problem considered so far.

In very recent work [46] that builds on [6] and [62], the authors introduce the notion of *outsider*-anonymous broadcast encryption, which is designed to hide the privileged set from any outsider but provides no privacy guarantees with respect to the legitimate users. The claim (in [46]) that this notion is justified since the content of the communication already reveals something about the recipient set is highly debatable, and certainly does not suit our motivation for the study of anonymity in BE. This primitive, however artificial it may seem, can be achieved using tree-based techniques from [67, 43], yielding schemes whose ciphertexts have similar (compact) size, i.e. linear in the number of revoked users.

4.1.3 Our contributions

We start by giving a unified security definition for anonymous broadcast encryption (ANOBE) in Section 4.2. Instead of separating anonymity and confidentiality as in [6], we use a combined security notion for ANOBE which helps to streamline our presentation and proofs. In addition, we strengthen the model to allow the adversary to make *adaptive* corruptions, *with all of our constructions achieving security in this setting*. In contrast, the definition of [6] is static, requiring the adversary to choose whom to corrupt before seeing the public key of the system. As a first step we show in Section 4.3.1 that our enhanced security definition is indeed satisfiable: adaptively secure ANOBE can be built based only on the existence of IND-CCA secure PKE (without requiring the base PKE scheme to have *any* anonymity properties itself). This construction results in a very efficient (constant) decryption procedure but has

ciphertexts whose size is linear in n, the number of users in the universe U.

In Section 4.3.2 we show that the generic construction for ANOBE suggested by Barth et al. [6] actually possesses adaptive security, and not merely static security as was established in [6]. This construction starts from any weakly robust (in the sense of [2]), key-private PKE scheme with chosen-ciphertext security. In comparison with our first generic construction, this result imposes stronger requirements on the underlying encryption scheme. However, it achieves shorter ciphertexts, with the size being linear in the size of the target set S.

In Section 4.4 we provide another generic construction which uses an identity-based encryption (IBE) scheme having suitable security properties, in the style of the CHK transformation (see Section 2.4.2). This alternative further increases the set of components that can be used to obtain ANOBE.

With the aim of setting the ground for a systematic study of ANOBE and its relations to other important primitives, we explore the interesting connection between attribute-based encryption (ABE) and BE. In Section 4.5 we prove that ANOBE can be securely achieved from ABE as well.

Having so far demonstrated the achievability of ANOBE, we next focus on improving the performance of the resulting schemes. To this end, we show how randomness re-use techniques originally developed for PKE in [59, 10, 9] can be modified for secure deployment in the ANOBE setting. In particular, we identify a slightly stronger notion of reproducibility that we call *key-less reproducibility*. We show in Section 4.6 that if our base PKE scheme has this property (in addition to the other properties needed in our generic construction) then it can be used with the same randomness across all ciphertext components in our main ANOBE construction. This not only allows the size of ciphertexts to be reduced further (by eliminating repeated ciphertext elements) but also reduces the sender's computational overhead.

We conclude the chapter by briefly presenting further results on ANOBE and by giving possible directions for future work.

4.2 Anonymous Broadcast Encryption

In this section we define a model for public-key broadcast encryption (BE), where algorithms are specified to allow for anonymity (similarly to [6]) and they are general enough to include the identity-based variant of BE introduced in [39].

Definition 4.1 (Broadcast encryption scheme) *Let $U = \{1, ..., n\}$ be the universe of users. A* broadcast encryption (BE) scheme *is defined by five algorithms, which are as follows.*

BE.PG: This algorithm takes as input the security parameter 1^λ and the number of users in the system n. It returns the system's parameters $pars$. These will include a description of the message space MsgSp and the ciphertext space CtSp of the scheme. We write this as $pars \leftarrow \mathsf{BE.PG}(1^\lambda, n)$.

BE.Setup: This algorithm takes as input $pars$ and returns a master public key BE-MPK and a master secret key BE-MSK. We write this as (BE-MPK, BE-MSK) $\leftarrow \mathsf{BE.Setup}(pars)$.

BE.KeyGen: This is a key generation algorithm that on input BE-MPK, BE-MSK and an index $i \in U$ outputs a secret key sk_i for user i. We write this as $sk_i \leftarrow \mathsf{BE.KeyGen}(\text{BE-MPK}, \text{BE-MSK}, i)$.

BE.Enc: This is an encryption algorithm that on input BE-MPK, a message $M \in \mathsf{MsgSp}$ and a subset $S \subseteq U$, the broadcast target set, returns a ciphertext $C \in \mathsf{CtSp}$. We write this as $C \leftarrow \mathsf{BE.Enc}(\text{BE-MPK}, M, S)$.

BE.Dec: This is a decryption algorithm that on input BE-MPK, a ciphertext C and a secret key sk_i returns either a message $M \in \mathsf{MsgSp}$ or a failure symbol $\perp$. We write this as $\mathsf{BE.Dec}(\text{BE-MPK}, C, sk_i) = M$ or $\perp$.

These algorithms are required to satisfy the following correctness property*: For every λ, for every set of parameters pars output by BE.PG, for every BE-MPK, BE-MSK output by BE.KeyGen, for every message $M \in \mathsf{MsgSp}$, for every index $i \in U$ and for every $S \subseteq U$, if $sk_i \leftarrow \mathsf{BE.KeyGen}(\text{BE-MPK}, \text{BE-MSK}, i)$, if $C \leftarrow \mathsf{BE.Enc}(\text{BE-MPK}, M, S)$ and if $i \in S$ then $\mathsf{BE.Dec}(\text{BE-MPK}, C, sk_i) = M$.*

We observe that this definition no longer requires the set S as an input to the decryption algorithm. This is crucial in developing the notion of **anonymous broadcast encryption** (ANOBE), for which we next provide an appropriate security model for the case of *adaptive* adversaries.

We define the notion of *anonymity and indistinguishability under chosen-ciphertext attacks* (ANO-IND-CCA) for BE as follows.

ANO-IND-CCA security game for BE

Setup. The challenger $\mathcal{C}$ runs $\mathsf{BE.PG}(1^\lambda, n)$ to generate $pars$ and $\mathsf{BE.Setup}(pars)$ to obtain the master public key BE-MPK and the master secret key BE-MSK and gives BE-MPK to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ has access to a secret-key-extraction oracle $\mathcal{O}^{\text{BE-MSK}}$, to obtain secret keys of any index $i \in U$. The oracle will respond by returning $sk_i = \mathsf{BE.KeyGen}(\text{BE-MPK}, \text{BE-MSK}, i)$. $\mathcal{A}$ has also access to a decryption oracle $\mathcal{O}^{\text{BE-MSK}}$, to which it submits queries of the type (C, i), where $i \in U$, and the oracle will return the decryption $\mathsf{BE.Dec}(\text{BE-MPK}, C, sk_i)$.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and sets $S_0, S_1 \subseteq U$ of users. We require that S_0 and S_1 be of equal size and also impose the restriction that $\mathcal{A}$ has not issued key queries for any $i \in S_0 \triangle S_1 = (S_0 \setminus S_1) \cup (S_1 \setminus S_0)$. Further, if there exists an $i \in S_0 \cap S_1$ for which $\mathcal{A}$ has queried the key, then we require that $M_0 = M_1$. $\mathcal{A}$ passes M_0, M_1, S_0, S_1 to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0,1\}$ and computes $C^\star \leftarrow \mathsf{BE.Enc}(\text{BE-MPK}, M_b, S_b)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to have access to a secret-key-extraction oracle $\mathcal{O}^{\text{BE-MSK}}$, with the restrictions that $i \notin S_0 \triangle S_1$ and that, if $i \in S_0 \cap S_1$, then $M_0 = M_1$. $\mathcal{A}$ may continue issuing decryption queries (C, i) with the restriction that if $C = C^\star$ then either $i \notin S_0 \triangle S_1$ or $i \in S_0 \cap S_1$ and $M_0 = M_1$.

Guess. The adversary outputs its guess b' for b.

Definition 4.2 *A BE scheme* $B = (\mathsf{BE.PG}, \mathsf{BE.Setup}, \mathsf{BE.KeyGen}, \mathsf{BE.Enc}, \mathsf{BE.Dec})$ *is adaptively* anonymous and indistinguishable under chosen-ciphertext attacks *(or*

ANO-IND-CCA secure*) if all p.p.t. adversaries have at most negligible advantage in the above game, where $\mathcal{A}$'s advantage is defined as* $Adv^{\text{ANO-IND-CCA}}_{\mathcal{A},BE}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Like the corresponding definition of [6, Section 2], Definition 4.2 does not require the ANOBE ciphertext to hide the number of receivers. However, specific schemes (such as the one in Section 4.3.1) can also conceal the cardinality of S.

We have introduced a new notion of security for BE, namely ANO-IND-CCA security against adaptive adversaries. Our next step is to show that this notion is indeed *feasible*, and we do so by presenting a generic construction that relies solely on the existence of an IND-CCA secure PKE scheme. We will then improve its performance by giving alternative generic constructions whose underlying primitives require additional security properties.

4.3 ANOBE from Public-Key Encryption

4.3.1 ANOBE from minimal assumptions

Since our aim is to provide a formal treatment of anonymous broadcast encryption, we begin by showing that ANOBE *can be achieved*. Indeed, by simply assuming the existence of an IND-CCA secure PKE scheme we can construct an adaptively secure ANO-IND-CCA BE scheme.

The idea is simple. We will encrypt a message to *all* users in the system under each of their respective public keys. More specifically, we will use the PKE scheme to encrypt the intended message M to the users in the target set, and a special valid message ε to all the other users. The BE ciphertext will be a concatenation of PKE ciphertexts on which a signature is performed. The intended recipients will efficiently recover the message M by selecting the ciphertext corresponding to their index, while for the unauthorized users the public-key decryption algorithm will return ε. It is fairly intuitive to see that no property other than IND-CCA security for the PKE scheme (and strong one-time unforgeability for the signature) has to

be assumed in order to yield a secure ANOBE scheme, proving feasibility of our new notion. Let us explore in more detail this initial construction, for which we will assume the message spaces and key space to be bit-strings of fixed length.

Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a PKE scheme with message space $\mathsf{MsgSp} = \{0,1\}^m$. Let Σ=(Gen,Sign,Ver) be a signature scheme. We assume that the key space of Σ is $\mathsf{KSp} = \{0,1\}^v$, for some $v \in \mathsf{poly}(\lambda)$. We use Π and Σ to generically instantiate a BE scheme, with message space $\{0,1\}^{m-v}$. In the description hereafter, we include the symbol ε as a valid but distinguished message in $\{0,1\}^{m-v}$: in other words, all the messages that receivers accept as legal plaintexts are different from ε. The construction is as follows.

$\mathsf{BE.PG}(1^\lambda, n)$: Generate $pars \leftarrow \mathsf{PKE.PG}(1^\lambda)$ and return $(pars, n)$.

$\mathsf{BE.Setup}(pars, n)$: For $i = 1$ to n, generate $(sk_i, pk_i) \leftarrow \mathsf{PKE.KeyGen}(pars)$. The master private key is BE-MSK $= \{sk_i\}_{i=1}^n$ and the master public key consists of

$$\text{BE-MPK} = \Big(pars,\ \Sigma,\ \{pk_i\}_{i=1}^n\Big).$$

$\mathsf{BE.KeyGen}(\text{BE-MPK}, \text{BE-MSK}, i)$: Parse the master secret key BE-MSK as $\{sk_i\}_{i=1}^n$ and output sk_i.

$\mathsf{BE.Enc}(\text{BE-MPK}, M, S)$: To encrypt a message M for a receiver set $S \subseteq \{1, \ldots, n\}$, generate a one-time signature key pair $(sigk, vk) \leftarrow \mathsf{Gen}(1^\lambda)$. Then, for each $j = 1$ to n, compute $C_j = \mathsf{PKE.Enc}(pars, M||vk, pk_j)$ if $j \in S$ and $C_j = \mathsf{PKE.Enc}(pars, \varepsilon||vk, pk_j)$ if $j \notin S$. The ANOBE ciphertext consists of $C = \big(C_1, \ldots, C_n, \sigma\big)$, where $\sigma = \mathsf{Sign}\big(sigk, (C_1, \ldots, C_n)\big)$.

$\mathsf{BE.Dec}(\text{BE-MPK}, C, sk_i)$: Given $C = \big(C_1, \ldots, C_n, \sigma\big)$, compute $M' = \mathsf{PKE.Dec}(C_i, sk_i)$. If $M' \neq \perp$, parse M' as $M' = M||vk$ for some bit strings $M \in \{0,1\}^{m-v}$ and $vk \in \{0,1\}^v$. Then, if $\mathsf{Ver}\big(vk, (C_1, \ldots, C_n), \sigma\big) = 1$ and $M \neq \varepsilon$ return M. Otherwise, output $\perp$.

The correctness of the BE scheme follows directly from the correctness of Π and Σ. This construction is reminiscent of generic constructions of chosen-ciphertext-secure multiple encryption [44] and it is easily seen to yield a secure ANOBE. The following result in fact holds.

Theorem 4.3 *Let Π be an IND-CCA secure PKE scheme and let Σ be a strongly unforgeable one-time signature scheme. The BE scheme constructed above is ANO-IND-CCA secure against adaptive adversaries.*

In our proof of adaptive security we make use of a sequence of hybrid arguments where ciphertext components are gradually modified at each step and each hybrid argument requires the reduction to guess upfront the identity of an uncorrupted user. We note that Gentry and Waters [51] already briefly mentioned that such an approach could potentially be useful to prove adaptive security but, to the best of our knowledge, no rigorous analysis of this type was previously given in the literature. Moreover, in the constructions that follow in the rest of this chapter, achieving adaptive security represents even more of a challenge since it is a non-trivial task to get this proof technique to suitably interact with the methods we present for improving the overall efficiency.

For the proof of Theorem 4.3 we consider a sequence of games starting with Game 0 where the adversary is given an encryption of M_0 for S_0. In the last game, the adversary obtains an encryption of M_1 under S_1.

Game 0_{real}: is the real game when the challenger's bit is set to $b = 0$. The ANOBE adversary $\mathcal{A}$ is given public parameters BE-MPK consisting of n public-key encryption keys $\{pk_i\}_{i=1}^n$. For each $i \in \{1, \ldots, n\}$, user i's private key is sk_i. In the first stage, $\mathcal{A}$ adaptively chooses indices $i \in \{1, \ldots, n\}$ and obtains the corresponding sk_i. The adversary may also query the decryption oracle by sending requests (C, i) which are answered using the relevant private key sk_i. In the challenge step, $\mathcal{A}$ chooses messages M_0, M_1 and two subsets $S_0, S_1 \subset \{1, \ldots, n\}$ of equal size $|S_0| = |S_1| = \ell$. The challenger generates a one-time signature key pair $(sigk^\star, vk^\star) \leftarrow \mathsf{Gen}(1^\lambda)$ and returns the challenge ciphertext $C^\star = \big(C_1, \ldots, C_n, \sigma\big)$ where $\sigma = \mathsf{Sign}(sigk^\star, (C_1, \ldots, C_n))$ and, for $j = 1$ to n, C_j is computed as $C_j = \mathsf{PKE.Enc}(M_0||vk^\star, pk_j)$ if $j \in S_0$ and $C_j = \mathsf{PKE.Enc}(\varepsilon||vk^\star, pk_j)$ if $j \notin S_0$. In the second phase, $\mathcal{A}$ is allowed to make more corruption queries for indices i such that $i \in \{1, \ldots, n\} \backslash (S_0 \triangle S_1)$ and is granted further access to the decryption oracle under the usual restriction. Upon termination, $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$ and we define E_0^{real} to be the event that $b' = 0$.

Game 0**:** is as Game 0_{real} with the difference that the challenger now rejects all post-challenge decryption queries $\big(C = (C_1, \ldots, C_n, \sigma), i\big)$ for which $C_i = C_i^\star$ (*i.e.*, the i-component of C coincides with that of the challenge ciphertext). Clearly, the only situation where the challenger rejects a ciphertext that would not have been rejected in Game 0_{real} is when $\mathcal{A}$ breaks the security of the one-time signature. It is easy to see since $C_i = C_i^\star$ decrypts to a message whose last v bits form the challenge verification key $vk^\star$ as in the challenge phase. We call E_0 the event that $\mathcal{A}$ outputs $b' = 0$ in Game 0.

To describe subsequent games, it is convenient to represent the sets S_0 and S_1 as n-bit strings $s_{01} \ldots s_{0n} \in \{0,1\}^n$ and $s_{11} \ldots s_{1n} \in \{0,1\}^n$ such that, for each $b \in \{0,1\}$ and $j \in \{1, \ldots, n\}$, $s_{bj} = 1$ if and only if $j \in S_b$.

Game k **(**$1 \leq k \leq n$**):** From the two adversarially-chosen sets $S_0, S_1 \subset \{1, \ldots, n\}$ and their respective n-bit strings $s_{01} \ldots s_{0n}$ and $s_{11} \ldots s_{1n}$, the challenger $\mathcal{B}$ generates the challenge ciphertext as follows.

1. If $s_{0j} = s_{1j} = 1$, set $C_j = \mathsf{PKE.Enc}(M_1||vk^\star, pk_j)$ if $j \leq k$ and $C_j = \mathsf{PKE.Enc}(M_0||vk^\star, pk_j)$ if $j > k$. If $s_{0j} = s_{1j} = 0$, set $C_j = \mathsf{PKE.Enc}(\varepsilon||vk^\star, pk_j)$.
2. If $s_{0j} = 1$ and $s_{1j} = 0$, set $C_j = \mathsf{PKE.Enc}(\varepsilon||vk^\star, pk_j)$ if $j \leq k$ and $C_j = \mathsf{PKE.Enc}(M_0||vk^\star, pk_j)$ if $j > k$.
3. If $s_{0j} = 0$ and $s_{1j} = 1$, set $C_j = \mathsf{PKE.Enc}(M_1||vk^\star), pk_j$ if $j \leq k$ and $C_j = \mathsf{PKE.Enc}(\varepsilon||vk^\star, pk_j)$ if $j > k$.

The adversary is then returned $C^\star = \big(C_1, \ldots, C_n, \sigma\big)$ and the second phase is handled as in previous games. We call E_k the event of $\mathcal{A}$ outputting $b' = 0$ at the end of Game k.

Game n_{real}**:** is identical to Game n with the difference that, when handling decryption queries, the challenger no longer returns $\perp$ in decryption queries $(C = (C_1, \ldots, C_n, \sigma), i)$ such that that $C_i = C_i^\star$. Game n_{real} thus coincides with the real game when the challenger's bit equals $b = 1$. We let E_n^{real} be the event that $\mathcal{A}$ outputs the bit $b' = 0$ at the end of Game n_{real}.

Game 0_{real} and Game 0 are clearly indistinguishable if the one-time signature is strongly unforgeable and the same argument can be made about Game n and Game n_{real}.

We thus have $|\Pr[E_0^{real}] - \Pr[E_0]| = |\Pr[E_n^{real}] - \Pr[E_n]| \leq \mathbf{Adv}^{\mathrm{SUF-1CMA}}_{\mathcal{A},\Sigma}(\lambda)$. As for other game transitions, they are justified by Lemma 4.4 which demonstrates that, if Game k and Game $k-1$ can be distinguished for some $k \in \{1, \ldots, n\}$, there must exist an IND-CCA adversary $\mathcal{B}$ against the underlying encryption scheme. Putting the above altogether, we find

$$|\Pr[E_0^{real}] - \Pr[E_n^{real}]| \quad \leq \quad 2 \cdot \mathbf{Adv}^{\mathrm{SUF-1CMA}}_{\mathcal{A},\Sigma}(\lambda) + n \cdot \mathbf{Adv}^{\mathrm{IND-CCA}}_{\mathcal{B},\Pi}(\lambda).$$

Lemma 4.4 *Let Π be an IND-CCA secure PKE scheme. Then for any $k \in \{1, \ldots, n\}$, Game k is indistinguishable from Game $k-1$. More precisely, we have*

$$|\Pr[E_k] - \Pr[E_{k-1}]| \leq \mathbf{Adv}^{\mathrm{IND-CCA}}_{\mathcal{B},\Pi}(\lambda).$$

Proof. Towards a contradiction, let us assume that an adversary $\mathcal{A}$ can distinguish Game k and Game $k-1$. We show that it implies a chosen-ciphertext adversary $\mathcal{B}$ against Π.

We first recall that, in the challenge phase, the adversarially-chosen messages M_0, M_1 and sets S_0, S_1 must be such that either

- $S_0 = S_1$ and $M_0 \neq M_1$, in which case the adversary cannot corrupt any user in $S_0 = S_1$ (and, of course, we must have $|S_0| = |S_1| \geq 1$).
- $S_0 \neq S_1$, in which case the adversary is disallowed to corrupt any user in $S_0 \triangle S_1$.

If we consider the n-bit strings $s_{01} \ldots s_{0n} \in \{0,1\}^n$ and $s_{11} \ldots s_{1n} \in \{0,1\}^n$ associated with S_0 and S_1, Game k is identical to Game $k-1$ if $s_{0k} = s_{1k} = 0$ (since C_k is an encryption of ε in both games) and we thus assume that $s_{0k} = s_{1k} = 1$ or $s_{0k} \neq s_{1k}$. Moreover, if $s_{0k} = s_{1k} = 1$ (in other words, if $k \in S_0 \cap S_1$), the adversary can only corrupt sk_k in the situation where $M_0 = M_1$, in which case Game k and Game $k-1$ are also identical. In the following, we can thus only consider the situation $s_{0k} \neq s_{1k}$ (*i.e.*, $k \in S_0 \triangle S_1$), in which the adversary cannot legally query sk_k.

Our IND-CCA adversary $\mathcal{B}$ receives the public parameters $pars$ and a public key $pk^\star$ from its challenger and, to prepare BE–MPK for $\mathcal{A}$, it has to generate n encryption keys $pk_1, \ldots, pk_n$. To do this, $\mathcal{B}$ defines $pk_k = pk^\star$. Then, $\mathcal{B}$ runs the key generation algorithm of Π itself and generates $n-1$ key pairs $(sk_i, pk_i) \leftarrow \mathsf{PKE.KeyGen}(pars)$ for each $i \in \{1, \ldots, n\} \backslash \{k\}$. It finally hands the master public key BE–MPK $= \big(pars, \{pk_i\}_{i=1}^n, \Sigma\big)$ to $\mathcal{A}$.

At any time, $\mathcal{A}$ can corrupt an arbitrary user $i \in \{1, \ldots, n\}$ depending on the previously collected information. At each corruption query, $\mathcal{B}$ can consistently answer the query since it knows secret keys $\{sk_i\}_{i \neq k}$. When $\mathcal{A}$ queries the decryption of a ciphertext $(C = (C_1, \ldots, C_n, \sigma), i)$, we assume that $i = k$ (*i.e.*, the query involves the challenge key $pk_k = pk^\star$) since $\mathcal{B}$ can always decrypt by itself otherwise. To simulate the behaviour of the decryption algorithm without knowing $sk_k = sk^\star$, $\mathcal{B}$ invokes its own decryption oracle on C_k. If the IND-CCA challenger's response is not $\perp$ and can be parsed as $M||vk$, for some message $M \in \{0,1\}^{m-v}$ and some bit string $vk \in \{0,1\}^v$, $\mathcal{B}$ returns M to $\mathcal{A}$ if $\mathsf{Ver}(vk, (C_1, \ldots, C_n), \sigma) = 1$ and $M \neq \varepsilon$. In any other situation, $\mathcal{B}$ returns $\perp$, meaning that C_k fails to decrypt properly under sk_k.

In the challenge phase, $\mathcal{A}$ outputs messages M_0, M_1 and two subsets $S_0, S_1 \subseteq \{1, \ldots, n\}$. At this step, $\mathcal{B}$ generates a one-time signature key pair $(sig k^\star, vk^\star) \leftarrow \mathsf{Gen}(1^\lambda)$ and constructs two messages M_0', M_1' as follows.

- If $s_{0k} = 1$ and $s_{1k} = 0$, it sets $M_0' = M_0||vk^\star$ and $M_1' = \varepsilon||vk^\star$.
- If $s_{0k} = 0$ and $s_{1k} = 1$, it sets $M_0' = \varepsilon||vk^\star$ and $M_1' = M_1||vk^\star$.

The two messages M_0' and M_1' are sent to $\mathcal{B}$'s IND-CCA challenger which returns a challenge ciphertext $C^\star = \mathsf{PKE.Enc}(M_b', pk^\star)$, for some internally flipped random bit $b \leftarrow \{0,1\}$. The ANOBE challenge ciphertext is generated by setting $C_k^\star = C^\star$ and by defining the remaining ciphertext components as follows, for $j = 1$ to n.

1. If $s_{0j} = s_{1j} = 1$, set $C_j^\star = \mathsf{PKE.Enc}(M_1||vk^\star, pk_j)$ if $j \leq k-1$ and $C_j^\star = \mathsf{PKE.Enc}(M_0||vk^\star, pk_j)$ if $j > k$. If $s_{0j} = s_{1j} = 0$ set $C_j = \mathsf{PKE.Enc}(\varepsilon||vk^\star, pk_j)$.
2. If $s_{0j} = 1$ and $s_{1j} = 0$, set $C_j^\star = \mathsf{PKE.Enc}(\varepsilon||vk^\star, pk_j)$ if $j \leq k-1$ and $C_j^\star = \mathsf{PKE.Enc}(M_0||vk^\star, pk_j)$ if $j > k$.

3. If $s_{0j} = 0$ and $s_{1j} = 1$, set $C_j^\star = \mathsf{PKE.Enc}(M_1||vk^\star, pk_j)$ if $j \leq k-1$ and $C_j^\star = \mathsf{PKE.Enc}(\varepsilon||vk^\star, pk_j)$ if $j > k$.

The ANOBE adversary $\mathcal{A}$ is given $C = (C_1^\star, \ldots, C_n^\star, \sigma)$, where $\sigma = \mathsf{Sign}(sigk^\star, (C_1^\star, \ldots, C_n^\star))$.

In Phase 2, $\mathcal{A}$ makes another series of adaptive corruption queries for indices $i \notin S_0 \triangle S_1$ (and *a fortiori* such that $i \neq k$) and $\mathcal{B}$ deals with them as in Phase 1. When $\mathcal{A}$ makes a decryption query (C, i), $\mathcal{B}$ parses the ciphertext C as $C = (C_1, \ldots, C_n, \sigma)$ and handles the query using $\{sk_i\}_{i \neq k}$ if $i \neq k$. If $i = k$, $\mathcal{B}$ returns $\perp$ if $C_k = C_k^\star$. If $C_k \neq C_k^\star$, $\mathcal{B}$ can query C_k for decryption to its IND-CCA challenger and proceed as in pre-challenge decryption queries.

At the end of the game, $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$ and $\mathcal{B}$ outputs the same result. It is easy to see that $\mathcal{B}$'s advantage as an IND-CCA adversary is exactly the difference between $\mathcal{A}$'s probabilities of outputting 0 in Game k and Game $k-1$. Indeed, if $\mathcal{B}$'s challenger chooses $b = 0$ (and encrypts M_0' in the challenge phase), $\mathcal{B}$ is playing Game $k-1$. If $b = 1$, $\mathcal{B}$ is rather playing Game k. □

We have described an ANOBE scheme from minimal assumptions. We note that the encryption time is linear in n but decryption is performed in *constant* time, since a user simply selects the ciphertext component to decrypt according to its index. However, the ciphertext size is *linear* in n, as we encrypt to each user in the universe. It is desirable to improve on this and achieve a realization of ANOBE with more compact ciphertexts.

We will next see how to modify this first generic construction, obtaining an ANOBE scheme whose ciphertext size is linear in the size of the *target set* S.

4.3.2 ANOBE from robust and key-private PKE

As mentioned in the introduction, a solution to the broadcast encryption problem is to encrypt the message under the public key of each user in the privileged set (there we call it the *natural* solution). This approach, so often discarded in most BE literature due to efficiency reasons, turns out to provide another generic construction for ANOBE, which differs from the previous one as now we deploy a public-key

encryption scheme only to encrypt the *message* to the users *in the target set*.

However natural this solution may seem, it should not be approached naively. For this to yield an ANO-IND-CCA secure BE scheme, the underlying PKE scheme has to be IND-CCA *and* key-private as per Definition 2.3. While the need for this property is fairly intuitive, the requirement for the PKE scheme to be additionally weakly robust [2] is slightly more involved. Weak robustness is necessary for the correctness of the scheme but also for simulation consistency in the security proof. We have thoroughly studied the notion of robustness in Chapter 3 and seen many flavours of varying strength. For this application, weak robustness suffices and it can be generically achieved for any PKE scheme by appending some publicly-known redundancy to the message and checking it upon decryption. We refer to [2] for further details.

The construction we present is essentially the same as the construction that was already suggested by Barth, Boneh and Waters [6]. The novelty is that we now prove that it is actually *adaptively* secure, rather than just statically secure, as was established in [6]. Indeed, achieving security against such a strong adversary represents an important advance in the context of BE, as discussed in Section 4.1.1.2. The construction is as follows.

Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a PKE scheme and $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme. We construct an ANOBE scheme in the following way.

$\mathsf{BE.PG}(1^\lambda, n)$: Generate $pars \leftarrow \mathsf{PKE.PG}(1^\lambda)$ and return $(pars, n)$.

$\mathsf{BE.Setup}(pars, n)$: For $i = 1$ to n, generate $(sk_i, pk_i) \leftarrow \mathsf{PKE.KeyGen}(pars)$. The master private key is BE-MSK $= \{sk_i\}_{i=1}^n$ and the master public key consists of

$$\text{BE-MPK} = \Big(pars,\ \Sigma,\ \{pk_i\}_{i=1}^n\Big).$$

$\mathsf{BE.KeyGen}(\text{BE–MPK}, \text{BE–MSK}, i)$: Parse the master secret key BE–MSK as $\{sk_i\}_{i=1}^n$ and output sk_i.

$\mathsf{BE.Enc}(\text{BE–MPK}, M, S)$: To encrypt message M for a receiver set $S = \{i_1, \ldots, i_\ell\} \subseteq \{1, \ldots, n\}$ of size ℓ, generate a one-time signature key pair $(sigk, vk) \leftarrow \mathsf{Gen}(1^\lambda)$.

Then, for each $j = 1$ to ℓ, compute $C_j = \mathsf{PKE.Enc}(pars, M||vk, pk_{i_j})$. The ANOBE ciphertext consists of $C = \big(vk, C_{\tau(1)}, \ldots, C_{\tau(\ell)}, \sigma\big)$, where $\sigma = \mathsf{Sign}\big(sigk, (C_{\tau(1)}, \ldots, C_{\tau(\ell)})\big)$ and $\tau : \{1, \ldots, \ell\} \to \{1, \ldots, \ell\}$ is a random permutation.

$\mathsf{BE.Dec}(\text{BE-MPK}, C, sk_i)$: Parse the ciphertext C as a tuple $\big(vk, C_1, \ldots, C_\ell, \sigma\big)$. If $\mathsf{Ver}\big(vk, C_1, \ldots, C_\ell, \sigma\big) = 0$, return $\perp$. Otherwise, repeat the following steps for $j = 1$ to ℓ.

1. Compute $M' = \mathsf{PKE.Dec}(C_j, sk_i)$. If $M' \neq \perp$ and can moreover be parsed as $M' = M||vk$ for some M of appropriate length, return M.
2. If $j = \ell$ output $\perp$.

The correctness of the ANOBE scheme follows directly from the correctness and weak robustness of Π.

We note that actually this construction differs from the one presented in [6, Section 4.1] in a subtle way. Our construction, in fact, enjoys a slightly more efficient decryption procedure, since a legitimate user *first* checks whether the one-time signature verifies and then attempts decryption on the various PKE ciphertext components, a step which requires linear time. In [6], the analogous construction allows the receiver to perform the signature check only *after* having identified the correct ciphertext component, from the decryption of which he can recover the necessary verification key.

Theorem 4.5 *Let Π be an IND-CCA, key-private and weakly robust PKE scheme, and let Σ be a strongly unforgeable one-time signature scheme. Then the BE scheme constructed above is adaptively ANO-IND-CCA.*

Similarly to the proof of Theorem 4.3, for the proof of Theorem 4.5 we consider a sequence of games where the adversary is given an encryption of M_0 under S_0 in Game 0 while, in the last game, the adversary gets an encryption of M_1 under S_1.

Game 0_{real}: corresponds to the real game when the challenger's bit is $b = 0$. Namely, the ANOBE adversary $\mathcal{A}$ is given public parameters BE-MPK con-

taining $pars$ and n public keys $\{pk_i\}_{i=1}^n$. For each $i \in \{1,\ldots,n\}$, user i's private key is sk_i. In the first phase, the adversary $\mathcal{A}$ adaptively chooses indices $i \in \{1,\ldots,n\}$ and obtains the corresponding sk_i. The adversary may also invoke the decryption oracle by making queries (C,i) which are handled using the relevant private key sk_i. In the challenge phase, the adversary $\mathcal{A}$ comes up with messages M_0, M_1 and two subsets $S_0, S_1 \subset \{1,\ldots,n\}$ of equal size $|S_0| = |S_1| = \ell$ with $S_0 \neq S_1$. The challenger generates a one-time signature key pair $(sigk^\star, vk^\star) \leftarrow \mathsf{Gen}(1^\lambda)$, parses S_0 as $\{\theta_1,\ldots,\theta_\ell\}$ and returns the challenge ciphertext $C^\star = \big(vk^\star, C_{\tau(1)},\ldots,C_{\tau(\ell)},\sigma\big)$ where $C_j = \mathsf{PKE.Enc}(pars, M_0||vk^\star, pk_{\theta_j})$ for $j = 1$ to ℓ and $\tau : \{1,\ldots,\ell\} \to \{1,\ldots,\ell\}$ is a random permutation. In the second phase, $\mathcal{A}$ is allowed to make more decryption queries (under the usual restriction) and key queries for arbitrary indices i such that $i \in \{1,\ldots,n\}\backslash(S_0 \triangle S_1)$. Eventually, $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$ and we define E_0^{real} to be the event that $b' = 0$.

Game 0**:** is as Game 0_{real} but the challenger now rejects all post-challenge decryption queries (C,i) where C contains the same verification key $vk^\star$ as in the challenge phase. We call E_0 the event that $\mathcal{A}$ outputs $b' = 0$ in Game 0.

Game k **(**$1 \leq k \leq \ell$**):** From the two adversarially-chosen sets $S_0, S_1 \subseteq \{1,\ldots,n\}$, the challenger $\mathcal{B}$ defines the value $\phi = |S_0 \cap S_1|$ and then considers two ordered sets $S_0' = \{\theta_1,\ldots,\theta_\phi,\theta_{\phi+1},\ldots,\theta_\ell\}$, $S_1' = \{\rho_1,\ldots,\rho_\phi,\rho_{\phi+1},\ldots,\rho_\ell\}$ that are obtained by ordering S_0 and S_1 in such a way that $\theta_j = \rho_j$ for each $j \in \{1,\ldots,\phi\}$ and $\theta_j \neq \rho_j$ if $j \in \{\phi+1,\ldots,\ell\}$. Then, $\mathcal{B}$ generates the challenge ciphertext as follows.

1. For $j = 1$ to ϕ, set $C_j = \mathsf{PKE.Enc}(pars, M_1||vk^\star, pk_{\theta_j})$ if $j \leq k$ and $C_j = \mathsf{PKE.Enc}(pars, M_0||vk^\star, pk_{\theta_j})$ if $j > k$.
2. For $j = \phi+1$ to ℓ, set $C_j = \mathsf{PKE.Enc}(pars, M_1||vk^\star, pk_{\rho_j})$ if $j \leq k$ and $C_j = \mathsf{PKE.Enc}(pars, M_0||vk^\star, pk_{\theta_j})$ if $j > k$.

The adversary is then returned $C^\star = \big(vk^\star, C_{\tau(1)},\ldots,C_{\tau(\ell)},\sigma\big)$, for a randomly chosen permutation $\tau : \{1,\ldots,\ell\} \to \{1,\ldots,\ell\}$, and the second phase is handled as in previous games. We call E_k the event of $\mathcal{A}$ outputting $b' = 0$ at the end of Game k.

Game ℓ_{real}: is identical to Game ℓ with the difference that, when handling decryption queries, the challenger no longer rejects ciphertexts that contain the verification key $vk^\star$. Game ℓ_{real} actually proceeds like the real game when the challenger's bit is $b = 1$. We let E_ℓ^{real} be the event that $\mathcal{A}$ outputs the bit $b' = 0$ at the end of Game ℓ_{real}.

Game 0_{real} and Game 0 are indistinguishable if the one-time signature is strongly unforgeable and the same argument can be made about Game ℓ and Game ℓ_{real}.

We thus have $|\Pr[E_0^{real}] - \Pr[E_0]| = |\Pr[E_\ell^{real}] - \Pr[E_\ell]| \leq \mathbf{Adv}^{\mathrm{SUF-1CMA}}_{\mathcal{A},\Sigma}(\lambda)$. As for other game transitions, they are justified by Lemmas 4.6 and 4.7 that separately consider the situations where $k \leq \phi$ and $k > \phi$. More precisely, we have that, if Game k and Game $k-1$ can be distinguished for some $k \in \{1, \ldots, \ell\}$, Lemmas 4.6 and 4.7 show that there exists either a IND-IK-CCA adversary $\mathcal{B}$ or a WROB-CCA adversary $\mathcal{B}'$ (see sections 2.3.1 and 3.2 for definitions of these two notions) against the encryption scheme. Putting the above arguments altogether, we obtain

$$\begin{aligned}|\Pr[E_0^{real}] - \Pr[E_\ell^{real}]| &\leq 2 \cdot \mathbf{Adv}^{\mathrm{SUF-1CMA}}_{\mathcal{A},\Sigma}(\lambda) + n^2 \cdot \ell \cdot \Big(\mathbf{Adv}^{\mathrm{IND-IK-CCA}}_{\mathcal{B},\Pi}(\lambda) \\ &\quad + \mathbf{Adv}^{\mathrm{WROB-CCA}}_{\mathcal{B}',\Pi}(\lambda)\Big) \\ &\leq 2 \cdot \mathbf{Adv}^{\mathrm{SUF-1CMA}}_{\mathcal{A},\Sigma}(\lambda) + n^3 \cdot \Big(\mathbf{Adv}^{\mathrm{IND-IK-CCA}}_{\mathcal{B},\Pi}(\lambda) \\ &\quad + \mathbf{Adv}^{\mathrm{WROB-CCA}}_{\mathcal{B}',\Pi}(\lambda)\Big).\end{aligned}$$

Lemma 4.6 *Let Π be an IND-CCA PKE scheme. Then for each $k \in \{1, \ldots, \phi\}$, Game k is indistinguishable from Game $k-1$. More precisely, we have*

$$|\Pr[E_k] - \Pr[E_{k-1}]| \leq n \cdot \boldsymbol{Adv}^{\mathrm{IND-CCA}}_{\mathcal{B},\Pi}(\lambda).$$

Proof. Assuming that an attacker $\mathcal{A}$ can distinguish Game k and Game $k-1$, we build a chosen-ciphertext adversary $\mathcal{B}$ against Π. For each $k \in \{1, \ldots, \phi\}$, we observe that Game k and Game $k-1$ are identical when $M_0 = M_1$ and we thus assume $M_0 \neq M_1$, so that the adversary cannot corrupt any user in $S_0 \cap S_1$.

$\mathcal{B}$ obtains *pars* and a public key $pk^\star$ from its challenger and it has to prepare a master public key BE-MPK comprising n encryption keys $pk_1, \ldots, pk_n$ for the ANOBE adversary $\mathcal{A}$. To this end, picks $i^\star \leftarrow \{1, \ldots, n\}$ at random and defines $pk_{i^\star} = pk^\star$. Then, $\mathcal{B}$ runs $\mathsf{PKE.KeyGen}$ and generates $n-1$ key pairs (sk_i, pk_i)

on its own for each $i \in \{1, \ldots, n\} \backslash \{i^\star\}$. It finally gives the master public key BE-MPK $= \big(pars, \Sigma, \{pk_i\}_{i=1}^n\big)$ to $\mathcal{A}$.

At any time, $\mathcal{A}$ is allowed to corrupt an arbitrary user $i \in \{1, \ldots, n\}$ depending on the information it gathered so far. At each corruption query, $\mathcal{B}$ aborts and fails in the event that $\mathcal{A}$ chooses to corrupt user $i^\star$. Otherwise, $\mathcal{B}$ is necessarily able to consistently answer the query since it knows secret keys $\{sk_i\}_{i \neq i^\star}$. When the adversary $\mathcal{A}$ makes a decryption query $(C = (vk, C_1, \ldots, C_\ell, \sigma), i)$, we assume that the query involves the challenge key $pk^\star$ since $\mathcal{B}$ can always decrypt itself using sk_i otherwise. To simulate the decryption algorithm without knowing the challenge private key $sk^\star$, $\mathcal{B}$ proceeds as follows. For $j = 1$ to ℓ, it resorts to its IND-CCA challenger and asks it for the decryption of C_j. If the IND-CCA challenger's response differs from $\perp$ and can be parsed as $M||vk$, for some message M of appropriate length, $\mathcal{B}$ returns M to $\mathcal{A}$. If the counter j reaches its maximal value ℓ and no decryption query provided a result of the form $M||vk$, $\mathcal{B}$ returns $\perp$ to indicate that the ciphertext fails to decrypt properly.

In the challenge phase, $\mathcal{A}$ outputs two equal-length messages M_0, M_1 and two subsets $S_0, S_1 \subseteq \{1, \ldots, n\}$ of equal size. $\mathcal{B}$ re-orders S_0, S_1 as $S_0' = \{\theta_1, \ldots, \theta_\phi, \theta_{\phi+1}, \ldots, \theta_\ell\}$, $S_1' = \{\rho_1, \ldots, \rho_\phi, \rho_{\phi+1}, \ldots, \rho_\ell\}$ where $\theta_j = \rho_j$ for each $j \in \{1, \ldots, \phi\}$. If $\theta_k \neq i^\star$, $\mathcal{B}$ aborts and declares failure and we denote by Good the event that $\theta_k = i^\star$.

If the event Good occurs, $\mathcal{B}$ chooses a one-time signature key pair $(sigk^\star, vk^\star) \leftarrow \mathsf{Gen}(1^\lambda)$ and sends the messages $(M_0||vk^\star)$, $(M_1||vk^\star)$ to its IND-CCA challenger. The latter replies by generating a challenge ciphertext $C^\star = \mathsf{PKE.Enc}(pars, M_b||vk^\star, pk^\star)$, for some internally flipped random bit $b \leftarrow \{0, 1\}$. The ANOBE challenge ciphertext is then generated as follows.

1. For $j = 1$ to $k - 1$, $\mathcal{B}$ sets $C_j = \mathsf{PKE.Enc}\big(pars, M_1||vk^\star, pk_{\theta_j}\big)$.
2. For $j = k + 1$ to ℓ, $\mathcal{B}$ sets $C_j = \mathsf{PKE.Enc}\big(pars, M_0||vk^\star, pk_{\theta_j}\big)$.
3. Finally, set $C_k = C^\star$.

$\mathcal{A}$ then receives $C = (vk^\star, C_{\tau(1)}, \ldots, C_{\tau(\ell)}, \sigma)$, where $\sigma = \mathsf{Sign}(sigk^\star, C_{\tau(1)}, \ldots, C_{\tau(\ell)})$ and $\tau : \{1, \ldots, \ell\} \to \{1, \ldots, \ell\}$ is a random permutation.

In the second phase, $\mathcal{A}$ makes another series of adaptive corruption queries for indices $i \notin S_0 \triangle S_1$ and $\mathcal{B}$ deals with them as in the first phase. Whenever $\mathcal{A}$ makes a decryption query (C, i), $\mathcal{B}$ parses the ciphertext C as $C = (vk, C_1, \ldots, C_\ell, \sigma)$ and

outputs $\perp$ if $vk = vk^\star$ or if σ is invalid. Otherwise, if $i \neq i^\star$, $\mathcal{B}$ simply runs the legal decryption procedure on its own since it knows sk_i. If $i = i^\star$, $\mathcal{B}$ appeals to its IND-CCA challenger and the decryption oracle it is given access to. Namely, ciphertexts $\{C_1, \ldots, C_\ell\}$ are handled by repeating the following steps for $j = 1$ to ℓ.

- If $C_j = C^\star$, $\mathcal{B}$ considers that C_j decrypts to $\perp$ under $sk^\star$ (which is legitimate since $C^\star$ would decrypt to $M_b||vk^\star$ and $vk \neq vk^\star$) and does not make use of its decryption oracle.
- If $C_j \neq C^\star$, $\mathcal{B}$ queries the decryption of C_j. If the result can be parsed as $M||vk$ for some plaintext M of appropriate length, $\mathcal{B}$ outputs M.

If the counter j reaches ℓ and no decryption query resulted in a plaintext of the form $M||vk$, $\mathcal{B}$ returns $\perp$.

Eventually, the adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$ and $\mathcal{B}$ outputs the same result. If $\mathcal{B}$ did not abort, its advantage as an IND-CCA adversary is as large as the difference between $\mathcal{A}$'s probabilities of outputting 0 in Game k and Game $k-1$. Indeed, if $\mathcal{B}$'s challenger chooses $b = 0$, then $\mathcal{B}$ is clearly playing Game $k-1$ whereas, if $b = 1$, $\mathcal{B}$ is playing Game k.

Now, let us assess $\mathcal{B}$'s probability not to abort. First, since $M_0 \neq M_1$ by hypothesis, $\mathcal{A}$ is not allowed to corrupt any user in $S_0 \cap S_1 = \{\theta_1, \ldots, \theta_\phi\}$. Since $\theta_k \in S_0 \cap S_1$, a sufficient condition for $\mathcal{B}$ not to be asked for the unknown private key $sk_{i^\star}$ is to be lucky when drawing $i^\star \leftarrow \{1, \ldots, n\}$ and have event Good occurring. This is the case with probability $\Pr[\mathsf{Good}] = 1/n$ since the choice of $i^\star$ is completely independent of $\mathcal{A}$'s view. $\square$

Lemma 4.7 *Let Π be a IND-IK-CCA and weakly robust PKE scheme. Then for each $k \in \{\phi+1, \ldots, \ell\}$, Game k is indistinguishable from Game $k-1$. More precisely, for any ANOBE adversary distinguishing the two games, there exists either an IND-IK-CCA adversary $\mathcal{B}$ or a WROB-CCA adversary $\mathcal{B}'$ such that*

$$|\Pr[E_k] - \Pr[E_{k-1}]| \leq n^2 \cdot \Big(\boldsymbol{Adv}^{\mathrm{IND-IK-CCA}}_{\mathcal{B},\Pi}(\lambda) + \boldsymbol{Adv}^{\mathrm{WROB-CCA}}_{\mathcal{B}',\Pi}(\lambda) \Big).$$

Proof. We prove that, if an ANOBE attacker $\mathcal{A}$ is able to distinguish Game k and Game $k-1$ for some $k \in \{\phi+1, \ldots, \ell\}$, we can either translate $\mathcal{A}$ into an IND-IK-

CCA adversary $\mathcal{B}$ against Π or break its WROB-CCA property.

The IND-IK-CCA adversary $\mathcal{B}$ takes as input $pars$ and two public keys $pk_0^\star$, $pk_1^\star$ from its IND-IK-CCA challenger and we call $sk_0^\star$ and $sk_1^\star$ the underlying private keys. Algorithm $\mathcal{B}$ has to generate a master public key BE-MPK containing n public key keys $pk_1, \ldots, pk_n$. To this end, $\mathcal{B}$ picks two distinct indices $i_0^\star, i_1^\star \leftarrow \{1, \ldots, n\}$ and defines $pk_{i_0^\star} = pk_0^\star$ and $pk_{i_1^\star} = pk_1^\star$. Then, $\mathcal{B}$ runs $\mathsf{PKE.Keygen}$ and generates $n-2$ key pairs (sk_i, pk_i) for each $i \in \{1, \ldots, n\} \backslash \{i_0^\star, i_1^\star\}$. The master public key BE-MPK $= \big(pars, \Sigma, \{pk_i\}_{i=1}^n\big)$ is provided as input to $\mathcal{A}$.

Throughout the game, $\mathcal{A}$ can adaptively corrupt any user $i \in \{1, \ldots, n\}$. At each corruption query, $\mathcal{B}$ aborts if the queried index i falls in $\{i_0^\star, i_1^\star\}$. Otherwise, $\mathcal{B}$ necessarily knows the queried secret key sk_i and hands it to $\mathcal{A}$. For each decryption query $(C = (vk, C_1, \ldots, C_\ell, \sigma), i)$ made by $\mathcal{A}$, $\mathcal{B}$ can handle the query on its own whenever $i \notin \{i_0^\star, i_1^\star\}$. If $i = i_0^\star$ (resp. $i = i_1^\star$), $\mathcal{B}$ queries its own decryption oracle up to ℓ times and successively asks for the decryption of $C_1, \ldots, C_\ell$ under $sk_0^\star$ (resp. $sk_1^\star$). At the first answer that differs from $\perp$ and can be parses as $M||vk$, for some M of the right length, $\mathcal{B}$ returns M. If $\mathcal{B}$ fails to obtain a decryption result of the form $M||vk$, for some M, $\mathcal{B}$ returns $\perp$ to $\mathcal{A}$, meaning that C does not properly decrypt under $sk_0^\star$ (resp. $sk_1^\star$).

In the challenge phase, $\mathcal{A}$ outputs two equal-length messages M_0, M_1 and subsets $S_0, S_1 \subseteq \{1, \ldots, n\}$ of equal size ℓ. These sets are re-ordered as $S_0' = \{\theta_1, \ldots, \theta_\phi, \theta_{\phi+1}, \ldots, \theta_\ell\}$ and $S_1' = \{\rho_1, \ldots, \rho_\phi, \rho_{\phi+1}, \ldots, \rho_\ell\}$ where $\theta_j = \rho_j$ for each $j \in \{1, \ldots, \phi\}$. If $\theta_k \neq i_0^\star$ or $\rho_k \neq i_1^\star$, $\mathcal{B}$ aborts. We denote by Good the event $(\theta_k = i_0^\star) \wedge (\rho_k = i_1^\star)$, which implies $pk_{\theta_k} = pk_0^\star$ and $pk_{\rho_k} = pk_1^\star$.

If Good occurs, $\mathcal{B}$ generates a one-time signature key pair $(sigk^\star, vk^\star) \leftarrow \mathsf{Gen}(\lambda)$ and sends the messages $(M_0||vk^\star)$, $(M_1||vk^\star)$ to its IND-IK-CCA challenger. The latter returns a challenge ciphertext $C^\star \leftarrow \mathsf{PKE.Enc}(pars, M_b||vk^\star, pk_b^\star)$, for some internally flipped random bit $b \leftarrow \{0,1\}$. The ANOBE adversary's challenge ciphertext is then obtained as follows.

1. For $j = 1$ to $k-1$, $\mathcal{B}$ sets $C_j = \mathsf{PKE.Enc}\big(pars, M_1||vk^\star, pk_{\rho_j}\big)$.
2. For $j = k+1$ to ℓ, $\mathcal{B}$ computes $C_j = \mathsf{PKE.Enc}\big(pars, M_0||vk^\star, pk_{\theta_j}\big)$.
3. Finally, set $C_k = C^\star$.

$\mathcal{A}$ receives the challenge $C = (vk^\star, C_{\tau(1)}, \ldots, C_{\tau(\ell)}, \sigma)$, where $\sigma = \mathsf{Sign}(sigk^\star,$

$(C_{\tau(1)}, \ldots, C_{\tau(\ell)}))$ and $\tau : \{1, \ldots, \ell\} \to \{1, \ldots, \ell\}$ is a random permutation.

In the second phase, $\mathcal{A}$ makes further adaptive corruption queries for indices $i \notin S_0 \triangle S_1$ and $\mathcal{B}$ handles them as previously. Decryption queries are handled as in the first phase with one difference: if $\mathcal{A}$ makes a decryption query $(C = (vk, C_1, \ldots, C_\ell, \sigma), i)$ for which we simultaneously have $i \in \{i_0, i_1\}$, $vk \neq vk^\star$ and $C_j = C^\star$ for some $j \in \{1, \ldots, \ell\}$, $\mathcal{B}$ considers that C_j decrypts to $\perp$ under sk_i without invoking its own decryption oracles on C_j. Since $vk \neq vk^\star$, it is clear that $C^\star$ cannot correctly decrypt to a message ending with vk under the private key $sk_b^\star$. Still, we have to rule out the possibility to have $\mathsf{PKE.Dec}(C^\star, sk_{1-b}^\star) = M||vk$, for some plaintext M, since this could render $\mathcal{A}$'s view inconsistent. If this event were to happen with non-negligible probability, algorithm $\mathcal{B}$ could be turned into a weak robustness (more precisely, WROB-CCA) adversary $\mathcal{B}'$. The latter would simply generate the ANOBE challenge ciphertext by computing $C_1, \ldots, C_\ell$ itself and waiting for $\mathcal{A}$ to make a decryption query $C = (vk, C_1, \ldots, C_\ell, \sigma)$ for which there exists $j \in \{1, \ldots, \ell\}$ such that C_j correctly decrypts under both sk_b and sk_{1-b}.

When $\mathcal{A}$ halts, it outputs a result $b' \in \{0, 1\}$ and $\mathcal{B}$ outputs b' as well. If $\mathcal{B}$ did not abort, its IND-IK-CCA advantage is as large as the gap between $\mathcal{A}$'s probabilities of outputting 0 in Game k and Game $k-1$. Indeed, if $\mathcal{B}$'s IND-IK-CCA challenger sets its challenge bit as $b = 0$, $\mathcal{B}$ is playing Game $k-1$ with $\mathcal{A}$ whereas, if the IND-IK-CCA challenger sets $b = 1$, $\mathcal{B}$ is playing Game k.

Now, let us assess $\mathcal{B}$'s probability not to abort. Recall that the adversary $\mathcal{A}$ cannot legally corrupt any user in $S_0 \triangle S_1 = \{\theta_{\phi+1}, \ldots, \theta_\ell, \rho_{\phi+1}, \ldots, \rho_\ell\}$. For this reason, a sufficient condition for $\mathcal{A}$ not to query the private keys sk_{θ_k} or sk_{ρ_k} is to have Good occurring. Since event Good occurs with probability $\Pr[\mathsf{Good}] = 1/n(n-1) > 1/n^2$, the claimed result follows. □

In terms of efficiency, from this construction we will obtain secure ANOBE schemes with typically very small (constant) private key storage requirements and ciphertexts which are $|S|$ times the size of the ciphertext of the underlying PKE scheme. Encryption and decryption have both cost linear in the size of S. If, for example (as suggested in [6]), we use the Cramer–Shoup PKE scheme to instantiate the ANOBE scheme, the private keys will have constant size (namely 5 elements in $\mathbb{Z}_p$), and the resulting ciphertext will consist of roughly $4 \cdot |S|$ group elements. The scheme will be adaptively secure in the standard model under the DDH assumption.

If we look at recent efficient instantiations of BE, for example that of Gentry and Waters [51], we have private keys whose size is linear in the number of users, and ciphertexts which consist of n bits plus 3 group elements (if we include the cost of transmitting a description of S as part of the ciphertext). It is clear that in general the solution of [51] is more efficient in terms of ciphertext size. The key point though is that it is not anonymous.

4.4 ANOBE from Identity-Based Encryption

In this section we present a generic construction for ANOBE from identity-based encryption (IBE). For this approach, we build upon the Canetti-Halevi-Katz (CHK) transformation [25] by applying carefully-crafted modifications. As recalled in Section 2.4, the CHK transformation takes a weakly secure IBE scheme (and a strongly one-time unforgeable signature scheme) and returns an IND-CCA secure PKE scheme. The idea is first to consider the master keys $(\mathsf{MPK}, \mathsf{MSK})$ of the IBE scheme as the key-pair (pk, sk) for the resulting PKE scheme and then to generate a signature key-pair $(sigk, vk)$. To encrypt message M to public key MPK run the identity-based encryption algorithm on master public key MPK, message M and identity vk. Finally sign the ciphertext and the verification key with $sigk$.

We modify the original transformation as follows. First of all, borrowing ideas from [73], we adapt it to be suitable for the anonymity setting. This involves using a multi-TA IBE scheme (as defined in Section 2.3.2), with the appropriate security property, namely sID-TAA-IND-CPA security. The work in [73] does this to achieve a key-private IND-CCA secure PKE scheme. For our purposes, we need to extend this technique to the BE setting, where multiple users are being addressed. The idea is that, within this transform, we encrypt m for the *same* identity vk under the $|S|$ different public keys. We then sign all ciphertexts and append the verification key vk (note that this signature binds all these ciphertexts together). Upon decryption, a user verifies the signature against vk and, if valid, proceeds to derive the decryption key for identity vk by running the IBE key-extraction algorithm on input his private key. We formalize this idea next.

Let $I = (\mathsf{IBE.PG}, \mathsf{IBE.TASetup}, \mathsf{IBE.KeyExt}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ be a weakly robust,

in a sense to be defined below, multi-TA IBE scheme and let Σ=(Gen,Sign,Ver) be a signature scheme. A multi-TA IBE scheme is **weakly robust** (as defined in [62]) if a polynomial-time adversary cannot find a valid message such that its encryption for identity id under a master public key returns a valid message when decrypted using the secret key corresponding to id, but extracted under another master public key. We use I and Σ to generically instantiate a BE scheme in the following way.

$\mathsf{BE.PG}(1^\lambda, n)$: Run $\mathsf{IBE.PG}$ on input 1^λ to obtain the system's parameters $pars$. Return $(pars, n)$.

$\mathsf{BE.Setup}(pars, n)$: For $i = 1$ to n, generate $(\mathrm{MSK}_i, \mathrm{MPK}_i) \leftarrow \mathsf{IBE.TASetup}(pars)$. The master private key is BE-MSK $= \{\mathrm{MSK}_i\}_{i=1}^n$ and the master public key consists of

$$\text{BE-MPK} = \Big(pars,\ \Sigma,\ \{\mathrm{MPK}_i\}_{i=1}^n\Big).$$

$\mathsf{BE.KeyGen}(\text{BE–MPK}, \text{BE–MSK}, i)$: Parse the master secret key BE–MSK as $\{\mathrm{MSK}_i\}_{i=1}^n$ and output MSK_i.

$\mathsf{BE.Enc}(\text{BE–MPK}, M, S)$: To encrypt M for a receiver set $S = \{i_1, \ldots, i_\ell\} \subseteq \{1, \ldots, n\}$ of size ℓ, generate a one-time signature key pair $(sigk, vk) \leftarrow \mathsf{Gen}(1^\lambda)$. Then, for each $j = 1$ to ℓ, compute $C_j \leftarrow \mathsf{IBE.Enc}(\mathrm{MPK}_i, M, vk)$. The ANOBE ciphertext is

$$C = \big(vk, C_{\tau(1)}, \ldots, C_{\tau(\ell)}, \sigma\big),$$

where $\sigma \leftarrow \mathsf{Sign}\big(sigk, C_{\tau(1)}, \ldots, C_{\tau(\ell)}\big)$ and $\tau : \{1, \ldots, \ell\} \to \{1, \ldots, \ell\}$ is a random permutation.

$\mathsf{BE.Dec}(\text{BE–MPK}, C, \mathrm{MSK}_i)$: Parse the ciphertext C as a tuple $\big(vk, C_1, \ldots, C_\ell, \sigma\big)$. If $\mathsf{Ver}\big(vk, C_1, \ldots, C_\ell, \sigma\big) = 0$, return $\perp$. Otherwise, compute $sk_{i_{vk}} \leftarrow \mathsf{IBE.KeyExt}(\mathrm{MPK}_i, \mathrm{MSK}_i, vk)$ and repeat the following steps for $j = 1$ to ℓ.

1. Compute $M' = \mathsf{IBE.Dec}(\mathrm{MPK}_i, sk_{i_{vk}}, C_j)$. If $M' \neq \perp$ return M'.
2. If $j = \ell$ output $\perp$.

The correctness of the BE scheme follows directly from the correctness and the weak robustness of the IBE scheme I used to construct it.

From the details of the security proof which we give next it will become apparent that in this construction the weak robustness of the underlying primitive is only needed for correctness, and it is not required in the simulation.

If instantiated with the multi-TA version of Gentry's IBE scheme [50, 73] (which can be made weakly robust simply by applying the transform in [2], as proved in [62]), this construction yields very short constant size private keys (just one element in $\mathbb{Z}_p^*$) and ciphertexts consisting of roughly $3 \cdot |S|$ group elements ($|S|$ in $\mathbb{G}$ and $2 \cdot |S|$ in $\mathbb{G}_T$) plus a signature and a verification key. Encryption and decryption have both cost linear in the size of S.

The following result holds.

Theorem 4.8 *Let I be a sID-TAA-IND-CPA secure IBE scheme and let Σ be a strongly unforgeable one-time signature. Then, the above BE scheme is adaptively ANO-IND-CCA secure.*

We first give some intuition for the proof. As observed earlier, in [73], the authors apply a modified version of the CHK transformation [25] using the same primitives as our generic construction to obtain a key-private IND-CCA PKE scheme. We introduced further modifications to build an ANO-IND-CCA secure broadcast encryption scheme. By similar arguments to those in [25] and [73], and by applying techniques analogous to those proving adaptive security in Theorem 4.5, we can show that adaptive ANO-IND-CCA security is achieved. We give details of the proof next.

Let us consider a sequence of games where the adversary is given an encryption of M_0 for S_0 in Game 0 while, in the last game, the adversary obtains an encryption of M_1 under S_1.

Game 0_{real}: corresponds to the real game when the challenger's bit is $b = 0$. In this game the challenger $\mathcal{C}$ first generates a one-time signature key pair $(sigk^\star, vk^\star) \leftarrow \mathsf{Gen}(1^\lambda)$. It then gives the ANOBE adversary $\mathcal{A}$ the public parameters BE–MPK containing $pars$ and n public keys $\{\text{MPK}_i\}_{i=1}^n$. For each

$i \in \{1, \ldots, n\}$, user i's private key is MSK_i. In the first phase, the adversary $\mathcal{A}$ adaptively chooses indices $i \in \{1, \ldots, n\}$ and obtains the corresponding MSK_i. The adversary may also invoke the decryption oracle by making queries (C, i) which are handled using the relevant private key MSK_i. In the challenge phase, the adversary $\mathcal{A}$ comes up with two equal-length messages M_0, M_1 and two subsets $S_0, S_1 \subset \{1, \ldots, n\}$ of equal size $|S_0| = |S_1| = \ell$ with $S_0 \neq S_1$. The challenger parses S_0 as $\{\theta_1, \ldots, \theta_\ell\}$ and returns the challenge ciphertext $C^\star = \big(vk^\star, C_{\tau(1)}, \ldots, C_{\tau(\ell)}, \sigma\big)$ where $C_j \leftarrow \mathsf{IBE.Enc}(\mathrm{MPK}_{\theta_j}, M_0, vk^\star)$ for $j = 1$ to ℓ, $\tau : \{1, \ldots, \ell\} \to \{1, \ldots, \ell\}$ is a random permutation and $\sigma \leftarrow \mathsf{Sign}(sigk^\star, (C_{\tau(1)}, \ldots, C_{\tau(\ell)}))$. In the second phase, $\mathcal{A}$ is allowed to make more decryption queries (under the usual restriction) and key queries for arbitrary indices i such that $i \in \{1, \ldots, n\} \backslash (S_0 \triangle S_1)$. Eventually, $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$ and we define E_0^{real} to be the event that $b' = 0$.

Game 0: is as Game 0_{real} but the challenger now rejects all decryption queries (C, i) where C contains the verification key $vk^\star$. We call E_0 the event that $\mathcal{A}$ outputs $b' = 0$ in Game 0.

Game k ($1 \leq k \leq \ell$): From the two adversarially-chosen sets $S_0, S_1 \subseteq \{1, \ldots, n\}$, the challenger $\mathcal{B}$ defines the value $\phi = |S_0 \cap S_1|$ and then considers two ordered sets $S_0' = \{\theta_1, \ldots, \theta_\phi, \theta_{\phi+1}, \ldots, \theta_\ell\}$, $S_1' = \{\rho_1, \ldots, \rho_\phi, \rho_{\phi+1}, \ldots, \rho_\ell\}$ that are obtained by ordering S_0 and S_1 in such a way that $\theta_j = \rho_j$ for each $j \in \{1, \ldots, \phi\}$ and $\theta_j \neq \rho_j$ if $j \in \{\phi + 1, \ldots, \ell\}$. Then, $\mathcal{B}$ generates the challenge ciphertext as follows.

1. For $j = 1$ to ϕ, set $C_j = \mathsf{IBE.Enc}(\mathrm{MPK}_{\theta_j}, M_1, vk^\star)$ if $j \leq k$ and $C_j = \mathsf{IBE.Enc}(\mathrm{MPK}_{\theta_j}, M_0, vk^\star)$ if $j > k$.
2. For $j = \phi + 1$ to ℓ, set $C_j = \mathsf{IBE.Enc}(\mathrm{MPK}_{\rho_j}, M_1, vk^\star)$ if $j \leq k$ and $C_j = \mathsf{IBE.Enc}(\mathrm{MPK}_{\theta_j}, M_0, vk^\star)$ if $j > k$.

The adversary is then returned $C^\star = \big(vk^\star, C_{\tau(1)}, \ldots, C_{\tau(\ell)}, \sigma\big)$, for a randomly chosen permutation $\tau : \{1, \ldots, \ell\} \to \{1, \ldots, \ell\}$, and the second phase is handled as in previous games. We call E_k the event of $\mathcal{A}$ outputting $b' = 0$ at the end of Game k.

Game ℓ_{real}: is identical to Game ℓ with the difference that, when handling decryption queries, the challenger no longer rejects ciphertexts that contain the

verification key $vk^\star$. Game ℓ_{real} actually proceeds like the real game when the challenger's bit is $b = 1$. We let E_ℓ^{real} be the event that $\mathcal{A}$ outputs the bit $b' = 0$ at the end of Game ℓ_{real}.

Game 0_{real} and Game 0 are indistinguishable if the one-time signature is strongly unforgeable and the same argument can be made about Game ℓ and Game ℓ_{real}.

We thus have $|\Pr[E_0^{real}] - \Pr[E_0]| = |\Pr[E_\ell^{real}] - \Pr[E_\ell]| \leq \mathbf{Adv}^{\mathrm{SUF-1CMA}}_{\mathcal{A},\Sigma}(\lambda)$. As for other game transitions, they are justified by Lemmas 4.9 and 4.10 that separately consider the situations where $k \leq \phi$ and $k > \phi$. More precisely, we have that, if Game k and Game $k-1$ can be distinguished for some $k \in \{1, \ldots, \ell\}$, Lemmas 4.9 and 4.10 show that there exists a sID-TAA-IND-CPA adversary $\mathcal{B}$ against the IBE scheme. Putting the above arguments altogether, we obtain

$$\begin{aligned} |\Pr[E_0^{real}] - \Pr[E_\ell^{real}]| &\leq 2 \cdot \mathbf{Adv}^{\mathrm{SUF-1CMA}}_{\mathcal{A},\Sigma}(\lambda) + n^2 \cdot \ell \cdot \mathbf{Adv}^{\mathrm{sID-TAA-IND-CPA}}_{\mathcal{B},\Pi}(\lambda) \\ &\leq 2 \cdot \mathbf{Adv}^{\mathrm{SUF-1CMA}}_{\mathcal{A},\Sigma}(\lambda) + n^3 \cdot \mathbf{Adv}^{\mathrm{sID-TAA-IND-CPA}}_{\mathcal{B},\Pi}(\lambda). \end{aligned}$$

Lemma 4.9 *Let I be a sID-IND-CPA multi-TA IBE scheme. Then for each $k \in \{1, \ldots, \phi\}$, Game k is indistinguishable from Game $k-1$. More precisely, we have*

$$|\Pr[E_k] - \Pr[E_{k-1}]| \leq n \cdot \boldsymbol{Adv}^{\mathrm{sID-IND-CPA}}_{\mathcal{B},I}(\lambda).$$

Proof. Assuming that an attacker $\mathcal{A}$ can distinguish Game k and Game $k-1$, we build a selective-id chosen-plaintext adversary $\mathcal{B}$ against I. For each $k \in \{1, \ldots, \phi\}$, we observe that Game k and Game $k-1$ are identical when $M_0 = M_1$ and we thus assume $M_0 \neq M_1$, so that the adversary cannot corrupt any user in $S_0 \cap S_1$.

$\mathcal{B}$ first generates a one-time signature key pair $(sigk^\star, vk^\star) \leftarrow \mathsf{Gen}(1^\lambda)$ and gives $vk^\star$ to its challenger $\mathcal{C}$. It then obtains $pars$ and the master public keys $\{\mathrm{MPK}_i\}_{i=1}^n$ of all TAs in the system from its challenger. At this point $\mathcal{B}$ selects the master public key $\mathrm{MPK}^\star$ it wishes to be challenged on. To this end, it picks $i^\star \leftarrow \{1, \ldots, n\}$ at random and defines $\mathrm{MPK}_{i^\star} = \mathrm{MPK}^\star$. It finally gives the master public key $\mathrm{BE\text{-}MPK} = \big(pars, \Sigma, \{\mathrm{MPK}_i\}_{i=1}^n\big)$ to $\mathcal{A}$.

At any time, $\mathcal{A}$ is allowed to corrupt an arbitrary user $i \in \{1, \ldots, n\}$ depending on the information it gathered so far. At each corruption query, $\mathcal{B}$ aborts and fails in the event that $\mathcal{A}$ chooses to corrupt user $i^\star$. Otherwise, $\mathcal{B}$ is able to consistently answer the query since it can forward the same corruption query to its own

challenger. When the adversary $\mathcal{A}$ makes a decryption query (C, i), $\mathcal{B}$ simulates the decryption algorithm in the following way. First, it parses the ciphertext C as $C = (vk, C_1, \ldots, C_\ell, \sigma)$ and outputs $\perp$ if $vk = vk^\star$ or if σ is invalid. Otherwise, $\mathcal{B}$ submits the query (i, vk) to its challenger to obtain $sk_{i_{vk}}$, the secret key corresponding to identity vk under TA i. Then, for $j = 1$ to ℓ, it runs $\mathsf{IBE.Dec}(\mathrm{MPK}_i, sk_{i_{vk}}, C_j)$ and returns M to $\mathcal{A}$ if it obtains a message M of the appropriate length. If the counter j reaches its maximal value ℓ and no decryption provided such a message, $\mathcal{B}$ returns $\perp$ to indicate that the ciphertext fails to decrypt properly.

In the challenge phase, $\mathcal{A}$ outputs two equal-length messages M_0, M_1 and two subsets $S_0, S_1 \subseteq \{1, \ldots, n\}$ of equal size. At this step, $\mathcal{B}$ re-orders S_0, S_1 as $S_0' = \{\theta_1, \ldots, \theta_\phi, \theta_{\phi+1}, \ldots, \theta_\ell\}$, $S_1' = \{\rho_1, \ldots, \rho_\phi, \rho_{\phi+1}, \ldots, \rho_\ell\}$ where $\theta_j = \rho_j$ for each $j \in \{1, \ldots, \phi\}$. If $\theta_k \neq i^\star$, $\mathcal{B}$ aborts and declares failure and we denote by Good the event that $\theta_k = i^\star$.

If the event Good occurs, $\mathcal{B}$ sends $i^\star$ and the two messages M_0, M_1 to its IND-CPA challenger. The latter replies by generating a challenge ciphertext $C^\star = \mathsf{IBE.Enc}(\mathrm{MPK}_{i^\star}, M_b, vk^\star)$, for some internally flipped random bit $b \leftarrow \{0, 1\}$. The ANOBE challenge ciphertext is then generated as follows.

1. For $j = 1$ to $k - 1$, $\mathcal{B}$ sets $C_j = \mathsf{IBE.Enc}(\mathrm{MPK}_{\rho_j}, M_1, vk^\star)$.
2. For $j = k + 1$ to ℓ, $\mathcal{B}$ sets $C_j = \mathsf{IBE.Enc}(\mathrm{MPK}_{\theta_j}, M_0, vk^\star)$.
3. Finally, set $C_k = C^\star$.

$\mathcal{A}$ then receives $C = (vk^\star, C_{\tau(1)}, \ldots, C_{\tau(\ell)}, \sigma)$, where $\sigma = \mathsf{Sign}(sigk^\star, C_{\tau(1)}, \ldots, C_{\tau(\ell)})$ and $\tau : \{1, \ldots, \ell\} \to \{1, \ldots, \ell\}$ is a random permutation.

In the second phase, $\mathcal{A}$ makes another series of adaptive corruption queries for indices $i \notin S_0 \triangle S_1$ and $\mathcal{B}$ deals with them as in the first phase. Similarly, whenever $\mathcal{A}$ makes a decryption query $(C = (vk, C_1, \ldots, C_\ell, \sigma), i)$, $\mathcal{B}$ handles it as before. In fact, (i, vk) is *always* a valid query to its challenger since $vk \neq vk^\star$. Eventually, the adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$ and $\mathcal{B}$ outputs the same result. If $\mathcal{B}$ did not abort, its advantage as an IND-CCA adversary is as large as the difference between $\mathcal{A}$'s probabilities of outputting 0 in Game k and Game $k - 1$. Indeed, if $\mathcal{B}$'s challenger chooses $b = 0$, then $\mathcal{B}$ is clearly playing Game $k - 1$ whereas, if $b = 1$, $\mathcal{B}$ is playing Game k.

Now, let us assess $\mathcal{B}$'s probability not to abort. First, since $M_0 \neq M_1$ by hypothesis, $\mathcal{A}$ is not allowed to corrupt any user in $S_0 \cap S_1 = \{\theta_1, \ldots, \theta_\phi\}$. Since $\theta_k \in S_0 \cap S_1$, a sufficient condition for $\mathcal{B}$ not to be asked for the unknown private key $\mathrm{MSK}_{i^\star}$ is to be lucky when drawing $i^\star \leftarrow \{1, \ldots, n\}$ and have event Good occurring. This is the case with probability $\Pr[\mathsf{Good}] = 1/n$ since the choice of $i^\star$ is completely independent of $\mathcal{A}$'s view. □

Lemma 4.10 *Let I be a sID-TAA-IND-CPA secure IBE scheme. Then for each $k \in \{\phi + 1, \ldots, \ell\}$, Game k is indistinguishable from Game $k-1$. More precisely, for any ANOBE adversary distinguishing the two games, there exists an sID-TAA-IND-CPA adversary $\mathcal{B}$ such that*

$$|\Pr[E_k] - \Pr[E_{k-1}]| \leq n^2 \cdot \boldsymbol{Adv}_{\mathcal{B},\Pi}^{\mathrm{sID-TAA-IND-CPA}}(\lambda).$$

Proof. We prove that, if an ANOBE attacker $\mathcal{A}$ is able to distinguish Game k and Game $k-1$ for some $k \in \{\phi + 1, \ldots, \ell\}$, we can translate $\mathcal{A}$ into a successful sID-TAA-IND-CPA adversary $\mathcal{B}$ against I.
$\mathcal{B}$ first generates a one-time signature key pair $(sigk^\star, vk^\star) \leftarrow \mathsf{Gen}(1^\lambda)$ and gives $vk^\star$ to its challenger $\mathcal{C}$. It then obtains $pars$ and the master public keys $\{\mathrm{MPK}_i\}_{i=1}^n$ of all TAs in the system from its challenger. At this point $\mathcal{B}$ selects the master public keys $\mathrm{MPK}_{0^\star}$ and $\mathrm{MPK}_{1^\star}$ it wishes to be challenged on. To this end, it picks $i_0^\star, i_1^\star \leftarrow \{1, \ldots, n\}$ at random and defines $\mathrm{MPK}_{i_0^\star} = \mathrm{MPK}_{0^\star}$ and $\mathrm{MPK}_{i_1^\star} = \mathrm{MPK}_{1^\star}$. It finally gives the master public key $\text{BE-MPK} = \big(pars, \Sigma, \{\mathrm{MPK}_i\}_{i=1}^n\big)$ to $\mathcal{A}$.

Throughout the game, $\mathcal{A}$ can adaptively corrupt any user $i \in \{1, \ldots, n\}$. At each corruption query, $\mathcal{B}$ aborts if the queried index i falls in $\{i_0^\star, i_1^\star\}$. Otherwise, $\mathcal{B}$ simply passes the corruption query to its challenger and forwards its response to $\mathcal{A}$. For each decryption query (C, i) made by $\mathcal{A}$, $\mathcal{B}$ parses C as $(vk, C_1, \ldots, C_\ell, \sigma)$ and outputs $\perp$ if $vk = vk^\star$ or or if σ is invalid. Otherwise, $\mathcal{B}$ submits the query (i, vk) to its challenger to obtain $sk_{i_{vk}}$, the secret key corresponding to identity vk under TA i. Then, for $j = 1$ to ℓ, it runs $\mathsf{IBE.Dec}(\mathrm{MPK}_i, sk_{i_{vk}}, C_j)$ and returns M to $\mathcal{A}$ if it obtains a message M of the appropriate length. If the counter j reaches its maximal value ℓ and no decryption provided such a message, $\mathcal{B}$ returns $\perp$ to indicate that the ciphertext fails to decrypt properly.

In the challenge phase, $\mathcal{A}$ outputs two equal-length messages M_0, M_1 and subsets $S_0, S_1 \subseteq \{1, \ldots, n\}$ of equal size ℓ. These sets are re-ordered as $S_0' = \{\theta_1, \ldots, \theta_\phi, \theta_{\phi+1},$

$\ldots, \theta_\ell\}$ and $S'_1 = \{\rho_1, \ldots, \rho_\phi, \rho_{\phi+1}, \ldots, \rho_\ell\}$ where $\theta_j = \rho_j$ for each $j \in \{1, \ldots, \phi\}$. If $\theta_k \neq i_0^\star$ or $\rho_k \neq i_1^\star$, $\mathcal{B}$ aborts. We denote by Good the event $(\theta_k = i_0^\star) \wedge (\rho_k = i_1^\star)$, which implies $\mathrm{MPK}_{\theta_k} = \mathrm{MPK}_{i_0^\star}$ and $\mathrm{MPK}_{\rho_k} = \mathrm{MPK}_{i_1^\star}$.

If the event Good occurs, $\mathcal{B}$ sends $i_0^\star$, $i_1^\star$ and the two messages M_0, M_1 to its IND-CPA challenger. The latter returns a challenge ciphertext $C^\star \leftarrow \mathsf{IBE.Enc}(\mathrm{MPK}_{i_b^\star}, M_b, vk^\star)$, for some internally flipped random bit $b \leftarrow \{0, 1\}$. The ANOBE adversary's challenge ciphertext is then obtained as follows.

1. For $j = 1$ to $k - 1$, $\mathcal{B}$ sets $C_j = \mathsf{IBE.Enc}(\mathrm{MPK}_{\rho_j}, M_1, vk^\star)$.
2. For $j = k + 1$ to ℓ, $\mathcal{B}$ computes $C_j = \mathsf{IBE.Enc}(\mathrm{MPK}_{\theta_j}, M_0, vk^\star)$.
3. Finally, set $C_k = C^\star$.

$\mathcal{A}$ receives the challenge $C = (vk^\star, C_{\tau(1)}, \ldots, C_{\tau(\ell)}, \sigma)$, where $\sigma = \mathsf{Sign}(sigk^\star, (C_{\tau(1)}, \ldots, C_{\tau(\ell)}))$ and $\tau : \{1, \ldots, \ell\} \to \{1, \ldots, \ell\}$ is a random permutation.

In the second phase, $\mathcal{A}$ makes further adaptive corruption queries for indices $i \notin S_0 \triangle S_1$ and $\mathcal{B}$ handles them as previously. Decryption queries are handled as in the first phase. As before, we note that since $vk \neq vk^\star$, (i, vk) is always a valid query to $\mathcal{B}$'s challenger.

When $\mathcal{A}$ halts, it outputs a result $b' \in \{0, 1\}$ and $\mathcal{B}$ outputs b' as well. If $\mathcal{B}$ did not abort, its advantage is as large as the gap between $\mathcal{A}$'s probabilities of outputting 0 in Game k and Game $k - 1$. Indeed, if $\mathcal{B}$'s challenger sets its challenge bit as $b = 0$, $\mathcal{B}$ is playing Game $k - 1$ with $\mathcal{A}$ whereas, if the challenger sets $b = 1$, $\mathcal{B}$ is playing Game k.

Now, let us assess $\mathcal{B}$'s probability not to abort. Recall that the adversary $\mathcal{A}$ cannot legally corrupt any user in $S_0 \triangle S_1 = \{\theta_{\phi+1}, \ldots, \theta_\ell, \rho_{\phi+1}, \ldots, \rho_\ell\}$. For this reason, a sufficient condition for $\mathcal{A}$ not to query the private keys sk_{θ_k} or sk_{ρ_k} is to have Good occurring. Since event Good occurs with probability $\Pr[\mathsf{Good}] = 1/n(n-1) > 1/n^2$, the claimed result follows. □

We have therefore proved that identity-based encryption can be used to obtain a secure ANOBE scheme. We will next study the relation between ANOBE and another important primitive, attribute-based encryption.

4.5 ANOBE from Attribute-Based Encryption

Our goal is to establish the connections between attribute-based encryption (ABE) and BE. As a warm up, we look at some security notions and properties in both settings, and see how they relate to each other.

Before looking at the relation between the security models for BE and ABE, we consider an important notion common to both primitives, namely **collusion resistance**. As mentioned in Chapter 2, one of the defining properties of ABE is collusion resistance, in the sense that users cannot combine their attributes and successfully decrypt something they could have not decrypted individually. By building a BE scheme from an ABE one, as we intend to do next, the collusion resistance of the original ABE scheme is automatically inherited by the BE scheme. However in the BE setting, the only *attribute* a user can be associated with is *to have* a certain identity. In this sense, the collusion resistance property of ABE seems to be significantly stronger than the notion having the same name in the BE setting. By pooling identities together the users do not gain any advantage: either *none* of them is a legitimate user, and so, by the security of the scheme, they still cannot decrypt, or *at least one* is, but in this case it would make no sense for that user to collude with others. Obviously a legitimate user can give his private key to a non-intended recipient, but this is a completely different issue which does not fall in the framework of collusion resistance.

Also in the BE setting there is a notion of resistance against collusion attacks, i.e., attacks mounted by a coalition of users not in the target set who wish to decrypt the broadcast message. If, as detailed next, we construct a BE scheme from an ABE scheme, this property is naturally met by the security of the underlying ABE scheme, as this guarantees that *only* the authorized users can decrypt successfully.

We now recall some types of adversary and security properties in the two settings. In BE, we can find the following types of adversary, each of which we relate to concepts in the ABE world.

- Static adversary. This is an adversary that commits to a set of users S^* in

an Initialize phase before the setup algorithm is run. S^* is the set he wishes to attack and therefore there are some standard restrictions to the key extraction and decryption queries the adversary can issue (see [17] for details). From an ABE perspective, this is precisely an adversary in the selective-set (or selective-policy) security model.

- Semi-Static adversary. This notion has only recently been introduced in [51] and addresses an adversary who has to commit to a set $\tilde{S}$ in the Initialize phase, who can make private key extraction queries for any $i \notin \tilde{S}$, and who must choose a target set S^* for the challenge ciphertext with the restriction that $S^* \subseteq \tilde{S}$. Such an adversary is weaker than an adaptive adversary, but stronger than a static one, as the BE adversary can *adaptively* choose what subset of $\tilde{S}$ to attack. There is no existing equivalent notion in the ABE world, but it can be naturally introduced. For instance, in the key-policy setting, the adversary will have to commit ahead to a set of attributes $\tilde{S}$; he will be able to issue queries for attributes not in $\tilde{S}$ and finally he will choose an arbitrary subset of $\tilde{S}$.

- Adaptive adversary. This is the strongest type of adversary against BE [51], as he need not commit to any set of users before the setup algorithm is run. In ABE, such an adversarial notion is captured by the standard security model in which only in the challenge phase does the adversary select a set of attributes (or a policy) he wishes to attack, with the restriction that none of the answered queries allow him to win trivially.

As we have stressed so far, **anonymity** is a crucial security property for which we aim in a BE system. In the context of relating BE to ABE, we can observe that anonymity corresponds to the notion of attribute/policy-hiding, as defined in [58]. Briefly, the adversary selects *two* sets (policies) in either the Initialize phase or the Challenge phase, and then he will have to guess under which set (policy) encryption was performed.

We will now show how to obtain BE from ABE. As a consequence, any progress in the efficiency and security of ABE can be translated directly into corresponding improvements for ANOBE.

4.5.1 Generic constructions for ANOBE from ABE

This section is dedicated to formalizing the relation between ABE and BE. To this end, we provide generic ways to achieve BE (and in particular ANOBE) from both flavours of ABE, namely ciphertext-policy ABE (CP-ABE) and key-policy ABE (KP-ABE). While it is fairly natural to think of a BE scheme as a CP-ABE scheme whose ciphertext policy is *set-membership*, it is perhaps not so intuitive to see BE as arising as a special instance of KP-ABE.

We explore these ideas next, giving first some intuition and then the detailed constructions and proofs.

BE and CP-ABE. Let U be the universe of users of the desired BE scheme. We consider a CP-ABE scheme whose universe of attributes is precisely U. *Being user i* will translate to *having attribute i*. Furthermore, *belonging to the target broadcast set S* will correspond to a *set-membership policy*. This can be expressed as a disjunction of attributes, i.e. users. It is then very intuitive to think of a BE scheme as a CP-ABE scheme with the above-mentioned encryption policy.

More formally, let $\Gamma = (\mathsf{CP_{ABE}.PG}, \mathsf{CP_{ABE}.Setup}, \mathsf{CP_{ABE}.KeyGen}, \mathsf{CP_{ABE}.Enc}, \mathsf{CP_{ABE}.Dec})$ be a CP-ABE scheme which efficiently supports disjunction policies. Let U be the universe of attributes, where $|U| = n$. We will construct a BE scheme from Γ, having U as the universe of users and the same message and ciphertext space, in the following way:

$\mathsf{BE.PG}(1^\lambda, n)$: Run $\mathsf{CP_{ABE}.PG}$ on input 1^λ and return $pars$, which implicitly contain n and a description of the message space MsgSp and the ciphertext space CtSp of the scheme.

$\mathsf{BE.Setup}(pars)$: Run $\mathsf{CP_{ABE}.Setup}(pars)$ to obtain the master public key CP-MPK and the master secret key CP-MSK. Let BE-MPK := CP-MPK, BE-MSK := CP-MSK and output (BE-MPK, BE-MSK).

$\mathsf{BE.KeyGen}(\text{BE–MPK}, \text{BE–MSK}, i)$: Run $\mathsf{CP_{ABE}.KeyGen}(\text{BE-MPK}, \text{BE-MSK}, i)$ to obtain sk_i, the secret key corresponding to user (attribute) i.

$\mathsf{BE.Enc}$(BE-MPK, M, S): Let $\bigvee_{j \in S} j$ be the policy representing the disjunction of attributes (users) $j \in S$. Let $\mathbb{A}$ be the access structure supporting this policy. Run $\mathsf{CP_{ABE}.Enc}$(BE-MPK, $M, \mathbb{A}$) to obtain a ciphertext C.

$\mathsf{BE.Dec}$(BE-MPK, C, sk_i): Run $\mathsf{CP_{ABE}.Dec}$ on the same input to obtain either a message M or a failure symbol $\perp$.

The correctness of the BE scheme follows directly from the correctness of the CP-ABE scheme Γ used to construct it. This guarantees that if user i satisfies the disjunction policy (i.e. $i \in S$) then he can successfully decrypt.

The following result holds.

Theorem 4.11 *Let Γ be a policy-hiding and IND-CCA secure CP-ABE scheme supporting disjunction policies. Then the BE scheme constructed as above is adaptively ANO-IND-CCA secure.*

Proof. Let $\mathcal{A}$ be an adversary against the ANO-IND-CCA security of the BE scheme. We will construct an adversary $\mathcal{B}$ that will interact with $\mathcal{A}$ to break the policy-hiding and IND-CCA security of Γ. The game proceeds as follows.

Setup. The challenger $\mathcal{C}$ runs $\mathsf{CP_{ABE}.PG}(1^\lambda)$ to obtain $pars$ and $\mathsf{CP_{ABE}.Setup}(pars)$ to generate the master public key CP-MPK and the master secret key CP-MSK and gives ($pars$, CP-MPK) to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can adaptively issue key-extraction queries for any user $i \in U$. Such a query is passed on to $\mathcal{B}$, who gives it to $\mathcal{C}$ as secret-key query for attribute i. $\mathcal{C}$ will respond to each query with the private key sk_i, which is passed to $\mathcal{B}$, who then forwards it to $\mathcal{A}$ as the key for user i. $\mathcal{A}$ may also issue decryption queries of the type (C, i), where $i \in U$. $\mathcal{B}$ passes such queries to $\mathcal{C}$ and forwards $\mathcal{C}$'s response to $\mathcal{A}$.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and two equal-size target sets $S_0, S_1 \subseteq U$, such that $i \notin S_0 \triangle S_1$ for any of the queries i made in Phase 1. If for any of such queries we have that $i \in S_0 \cap S_1$ the we

require $M_0 = M_1$. $\mathcal{A}$ passes M_0, M_1, S_0, S_1 to $\mathcal{B}$. $\mathcal{B}$ creates access structures $\mathbb{A}_0$ and $\mathbb{A}_1$ expressing the disjunction of all the indices $j \in S_0$ and $k \in S_1$, respectively. It passes $M_0, M_1, \mathbb{A}_0, \mathbb{A}_1$ to $\mathcal{C}$. $\mathcal{C}$ chooses a random $b \leftarrow \{0,1\}$, computes $C^\star \leftarrow \mathsf{CP_{ABE}.Enc}(\text{CP-MPK}, M_b, \mathbb{A}_b)$ and passes $C^\star$ to $\mathcal{B}$, who in turn passes it to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to issue secret-key-extraction queries with the restriction that $i \notin S_0 \triangle S_1$ and if $i \in S_0 \cap S_1$ then we require $M_0 = M_1$. These queries are dealt with by $\mathcal{B}$ as in Phase 1. $\mathcal{A}$ can also issue decryption queries with the restriction that if $C = C^\star$ then either $i \notin S_0 \triangle S_1$ or $i \in S_0 \cap S_1$ and $M_0 = M_1$.

Guess. The adversary outputs its guess b' for b and $\mathcal{B}$ outputs the same guess.

$\mathcal{B}$ perfectly simulates the ANO-IND-CCA game for $\mathcal{A}$. Hence $\mathcal{A}$'s advantage is as it would be in the real game and by construction $\mathcal{B}$ wins whenever $\mathcal{A}$ does. □

BE and KP-ABE. Not so intuitive is to see BE as arising as a special instance of KP-ABE. In the definition of KP-ABE the policy input to the key generation algorithm is expressed by an access structure, which in our model is represented by an access tree. In order to realize BE using ABE, we select as the tree for generating the key of user i the trivial tree consisting of a single leaf for attribute i. In KP-ABE, decryption succeeds if the set of attributes S specified in the encryption phase satisfies the policy given during key generation. In particular, for the degenerate case of BE, the policy (tree) is satisfied if and only if the attribute (user) i is in S. This is precisely what is needed in the BE setting, i.e., a user i can decrypt if and only if $i \in S$, where S is the target broadcast set. Hence, BE can also be cast in the framework of KP-ABE.

We formalize this by considering a KP-ABE scheme $K = (\mathsf{KP_{ABE}.PG}, \mathsf{KP_{ABE}.Setup}, \mathsf{KP_{ABE}.KeyGen}, \mathsf{KP_{ABE}.Enc}, \mathsf{KP_{ABE}.Dec})$. Let U be the universe of attributes, where $|U| = n$. We will construct a BE scheme from K, having U as the universe of users and the same message and ciphertext space, in the following way:

$\mathsf{BE.PG}(1^\lambda, n)$: Run $\mathsf{KP_{ABE}.PG}$ on input 1^λ and return $pars$, which implicitly contain n and a description of the message space MsgSp and the ciphertext space CtSp of the scheme.

$\mathsf{BE.Setup}(pars)$: Run $\mathsf{KP_{ABE}.Setup}(pars)$ to obtain the master public key KP-MPK and the master secret key KP-MSK. Let BE-MPK := KP-MPK, BE-MSK := KP-MSK and output (BE-MPK, BE-MSK).

$\mathsf{BE.KeyGen}(\text{BE–MPK}, \text{BE–MSK}, i)$: Run $\mathsf{KP_{ABE}.KeyGen}(\text{BE-MPK}, \text{BE-MSK}, i)$ to obtain sk_i, the secret key corresponding to the single-leaf access tree i.

$\mathsf{BE.Enc}(\text{BE–MPK}, M, S)$: Run $\mathsf{KP_{ABE}.Enc}(\text{BE-MPK}, M, S)$ to obtain a ciphertext C.

$\mathsf{BE.Dec}(\text{BE–MPK}, C, sk_i)$: Run $\mathsf{KP_{ABE}.Dec}$ on the same input to obtain either a message M or a failure symbol $\perp$.

The correctness of the BE scheme follows from the correctness of the KP-ABE scheme K used to construct it since in our model a set of attributes S satisfies a single-leaf access tree if and only if the attribute associated with that leaf is in S.

The following result holds.

Theorem 4.12 *Let K be an attribute-hiding and IND-CCA secure KP-ABE scheme. Then the BE scheme constructed as above is adaptively ANO-IND-CCA secure.*

Proof. Let $\mathcal{A}$ be an adversary against the ANO-IND-CCA security of the BE scheme. We will construct an adversary $\mathcal{B}$ that will interact with $\mathcal{A}$ to break the attribute-hiding and IND-CCA security of K. The game proceeds as follows.

Setup. The challenger $\mathcal{C}$ runs $\mathsf{KP_{ABE}.PG}(1^\lambda)$ to obtain $pars$ and $\mathsf{KP_{ABE}.Setup}(pars)$ to generate the master public key KP-MPK and the master secret key KP-MSK and gives ($pars$, KP-MPK) to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can adaptively issue key-extraction queries for any user $i \in U$. Such a query is passed on to $\mathcal{B}$, who gives it to $\mathcal{C}$ as secret-key query for the degenerate single-leaf access tree i. $\mathcal{C}$ will respond to each query with the private key sk_i, which is passed to $\mathcal{B}$, who then forwards it to $\mathcal{A}$ as the key for user i. $\mathcal{A}$ may also issue decryption queries of the type (C, i), where $i \in U$. $\mathcal{B}$ passes such queries to $\mathcal{C}$ and forwards to $\mathcal{C}$'s response to $\mathcal{A}$.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and two equal-size target sets $S_0, S_1 \subseteq U$, such that $i \notin S_0 \triangle S_1$ for any of the queries i made in Phase 1. If for any of such queries we have that $i \in S_0 \cap S_1$ the we require $M_0 = M_1$. $\mathcal{A}$ passes M_0, M_1, S_0, S_1 to $\mathcal{B}$. $\mathcal{B}$ passes M_0, M_1, S_0, S_1 to $\mathcal{C}$. $\mathcal{C}$ chooses a random $b \leftarrow \{0,1\}$, computes $C^\star \leftarrow \mathsf{KP_{ABE}.Enc}(\text{KP-MPK}, M_b, S_b)$ and passes $C^\star$ to $\mathcal{B}$, who in turn passes it to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to issue secret-key-extraction queries with the restriction that $i \notin S_0 \triangle S_1$ and if $i \in S_0 \cap S_1$ then we require $M_0 = M_1$. These queries are dealt with by $\mathcal{B}$ as in Phase 1. $\mathcal{A}$ can also issue decryption queries with the restriction that if $(C, i) = (C^\star, i)$ then either $i \notin S_0 \triangle S_1$ or $i \in S_0 \cap S_1$ and $M_0 = M_1$.

Guess. The adversary outputs its guess b' for b and $\mathcal{B}$ outputs the same guess.

$\mathcal{B}$ perfectly simulates the ANO-IND-CCA game for $\mathcal{A}$. Hence $\mathcal{A}$'s advantage is as it would be in the real game and by construction $\mathcal{B}$ wins whenever $\mathcal{A}$ does. □

What we have just proved provides us with a potentially powerful tool: all the progress achieved in the area of ABE could be highly relevant to the improvement of the state of the art of BE. In practice, this approach is limited to the currently available ABE schemes. In particular, what we are looking for ideally is an *attribute(policy)-hiding* KP(CP)-ABE scheme, with *short ciphertexts*, involving *efficient* algorithms, and which is *non-selectively IND-CCA* secure in the standard model. Unfortunately, we do not have such a scheme. We hence briefly recall what are the features of the existing ABE schemes.

The ABE research community has focused on obtaining schemes allowing for policies that could be as expressive as possible [53, 32, 12, 70]. These schemes however only achieve *selective* security and provide no anonymity guarantee. Furthermore, they are not very efficient. In [54], the authors present a *threshold* CP-ABE scheme with constant-size ciphertext. Even if this limited policy could fit our purposes, the scheme unfortunately suffers from the same security drawbacks as the other ABE schemes.

In a related line of research called *predicate encryption* [58], Katz et al. address

the issue of anonymity, but the scheme the authors present is characterized by large keys and ciphertexts, linear in n as opposed to the size of S. Similar considerations go for the work in [61], where Lewko et al. introduce the notion of *functional encryption* and provide a fully secure ABE scheme, but the private key and ciphertext sizes prevent it from being a suitable candidate for the construction of BE. Follow-up work in the area (for instance, [69]) suffers from the same limitation. Indeed these recently proposed primitives, which have ABE as a special case, are very powerful and their instantiation allows for great expressiveness. However, for the use of ABE that we have in mind, namely constructing ANOBE schemes, finding anonymous and fully secure ABE schemes that support even simple policies but that are efficient is still an open problem.

4.6 Reducing the Size of the Ciphertext with Randomness Re-Use

Our main focus so far has been to show that ANOBE is achievable. Indeed we have proved that we can obtain it starting from public-key, identity-based and attribute-based encryption. Our next step is to look at how to improve on the efficiency of certain constructions, striving towards more practical ANOBE schemes.

To this end, we investigate the technique of *randomness re-use*, a rigorous and formal study of which can be found in [9] (followed by [5]). The usefulness of this technique appears obvious in the context of multi-recipient encryption, where the same "base" encryption scheme is used to send messages to multiple receivers. Re-using randomness has several practical implications: first of all, randomness is not cheap, and therefore generating less of it already represents a performance improvement. Secondly, it implies a saving in computational costs, since some components will be *re-used* in the encryption process. Finally, it allows for smaller bandwidth consumption.

While the efficiency benefits of applying this technique have always been clear, the impact of randomness re-use on the security of the scheme has required some attention. In particular, the authors of [9] provide a condition under which randomness re-use is *secure* in the setting of public-key encryption. Namely, if the

base scheme satisfies a certain property (called *reproducibility*) then sharing the randomness across ciphertext components can be done without altering the security of the scheme. Informally, a scheme is reproducible if there exists a polynomial-time algorithm that on input the public parameters, a public key pk, a ciphertext C, encryption of a message M under pk using randomness r, a public/secret key-pair (pk', sk') and a message M', returns C', the encryption of M' under pk' using the *same* randomness r. We recall the formal definition from [9].

Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a PKE scheme. Let MsgSp and RSp be the message and randomness space of Π, respectively. Let R be an algorithm that takes as input the public parameters, a public key, a ciphertext generated under such public key, a random message and a key-pair, and outputs a ciphertext. Consider the following experiment.

$\mathbf{Exp}^{\mathsf{Rep}}_{\Pi,R}(1^\lambda)$
$(pars) \leftarrow \mathsf{PKE.PG}(1^\lambda)$
$(pk, sk) \leftarrow \mathsf{PKE.KeyGen}(pars)$
$M \leftarrow \mathsf{MsgSp}$; $r \leftarrow \mathsf{RSp}$
$C = \mathsf{PKE.Enc}(pars, M, pk; r)$
$(pk', sk') \leftarrow \mathsf{PKE.KeyGen}(pars)$
$M' \leftarrow \mathsf{MsgSp}$
`return` 1 `if` $\mathsf{PKE.Enc}(pars, M', pk'; r) = R(pars, pk, C, M', pk', sk')$
`and` 0 `otherwise`.

Definition 4.13 (Reproducibility) Π *is* reproducible *if for any* λ *there is a p.t. algorithm* R *such that the experiment* $\mathbf{Exp}^{\mathsf{Rep}}_{\Pi,R}(1^\lambda)$ *outputs 1 with probability 1.*

Informally, the main reproducibility theorem [9, Theorem 1] implies that if a PKE scheme is reproducible and IND-CCA secure, then the corresponding randomness re-using, multi-recipient PKE scheme is also IND-CCA secure.

Now, a crucial observation is that effectively some of our constructions for ANOBE (we will focus on the one in Section 4.3.2) originate from a base encryption scheme which is used repeatedly to encrypt the *same* message to multiple receivers.

The relation between multi-recipient and broadcast encryption was briefly discussed in [9]: a multi-recipient encryption scheme can indeed be transformed into a BE scheme by encrypting the same message to each user in the target set, by broadcasting the whole vector of ciphertext components, and by specifying a decryption procedure that will allow each legitimate user to decrypt. Therefore it seems natural to consider randomness re-use as an efficiency-enhancing technique in the context of anonymous broadcast encryption.

It turns out that, in order to do so in a provably secure way, we have to introduce a new notion of reproducibility, called *key-less reproducibility*, better suited for a setting where anonymity is needed. In a nutshell, key-less reproducibility differs from reproducibility in that the reproduction algorithm no longer requires as input pk, the public key under which C was created. We formalize this as follows.

Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a PKE scheme. Let MsgSp and RSp be the message and randomness space of Π, respectively. Let R be an algorithm that takes as input public parameters, a ciphertext, a random message and a key-pair, and outputs a ciphertext. Consider the experiment:

$\mathbf{Exp}^{\mathsf{KLRep}}_{\Pi,R}(1^\lambda)$
$(pars) \leftarrow \mathsf{PKE.PG}(1^\lambda)$
$(pk, sk) \leftarrow \mathsf{PKE.KeyGen}(pars)$
$M \leftarrow \mathsf{MsgSp}$; $r \leftarrow \mathsf{RSp}$
$C = \mathsf{PKE.Enc}(pars, M, pk; r)$
$(pk', sk') \leftarrow \mathsf{PKE.KeyGen}(pars)$
$M' \leftarrow \mathsf{MsgSp}$
`return` 1 `if` $\mathsf{PKE.Enc}(pars, M', pk'; r) = R(pars, C, M', pk', sk')$
`and` 0 `otherwise`.

Definition 4.14 (Key-less reproducibility) Π *is* key-less reproducible *if for any* λ *there is a p.t. algorithm* R *such that the experiment* $\mathbf{Exp}^{\mathsf{KLRep}}_{\Pi,R}(1^\lambda)$ *outputs 1 with probability 1.*

We note that we can recover the original reproducibility notion simply by including a description of pk in the ciphertext C.

We now apply the technique of randomness re-use to obtain more efficient instantiations for ANOBE. Let us reconsider the generic construction presented in Section 4.3.2.

Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a key-less reproducible PKE scheme, and let $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme. We call $\mathrm{ANOBE}_{rr}^{\Pi,\Sigma}$ the scheme constructed from Π and Σ as follows.

$\mathsf{BE.PG}, \mathsf{BE.KeyGen}, \mathsf{BE.Dec}$ are as in Section 4.3.2.

$\mathsf{BE.Enc}(\text{BE-MPK}, M, S)$: to encrypt M for a receiver set $S = \{i_1, \ldots, i_\ell\} \subseteq \{1, \ldots, n\}$ of size $\ell = |S|$, generate a signature key-pair $(sigk, vk) \leftarrow \mathsf{Gen}(1^\lambda)$. Choose $r \leftarrow \mathsf{RSp}$, where RSp is the randomness space of Π. Then, for each $j = 1$ to ℓ, compute $C_j = \mathsf{PKE.Enc}(pars, M||vk, pk_{i_j}; r)$. The final BE ciphertext consists of $C = \big(vk, C_{\tau(1)}, \ldots, C_{\tau(\ell)}, \sigma\big)$, where $\sigma = \mathsf{Sign}\big(sigk, C_{\tau(1)}, \ldots, C_{\tau(\ell)}\big)$ and $\tau : \{1, \ldots, \ell\} \to \{1, \ldots, \ell\}$ is a random permutation.

Theorem 4.15 *Let Π be an IND-IK-CCA secure, weakly robust and key-less reproducible PKE scheme. Let Σ be a strongly unforgeable one-time signature scheme. Then $\mathrm{ANOBE}_{rr}^{\Pi,\Sigma}$ is adaptively ANO-IND-CCA secure.*

Proof. The proof follows precisely the proof of Theorem 4.5 up until the BE challenge ciphertext is generated. The modifications are in the following steps and apply to both Lemma 4.6 and Lemma 4.7.

1. For $j = 1$ to $k - 1$, $\mathcal{B}$ sets $C_j = R(pars, C^\star, M_1||vk^\star, pk_{\rho_j}, sk_{\rho_j})$.
2. For $j = k + 1$ to ℓ, $\mathcal{B}$ computes $C_j = R(pars, C^\star, M_0||vk^\star, pk_{\theta_j}, sk_{\theta_j})$.
3. Finally, set $C_k = C^\star$.

We observe that $\mathcal{B}$ knows all the necessary secret keys since it generated them on its own at the beginning of the simulation. The proof then continues as in Theorem 4.5.

We note that there is no further loss in the security reduction since the key-less reproducibility property of Π implies that $\mathsf{PKE.Enc}(pars, M', pk'; r) = R(pars, \mathsf{PKE.Enc}(pars, M, pk; r), M', pk', sk')$ with probability 1. □

We have shown that the key-less reproducibility of a PKE scheme guarantees that randomness can be re-used securely. We can exploit this property to compress the ANOBE ciphertexts and, depending on the concrete instantiation, significantly increase the efficiency of the scheme. More precisely, given an $\text{ANOBE}_{rr}^{\Pi,\Sigma}$ ciphertext $C = (vk, C_{\tau(1)}, \ldots, C_{\tau(\ell)}, \sigma)$, let ccc denote the *common ciphertext components* that may arise in $C_{\tau(1)}, \ldots, C_{\tau(\ell)}$ from sharing randomness across PKE components, i.e.,

$$C_{\tau(1)} = (\mathsf{ccc}, \tilde{c}_{\tau(1)}), \ldots, C_{\tau(\ell)} = (\mathsf{ccc}, \tilde{c}_{\tau(\ell)}).$$

The compressed ANOBE ciphertext will be $\tilde{C} = (vk, \mathsf{ccc}, \tilde{c}_{\tau(1)}, \ldots, \tilde{c}_{\tau(\ell)}, \sigma)$. Upon receipt, the user simply reconstitutes the original ciphertext C and runs $\mathsf{BE.Dec}$ as usual. In Section 4.7 we will discuss briefly a possible instantiation of these ideas.

4.7 Further results

The results in [62] include parts of the work presented so far and go even further in improving the efficiency of the introduced constructions. The authors provide a way to speed-up decryption and finally they present a concrete ANOBE scheme, which can be considered the state of the art. We give an overview of these results next.

4.7.1 Efficient decryption in the standard model

One major drawback of some of our constructions is that decryption takes linear time in the size of the set S. This arises from the fact that users do not know which ciphertext component is intended for them, and hence will have to perform an average of $|S|/2$ decryptions before recovering the message. Clearly this procedure is quite cumbersome. In [62], the authors present a technique allowing for constant

decryption cost and which is proved secure in the standard model (*i.e.*, without random oracles) using our enhanced security definition. So far, the only known technique – put forth by Barth et al. [6] – enabling constant-time decryption requires the random oracle heuristic in the security analysis. To eliminate the random oracle, the authors of [62] introduce a new primitive, called an *anonymous hint system*. In essence, this primitive provides a way for an encrypter to securely tell receivers which ciphertext component is intended for them, allowing them to ignore all but one ciphertext component and so decrypt more efficiently. The hint primitive, for which they provide an implementation based on the Decision-Diffie-Hellman (DDH) assumption, is defined and realized in such a way that its integration with our generic ANOBE constructions maintains compatibility with our proofs of adaptive security, allowing to generically obtain ANOBE with efficient decryption.

4.7.2 A concrete ANOBE scheme

In [62] it is also established that the Kurosawa–Desmedt (KD) [60] hybrid encryption scheme can be tweaked to have all the properties that are needed of the base PKE scheme in our constructions. The KD scheme is an ideal starting point since it is one of most efficient PKE schemes with IND-CCA security in the standard model. The authors present KD*, a modified version of the KD scheme, that is *strongly robust* (although weak robustness suffices for our purposes), assuming that its symmetric components satisfy some relatively mild conditions; *anonymous* under the DDH assumption (and, again, under mild assumptions on its symmetric components) and *key-less reproducible*. Tying everything together and using KD* as the base scheme, they obtain the *currently most efficient instantiation* of an ANOBE scheme, for which ciphertexts contain only 2 group elements and $|S|$ symmetric ciphertexts (plus a signature and a verification key). Decryption can be achieved in constant time by combining this scheme with the DDH-based hint system mentioned in the previous section, with an additional $2|S| + 1$ group elements in the ciphertext.

Table 4.1: Comparison of broadcast encryption schemes

Scheme	Security	Private key size	Ciphertext size
GW	IND-CCA	$n \cdot \|g\|$	$n + 3\|g\|$
ANOBE$_{CS}$	ANO-IND-CCA	$5 \cdot \|a\|$	$s \cdot 4\|g\| + \omega$
ANOBE$_{mG}$	ANO-IND-CCA	$1 \cdot \|a\|$	$s \cdot (\|g\| + 2\|g_T\|) + \omega$
ANOBE$_{KD^*}$	ANO-IND-CCA	$4 \cdot \|a\|$	$s \cdot \|c\| + 2\|g\| + \omega$

Let n be the size of the universe and s the size of the target set, let $a \in \mathbb{Z}_p$, $g \in \mathbb{G}$, $g_T \in \mathbb{G}_T$, let ω be the cost of transmitting a signature and a verification key and let c be a symmetric ciphertext.

In light of this result, it could be helpful to summarize our findings and compare the state of the art in this area. We do this in Table 4.1, where we consider various BE schemes and focus on three important features: security level achieved, private key storage requirements and ciphertext length. This choice is motivated by the fact that these are the most commonly observed features. We note that other parameters such as public key size, strength of security assumptions and computational costs may be considered.

The notation used in the table is specified as follows.

- GW is the scheme by Gentry and Waters [51] recalled in Section 4.1.2.1;
- ANOBE$_{CS}$ is the scheme that results from instantiating the construction in Section 4.3.2 with the Cramer–Shoup encryption scheme;
- ANOBE$_{mG}$ is the scheme that results from instantiating the construction in Section 4.4 with the multi-TA version of Gentry's IBE scheme [50, 72];
- ANOBE$_{KD^*}$ is the scheme in [62] mentioned in this section.

From Table 4.1 we can observe that the ANOBE schemes presented in this chapter enjoy increasingly improved performance, with ANOBE$_{KD^*}$ being almost competitive with the non-anonymous but very efficient GW scheme. Once again we see the challenges of maintaining a balance between security and efficiency.

4.7.3 Extensions to identity-based broadcast encryption

When giving the formal definition of a public-key BE scheme in Section 4.2 we observed that this was general enough to include the identity-based variant introduced in [39]. This flexibility allows to readily extend some of our results to the identity-based setting. Using an anonymous and weakly robust IBE scheme (see Section 2.3.2 and [2] for the relevant definitions) we can indeed obtain an *anonymous* identity-based broadcast encryption scheme with a construction similar to the one presented in Section 4.3.2. We can also extend the results in Section 4.4 by deploying a hierarchical IBE scheme, and furthermore analyze techniques for randomness re-use in this new setting. We leave details of these extensions to future work.

4.8 Conclusions

We have seen that in the context of broadcast encryption the main focus of research to date has been on reducing ciphertext size. Achieving this has entailed sacrificing *all* anonymity properties. Yet we have argued that anonymity is a *fundamental property* to strive for in broadcast encryption. One may wonder why this feature has been so neglected in the BE literature. It seems that a natural concern could be that adding anonymity would severely damage the performance of the BE scheme. Therefore, the obvious question is: how much does anonymity cost?

4.8.1 The price of anonymity

As can be seen from the details of our constructions, achieving anonymity does not add *any* cost to the encryption process compared to non-anonymous schemes (for example, [17, 51]): in our ANOBE schemes, encryption requires a number of group operations that is linear in $|S|$. As for decryption, the speed-up technique presented in [62] allows the legitimate user to recover the message in constant time. Our ciphertext size is linear in $|S|$ (and thus linear in n and of the same order of magnitude as the *true* ciphertext size in existing BE schemes). Thus one interpretation of our results is that anonymity does not "cost" anything in an asymptotic sense. Naturally,

the constants matter in practice, and reducing the constant in the ciphertext size for ANOBE to something closer to what can be achieved in the non-anonymous setting is a major open problem. However, we reiterate that reducing the *true* size of ciphertexts below linear in n in either the anonymous or non-anonymous setting is impossible.

4.8.2 Open problems

With the aim of highlighting the importance of anonymity in broadcast encryption, we have formally defined the notion of anonymous broadcast encryption (ANOBE) and given several constructions for this primitive. We have also shown how these constructions can be improved via randomness re-use techniques (to reduce the ciphertext size and the computational costs of encryption) and pointed to further results developed in [62]. Our constructions set a yardstick by which future ANOBE schemes can be measured. Much work still needs to be done in this area, from improving the efficiency of ANOBE schemes to considering all the additional properties that can be found in standard BE, such as traitor tracing, revocation, dynamism of users joining the system, and realising them in the anonymous setting. There is still a gap between the sizes of ciphertexts in state-of-the-art BE schemes and our ANOBE schemes. This gap is hidden in the constants in an asymptotic evaluation of ciphertext size (when the true size of ciphertexts is measured) but is nevertheless significant in practice. A major challenge, then, is to further reduce the size of ciphertexts in ANOBE, whilst maintaining its full anonymity properties.

Anonymity is of crucial importance in modern cryptography, in particular for primitives which have immediate real-life applications, such as BE. We hence believe that future work in this area should consider anonymity as one of the primary goals.

Chapter 5

Time-Specific Encryption

Contents

This chapter introduces and explores the concept of time-specific encryption (TSE), a newly designed primitive which allows the sender to specify in what time interval a ciphertext can be decrypted. We present several flavours of TSE and provide generic constructions to achieve them securely in the standard model. Finally, we suggest applications for our new primitive, and discuss its relationships with existing primitives, such as timed-release encryption and broadcast encryption. The work in this chapter, fruit of a collaboration with Kenneth G. Paterson, was published as [71] at the international cryptographic conference Security and Cryptography for Networks 2010, and won the Best Paper Award.

5.1 Introduction

5.1.1 Time and encryption

Time has always played an important role in communication. Information can become useless after a certain point, sensitive data may not be released before a particular time, or we may wish to enable access to information for only a limited period of time. In this context, being able to specify *when* a ciphertext can be decrypted by a receiver is a useful and interesting property. Indeed, the idea of sending a message "into the future", i.e. encrypting a message so that it cannot be decrypted *before* a pre-determined release time, has generated quite some interest in the cryptographic community due to its many real-world applications such as electronic auctions, press releases, etc. Timed-release encryption (TRE) precisely addresses this problem.

5.1.1.1 Timed-release encryption

Since its introduction by May [65], TRE has enjoyed significant development (see [26, 27, 56, 41, 33], to name a few).

In TRE there are typically two potential approaches: the *time-lock puzzle* approach and the *time server* approach. In the former [78, 63, 20] the receiver is required to make a substantial computational effort to solve a certain problem before recovering the message. This technique has some obvious practical limitations since it is rather expensive for the receiver and also it guarantees only a *delay* in the decryption time rather than providing the precise functionality of being able to decrypt after a specified time. The time server approach overcomes these limitations by introducing an entity, the time server (TS), which provides a common and absolute time reference for all users in the system. In this setting, TS broadcasts what we will call a time instant key (TIK) at each time unit and, ideally, it should not interact with either the sender or the receiver. Most TRE literature follows this approach.

A number of results have been achieved in this area, from providing scalable and non-interactive systems [26, 27], to equipping TRE schemes with the additional

functionality called pre-open capability [56, 41], which allows the receiver to decrypt the ciphertext *before* the release time at the sender's discretion. In all these systems the key feature though is that at any point *after* the release time t the receiver should be able to decrypt. However, in practice, in existing TRE schemes, successful decryption occurs *only* with the TIK broadcast by TS at time t: any other key broadcast after time t will not be useful to recover the message. Therefore these schemes suffer from the limitation that some back-up mechanism must be provided in case the receiver misses the key broadcast by the server at the release time. Typically in the literature, it is assumed that the time server (or some other agency) will make old keys available on a public server. Clearly this may be inconvenient and would require additional infrastructure on top of the broadcast capability. The consideration of this practical issue was key to the introduction and development of a new cryptographic primitive which we call time-specific encryption (TSE).

5.1.1.2 Time-specific encryption

In time-specific encryption (TSE), as with TRE, a time server broadcasts a key at the beginning of each time unit, a time instant key (TIK). But now, the sender of a message can specify any *time interval* during the encryption process; the receiver can decrypt to recover the message only if it has a TIK that corresponds to a time in that interval.

More specifically, we consider a setting in which we have a (semi-)trusted time server (TS). TS broadcasts a TIK k_t at each time unit or "tick" of its clock, t, where $0 \leq t \leq T-1$. This TIK is available to all users, and we implicitly assume that it contains a description of t. A sender can specify any interval $[t_0, t_1]$, where $t_0 \leq t_1$, when encrypting a plaintext M to form a ciphertext C. In *Plain* TSE, we wish to achieve the property that C can only be decrypted by a receiver to recover M if the receiver is in possession of a TIK k_t for some t with $t \in [t_0, t_1]$. Notice that we cannot enforce the property that the receiver can only decrypt during the decryption time interval (DTI) $[t_0, t_1]$, since a receiver can always obtain an appropriate TIK and then use it at any time later on to perform decryption. Achieving this stronger notion could be done using trusted hardware, for example. Yet, as we discuss below, TSE has several intriguing applications exploiting its defining property that a receiver

must obtain a suitable TIK before being able to decrypt.

First of all, TSE generalizes TRE: indeed, TRE represents the special case of TSE in which the sender can specify only intervals of the form $[t, t]$. As for the issue of missed keys described above, TSE provides an elegant solution: if the sender specifies an interval of the form $[t, T-1]$ (where $T-1$ is the maximum time supported by the scheme) then a receiver can decrypt using *any* TIK $k_{t'}$ broadcast by the time server at time $t' \geq t$. We note that the use of tree techniques to achieve this capability was sketched in [26, 33], but without any formal security analysis. TSE, then, provides a useful extension of TRE that can be exploited in any of the many applications that have already been proposed for TRE in the literature, which, as mentioned before, include electronic auctions, key escrow, on-line gaming, timed release of information such as press releases, and so on.

However, TSE is more flexible than this in the range of applications that it supports. For example, the encrypting party may specify an interval of the form $[0, t]$, meaning that a receiver can decrypt the ciphertext as soon as it is received and a TIK has been obtained, but only up to time t. After this time, TIKs issued by the time server will not help in decryption. Yet, a user might obtain a useful TIK from some other user in the system, so this application of TSE only makes sense in situations where users have a vested interest in not sharing TIKs with one another, such as in situations where users are in competition with one another. For example, the ciphertext may encrypt a ticket for accessing a service that is valid up to time t. More generally, TSE can be used to support any application in which a user benefits from accessing plaintext in a timely manner, and where the utility of a TIK becomes limited shortly after its broadcast time. We sketch an example of such an application in the domain of entity authentication next.

Consider a typical time-stamp based network authentication protocol, in which entities A and B share a symmetric key K and in which A sends B messages of the form $\mathrm{MAC}_K(T||B)$ where T is the current time (at A) and MAC_K denotes a secure MAC algorithm using the key K. Such a protocol requires roughly synchronised clocks, and B needs to allow a "window of acceptance" for values T in A's messages, to cater for any loss of accuracy in synchronisation and network delay. In turn, this means that B needs to keep a log of recently received messages to prevent replays by

an attacker during the window. How can TSE help? Suppose B generates a nonce N, encrypts it using a TSE scheme with an interval $[t_0, t_1]$, where $t_1 - t_0$ is equal to the width of a suitable window of acceptance (to cater for network delay and clock drift between A and B), and broadcasts the resulting ciphertext. Now A's ability to send a message of the form $\mathrm{MAC}_K(N||B)$ to B before time t_1 is a proof that A obtained a TIK k_t during the interval $[t_0, t_1]$ and decrypted to obtain the nonce N. Thus B obtains a proof of liveness of A within a certain window of acceptance, so authenticating A to B. This basic protocol can be extended in a number of ways. For example, B's ciphertexts can be pre-distributed to A, giving A a set of tokens which she can use to authenticate to B during specified time intervals. We can also adapt the basic scheme to use pseudo-randomly generated nonces, so saving state at B. We can modify it to provide key transport, by replacing the MAC with an authenticated encryption scheme and including a session key in A's message. We can also add mutual authentication in obvious ways. What is notable about the protocol design is that we no longer require synchronized clocks, and we have a window of acceptance for responses by design. These features arise from the use of TSE.

5.1.2 Further related work

Range queries over encrypted data and related ideas: Shi et al. [84] proposed schemes enabling multi-dimensional range queries over encrypted data (MRQED). In the one-dimensional version of this primitive, data is associated with a single value and is encrypted with respect to that value, while users are equipped with keys enabling them to decrypt data whose values are in a given range. In contrast, in TSE, encryption is performed with respect to a range, while the time server makes available keys specific to a particular time value. Thus, our notion of Plain TSE is precisely equivalent to the notion of dual MRQED, also introduced but not formalised by Shi et al. [84]. We note that [84] gives a construction which builds a dual MRQED scheme from a normal MRQED scheme, but this seems to involve a doubling of dimension and, therefore, a significant loss of efficiency, a problem from which our constructions do not suffer. In addition, in our work, we give constructions achieving non-selective security against chosen-ciphertext attackers, whereas [84] only considers selective security notions and chosen-plaintext attackers in any detail (and then in the MRQED setting rather than its dual). Moreover,

we consider plain, public-key and identity-based settings, whereas [84] only handles what amounts to the plain setting. In work related to that of Shi et al., Srivatsa et al. [87] introduced trust-and-identity based encryption (TIBE). Replacing "trust" with time in TIBE, and ignoring the identity-based aspects, we recover a special case of MRQED of dimension 1, but handling only intervals of the form $[t, T-1]$. Another related idea is sketched in [14], where it is shown how to transform a hierarchical identity-based encryption scheme into an encryption system that can send messages into the future. Translated into the language of this paper, this yields a Plain TSE scheme that can only support intervals of the form $[t, T-1]$. Unfortunately, because of specific details of the construction used, this approach does not seem capable of being extended to support more general intervals.

ABE and PE: TSE can be seen as arising from a special case of ciphertext-policy attribute-based encryption (ABE) [53, 12, 54], itself a special case of predicate encryption (PE) [58], for a class of policies which express interval membership and attributes which express specific times, and with the time server playing the role of Attribute Authority. We note that most work on ABE and PE to date is limited to the "selective-attribute" case. In the context of TSE, converting to a non-selective security model would incur a cost of roughly $O(\frac{1}{T^2})$ in the tightness of the security reduction. However, our constructions for TSE in this chapter already achieve fully adaptive security in the standard model with a *tight* reduction to the security of the IBE scheme used in the specific instantiation. On the other hand, ABE schemes that do achieve full security, such as [61] and follow-up work, suffer from low performance, as discussed in Section 4.5.

Broadcast encryption: As discussed in Chapter 4, broadcast encryption (BE) is a cryptographic primitive designed to address the issue of broadcasting a message to an arbitrary subset drawn from a universe of users. Although conceptually opposites (in TSE the *keys* are broadcast while the message is sent beforehand), it can be shown that a BE scheme can be used to construct a Plain TSE scheme: assume the users in the BE scheme can be labeled with elements from $[0, T-1]$, consider a DTI as the target subset of addressed users in the BE encryption algorithm, and broadcast the secret key for user with label t at time t. In Section 5.4.1 we show that a correct

Plain TSE scheme results from this transformation, and moreover, that the security of this scheme can be related to standard security properties of the underlying BE scheme. There are however some *caveats* to this approach, which we discuss in detail in Section 5.4.1. In particular, the advantages and shortcomings of BE over our approach to the realisation of Plain TSE, as developed in Section 5.3.1, can be cast in a framework of trade-offs between the sizes of public parameters, secret keys and ciphertexts, together with computational costs and strength of security achieved.

Temporal access control: Significant related work in the symmetric-key setting exists in the area of cryptographically-enabled "temporal access control", see for example [38] and the references therein. In this line of work, a key is associated with each time "point", and a key assignment scheme is used to ensure that an authorized user is able to derive keys for all the points contained in a designated interval. Such schemes generally require the publication of rather large amounts of information in order to achieve the desired functionality, but do allow efficient derivation of keys associated with time points. In contrast, we use public-key techniques, achieving small public parameters and greater flexibility in operation, at the cost of increased computation.

5.1.3 Our contributions

In this chapter we first develop the basic flavour of TSE, namely Plain TSE, as described in the previous section. We then extend Plain TSE to the public-key and identity-based settings, where receivers are additionally equipped with secret keys and either public keys or identities, and where decryption now requires the use of the relevant secret key as well as an appropriate TIK. This provides protection against a curious time server, as well as ensuring that a ciphertext is decryptable only by a specified party. We introduce security models for the plain, public-key and identity-based settings, considering both chosen-plaintext and chosen-ciphertext adversaries.

We provide constructions for schemes in the different settings. Firstly, we build

Plain TSE schemes by adapting ideas of [84, 87] which themselves employ identity-based and tree techniques. Secondly, we show how to combine Plain TSE with public-key and identity-based encryption schemes to obtain chosen-plaintext secure TSE schemes in the public-key and identity-based settings. Thirdly, we show how to adapt the CHK transformation [25] to the TSE setting, obtaining a generic construction for a chosen-ciphertext secure TSE scheme in the public-key setting from a chosen-plaintext secure, identity-based TSE scheme. Our focus is on providing generic constructions that are secure in the standard model. Naturally, more efficient constructions and concrete schemes can be obtained by working in the random oracle model (ROM), and we sketch such constructions where appropriate. In our closing section, we discuss possible extensions of our ideas and areas for future work.

5.2 Definitions and Security Notions for TSE

5.2.1 Notation

Throughout the chapter we will consider time as a discrete set of time units, regarding these as integers between 0 and $T-1$, where T represents the number of time units supported by the system. We denote by $[t_0, t_1]$, where $t_0 \leq t_1$, the interval containing all time units from t_0 to t_1 inclusive.

5.2.2 Plain TSE

We start by providing the definition and the security models for the basic form of time-specific encryption, namely Plain TSE.

Definition 5.1 (Plain TSE) *Let* $\mathsf{TSp} = [0, T-1]$ *be the time space supported by the scheme and let the parties involved be the Time Server (TS), the sender (S) and a user (U). A* Plain TSE scheme *is defined by four algorithms, which are as follows.*

Plain.Setup: Run by TS, this algorithm takes as input the security parameter 1^λ and T, and returns a master public key TS-MPK (which includes a description

of the system's parameters, the message space MsgSp and the ciphertext space CtSp) and a master secret key TS-MSK. We write $(\text{TS-MPK}, \text{TS-MSK}) \leftarrow \mathsf{Plain.Setup}(1^\lambda, T)$.

Plain.TIK-Ext: Run by TS, this is a key extraction algorithm that on input TS-MPK, TS-MSK and a time instant $t \in \mathsf{TSp}$ outputs the time instant key (TIK) k_t. We write this as $k_t \leftarrow \mathsf{Plain.TIK\text{-}Ext}(\text{TS-MPK}, \text{TS-MSK}, t)$. This is broadcast by TS at time t.

Plain.Enc: Run by S, this is an encryption algorithm that on input TS-MPK, a message $M \in \mathsf{MsgSp}$ and a decryption time interval (DTI) $[t_0, t_1] \subseteq \mathsf{TSp}$ returns a ciphertext $C \in \mathsf{CtSp}$. We write this as $C \leftarrow \mathsf{Plain.Enc}(\text{TS-MPK}, M, [t_0, t_1])$. The ciphertext C is broadcast by S to all users.

Plain.Dec: Run by U, this is a decryption algorithm that on input TS-MPK, a ciphertext C and a key k_t returns either a message $M \in \mathsf{MsgSp}$ or a failure symbol $\perp$. We write this as $\mathsf{Plain.Dec}(\text{TS-MPK}, C, k_t) = M$ or $\perp$.

These algorithms are required to satisfy the following correctness property*: For every λ and every T, for every TS-MPK, TS-MSK output by $\mathsf{Plain.Setup}$, for every message $M \in \mathsf{MsgSp}$, every time instant $t \in \mathsf{TSp}$ and time interval $[t_0, t_1] \subseteq \mathsf{TSp}$, if $k_t \leftarrow \mathsf{Plain.TIK\text{-}Ext}(\text{TS-MPK}, \text{TS-MSK}, t)$ and if $C \leftarrow \mathsf{Plain.Enc}(\text{TS-MPK}, M, [t_0, t_1])$ then $\mathsf{Plain.Dec}(\text{TS-MPK}, C, k_t)$ returns M when $t \in [t_0, t_1]$ and $\perp$ otherwise.*

We note that for this primitive, as well as for the other flavours of TSE, the correctness property requires the decryption algorithm to return $\perp$ if an inappropriate TIK is used (i.e. a TIK not in the DTI). Including this condition here is largely motivated by the ABE and PE literature, where correctness properties deal with decryption with "incorrect" keys in a similar way. An alternative approach could be to not specify this as a correctness requirement, but to consider it as a security issue, and hence defer its handling to the security property of robustness.

We define the security notion of *indistinguishability under chosen-plaintext attacks* (IND-CPA) for a Plain TSE scheme $P = (\mathsf{Plain.Setup}, \mathsf{Plain.TIK\text{-}Ext}, \mathsf{Plain.Enc}, \mathsf{Plain.Dec})$ as follows.

IND-CPA security game for Plain TSE

Setup. The challenger $\mathcal{C}$ runs $\mathsf{Plain.Setup}(1^\lambda, T)$ to obtain the master public key TS-MPK and the master secret key TS-MSK and gives TS-MPK to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can adaptively issue TIK extraction queries to an oracle $\mathcal{O}^{\text{TS-MSK}}$ for any time $t \in \mathsf{TSp}$. The oracle will respond to each query with $k_t \leftarrow \mathsf{Plain.TIK\text{-}Ext}(\text{TS-MPK}, \text{TS-MSK}, t)$.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and a time interval $[t_0, t_1] \subseteq \mathsf{TSp}$ with the restriction that $t \notin [t_0, t_1]$ for all of the TIK extraction queries t in Phase 1. $\mathcal{A}$ passes $M_0, M_1, [t_0, t_1]$ to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0, 1\}$ and computes $C^\star \leftarrow \mathsf{Plain.Enc}(\text{TS-MPK}, M_b, [t_0, t_1])$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to have access to a TIK extraction oracle $\mathcal{O}^{\text{TS-MSK}}$, with the same restriction we have in the Challenge phase.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage in the above game as $\mathbf{Adv}^{\text{IND-CPA}}_{\mathcal{A},P}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 5.2 (IND-CPA) *A Plain TSE scheme $P = (\mathsf{Plain.Setup}, \mathsf{Plain.KeyExt}, \mathsf{Plain.Enc}, \mathsf{Plain.Dec})$ is* indistinguishable under chosen-plaintext attacks *(or is* IND-CPA secure*) if all p.p.t. adversaries have at most negligible advantage in the above game.*

We can extend this definition to address IND-CCA security by considering, in addition, a decryption oracle that acts as follows. On input the pair (C, t), where C is a ciphertext and $t \in \mathsf{TSp}$, it passes t to the TIK extraction oracle, which will respond with k_t. The decryption oracle will then compute $\mathsf{Plain.Dec}(\text{TS-MPK}, C, k_t)$ and return either a message M or a failure symbol $\perp$ to the adversary. The decryption oracle can be adaptively issued queries (C, t) in both Phase 1 and Phase 2, but in the latter phase with the restriction that if $C^\star$ and $[t_0, t_1]$ are the challenge ciphertext and time interval, respectively, then the adversary cannot make a decryption query

(C, t) where $C = C^\star$ and $t \in [t_0, t_1]$. This restriction prevents the adversary from winning the game trivially.

5.2.3 Public-key TSE

We now define another version of TSE called public-key TSE (PK-TSE) in which the sender S encrypts a message M to a *particular* receiver R who holds a key-pair (pk, sk). The message M has an associated decryption time interval $[t_0, t_1]$ specified by S. R can decrypt if he has his private key sk and a time instant key (TIK) issued by TS between time t_0 and time t_1. We provide a formal definition of PK-TSE next.

Definition 5.3 *Let* $\mathsf{TSp} = [0, T-1]$ *be the time space supported by the scheme and let the parties involved be the Time Server (TS), the sender (S) and the receiver (R). A* PK-TSE scheme *is defined by five algorithms, which are as follows.*

PK.Setup: Run by TS, this algorithm takes as input the security parameter 1^λ and T, and returns a master public key TS-MPK (which includes a description of the system's parameters, the message space MsgSp and the ciphertext space CtSp) and a master secret key TS-MSK. We write $(\text{TS-MPK}, \text{TS-MSK}) \leftarrow \mathsf{PK.Setup}(1^\lambda, T)$.

PK.TIK-Ext: Run by TS, this is a key extraction algorithm that on input TS-MPK, TS-MSK and a time instant $t \in \mathsf{TSp}$ outputs the TIK k_t. We write this as $k_t \leftarrow \mathsf{PK.TIK\text{-}Ext}(\text{TS-MPK}, \text{TS-MSK}, t)$. This is broadcast by TS at time t.

PK.KeyGen: Run by R, this is a key generation algorithm that on input the security parameter 1^λ outputs a key-pair (pk, sk). We write this as $(pk, sk) \leftarrow \mathsf{PK.KeyGen}(1^\lambda)$.

PK.Enc: Run by S, this is an encryption algorithm that on input TS-MPK, a message $M \in \mathsf{MsgSp}$, a decryption time interval (DTI) $[t_0, t_1] \subseteq \mathsf{TSp}$ and a public key pk returns a ciphertext $C \in \mathsf{CtSp}$. We write this as $C \leftarrow \mathsf{PK.Enc}(\text{TS-MPK}, M, [t_0, t_1], pk)$.

PK.Dec: Run by R, this is a decryption algorithm that on input TS-MPK, a ciphertext C, a TIK k_t and a private key sk returns either a message $M \in \mathsf{MsgSp}$ or $\perp$. We write this as $\mathsf{PK.Dec}(\text{TS-MPK}, C, k_t, sk) = M$ or $\perp$.

These algorithms are required to satisfy the following correctness property*: For every λ and every T, for every TS-MPK, TS-MSK output by PK.Setup, for every message $M \in$ MsgSp, every time instant $t \in$ TSp and time interval $[t_0, t_1] \subseteq$ TSp, and every (pk, sk) output by PK.KeyGen, if $k_t \leftarrow$ PK.TIK-Ext$(TS\text{-}MPK, TS\text{-}MSK, t)$ and if $C \leftarrow$ PK.Enc$(TS\text{-}MPK, M, [t_0, t_1], pk)$ then PK.Dec$(TS\text{-}MPK, C, k_t, sk)$ returns M when $t \in [t_0, t_1]$ and $\perp$ otherwise.*

To model the security of a PK-TSE scheme, we consider (as in [41] for the TRE case) the following kinds of adversaries:

- A curious TS who holds TS-MSK and wishes to break the confidentiality of the message.
- An intended but curious receiver who wishes to decrypt the message outside of the appropriate decryption time interval.

We observe that security against an outside adversary (who is not the intended recipient and does not know TS-MSK) trivially follows from security against either type of adversary considered above.

Remark 2 In Plain TSE there is only one type of adversary, i.e. the *curious user*, since there is no specific receiver and TS can trivially decrypt any message.

We define the security notion of *indistinguishability under chosen-plaintext attacks against a curious TS* (IND-CPA$_{TS}$) for a PK-TSE scheme $PK = (\mathsf{PK.Setup}, \mathsf{PK.TIK\text{-}Ext}, \mathsf{PK.KeyGen}, \mathsf{PK.Enc}, \mathsf{PK.Dec})$ as follows[1].

[1]We note that in defining the security models for PK-TSE we consider, for simplicity, a single-user setting. This also justifies the fact that a separate parameter generation algorithm is not required in the definition of PK-TSE.

IND-CPA$_{TS}$ security game for PK-TSE

Consider the following game, which we call Game$_{\text{PK-TS}}$.

Setup. The challenger $\mathcal{C}$ runs $\mathsf{PK.Setup}(1^\lambda, T)$ to obtain the master public key TS-MPK and the master secret key TS-MSK and runs $\mathsf{PK.KeyGen}(1^\lambda)$ to get a pair (pk, sk). $\mathcal{C}$ gives (TS-MPK, TS-MSK, pk) to the adversary $\mathcal{A}$.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and a time interval $[t_0, t_1] \subseteq \mathsf{TSp}$. $\mathcal{A}$ passes $M_0, M_1, [t_0, t_1]$ to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0, 1\}$ and computes $C^\star \leftarrow \mathsf{PK.Enc}(\text{TS-MPK}, M_b, [t_0, t_1], pk)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage in the above game as $\mathbf{Adv}^{\text{IND-CPA}_{TS}}_{\mathcal{A},PK}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 5.4 (IND-CPA$_{TS}$) *We say that a PK-TSE scheme is* IND-CPA$_{TS}$ secure *if all p.p.t. adversaries have at most negligible advantage in Game$_{PK\text{-}TS}$.*

We can extend this definition to address IND-CCA$_{TS}$ security by considering, in addition, a decryption oracle that on input the pair (C, t), where C is a ciphertext and $t \in \mathsf{TSp}$, returns either a message M or failure symbol $\perp$ to the adversary. The decryption oracle can be adaptively issued queries (C, t) before and after the Challenge phase, but with the obvious restriction that the adversary cannot make queries of the form (C, t) where $C = C^\star$ and $t \in [t_0, t_1]$ after the Challenge phase.

For both IND-CPA and IND-CCA settings, it is possible to model security games in which the public keys are maliciously generated. In our work, however, we will consider only honest key-generation, leaving such extension for future research.

We now address IND-CPA security against a curious receiver (IND-CPA$_{CR}$).

IND-CPA$_{CR}$ security game for PK-TSE

Consider the following game, which we call Game$_{\text{PK-CR}}$.

Setup. The challenger $\mathcal{C}$ runs $\mathsf{PK.Setup}(1^\lambda, T)$ to obtain the master public key TS-MPK and the master secret key TS-MSK, and runs $\mathsf{PK.KeyGen}(1^\lambda)$ to get a key-pair (pk, sk). $\mathcal{C}$ gives (TS-MPK, pk, sk) to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can adaptively issue TIK extraction queries for any time $t \in \mathsf{TSp}$. The challenger responds to each query with $k_t \leftarrow \mathsf{PK.TIK\text{-}Ext}(\text{TS-MPK}, \text{TS-MSK}, t)$.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and a time interval $[t_0, t_1] \subseteq \mathsf{TSp}$ with the restriction that $t \notin [t_0, t_1]$ for all of the TIK extraction queries t in Phase 1. $\mathcal{A}$ passes $M_0, M_1, [t_0, t_1]$ to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0, 1\}$ and computes $C^\star \leftarrow \mathsf{PK.Enc}(\text{TS-MPK}, M_b, [t_0, t_1], pk)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to make TIK extraction queries with the same restriction we have in the Challenge phase.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage in the above game as $\mathbf{Adv}^{\text{IND-CPA}_{CR}}_{\mathcal{A},PK}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 5.5 (IND-CPA$_{CR}$) *We say that a PK-TSE scheme is* IND-CPA$_{CR}$ secure *if all p.p.t. adversaries have at most negligible advantage in Game$_{PK\text{-}CR}$.*

We observe that this chosen-plaintext notion of security is sufficient to capture all realistic attacks that can be mounted by a curious receiver, so that a chosen-ciphertext notion of security is not required for curious receivers. Indeed such a receiver would be the only entity in possession of its private key, and so would be the only entity that could implement a general decryption oracle in practice. We show next that such an oracle would either allow the receiver to win trivially or would not provide any advantage over just having access to the TIK extraction oracle in the chosen-plaintext setting. More precisely, suppose $C^\star$ is the challenge ciphertext and $[t_0, t_1]$ the challenge interval. To handle decryption queries of the form (C, t) where $C \neq C^\star$ and $t \in [t_0, t_1]$, the receiver would need to use its private key in combination with a key k_t obtained from TS. But then having such a key k_t would also allow the curious receiver to decrypt $C^\star$ and so win the security game trivially. For any other decryption query (C, t), where now $t \notin [t_0, t_1]$, the curious

receiver would be able to obtain the key k_t by making a TIK extraction query, and then use k_t together with its private key sk to decrypt c. So, in this situation, the curious receiver would gain no advantage from having access to a decryption oracle.

We formalize the unified notion of indistinguishability under chosen-plaintext attacks for PK-TSE as follows.

Definition 5.6 (IND-CPA) *We say that a PK-TSE scheme* $PK = (\mathsf{PK.Setup}, \mathsf{PK.TIK\text{-}Ext}, \mathsf{PK.KeyGen}, \mathsf{PK.Enc}, \mathsf{PK.Dec})$ *is* IND-CPA secure *if it is both is IND-CPA*$_{TS}$ *and IND-CPA*$_{CR}$ *secure.*

5.2.4 Identity-based TSE

We finally consider an identity-based version of TSE, called ID-TSE, in which the sender encrypts a message M under the *identity* of a particular receiver. The message M has an associated decryption time interval $[t_0, t_1]$ specified by the sender. The receiver can decrypt if he holds the private key associated with his identity (as issued by a (semi-)trusted authority TA) and a time instant key (TIK) issued by TS between time t_0 and time t_1. We now provide the formal definition of ID-TSE.

Definition 5.7 *Let* $\mathsf{TSp} = [0, T-1]$ *be the time space supported by the scheme and let the parties involved be the Time Server (*TS*), a trusted authority (*TA*), the sender (*S*) and the receiver (*R*). An* ID-TSE scheme *is defined by six algorithms, which are as follows.*

TS.Setup: Run by TS, this algorithm takes as input the security parameter 1^λ and T, and returns a master public key TS-MPK (which includes a description of the system's parameters, the message space MsgSp, the identity space IdSp and the ciphertext space CtSp) and a master secret key TS-MSK. We write $(\text{TS-MPK}, \text{TS-MSK}) \leftarrow \mathsf{TS.Setup}(1^\lambda, T)$.

ID.Setup: Run by TA, this algorithm takes as input the security parameter 1^λ and outputs the master public key ID-MPK and the master secret key ID-MSK. We write this as $(\text{ID-MPK}, \text{ID-MSK}) \leftarrow \mathsf{ID.Setup}(1^\lambda)$.

ID.TIK-Ext: Run by TS, this is a key extraction algorithm that on input TS-MPK, TS-MSK and a time instant $t \in \mathsf{TSp}$ outputs the time instant key (TIK) k_t. We write this as $k_t \leftarrow \mathsf{ID.TIK\text{-}Ext}(\text{TS-MPK}, \text{TS-MSK}, t)$. This is broadcast by TS at time t.

ID.Key-Ext: Run by TA, this is a key extraction algorithm that on input ID-MPK, ID-MSK and an $id \in \mathsf{IdSp}$ outputs the secret key sk_{id} corresponding to id. We write $sk_{id} \leftarrow \mathsf{ID.Key\text{-}Ext}(\text{ID-MPK}, \text{ID-MSK}, id)$.

ID.Enc: Run by S, this is an encryption algorithm that on input TS-MPK, ID-MPK, a message $M \in \mathsf{MsgSp}$, a decryption time interval (DTI) $[t_0, t_1] \subseteq \mathsf{TSp}$ and an identity $id \in \mathsf{IdSp}$ returns a ciphertext $C \in \mathsf{CtSp}$. We write this as $C \leftarrow \mathsf{ID.Enc}(\text{TS-MPK,ID-MPK}, M, [t_0, t_1], id)$.

ID.Dec: Run by R, this is a decryption algorithm that on input TS-MPK, ID-MPK, a ciphertext C, a TIK k_t and a private key sk_{id} returns either a message $M \in \mathsf{MsgSp}$ or a failure symbol $\perp$. We write this as $\mathsf{ID.Dec}(\text{TS-MPK}, \text{ID-MPK}, C, k_t, sk_{id}) = M$ or $\perp$.

These algorithms are required to satisfy the following correctness property*: For every λ and every T, for every TS-MPK, TS-MSK output by* $\mathsf{TS.Setup}$*, for every ID-MPK, ID-MSK output by* $\mathsf{ID.Setup}$*, for every message $M \in \mathsf{MsgSp}$, every time instant $t \in \mathsf{TSp}$, every time interval $[t_0, t_1] \subseteq \mathsf{TSp}$, and for every $id \in \mathsf{IdSp}$, if $sk_{id} \leftarrow \mathsf{ID.Key\text{-}Ext}(\text{ID-MPK}, \text{ID-MSK}, id)$, if $k_t \leftarrow \mathsf{ID.TIK\text{-}Ext}(\text{TS-MPK}, \text{TS-MSK}, t)$ and if $C \leftarrow \mathsf{ID.Enc}(\text{TS-MPK}, \text{ID-MPK}, M, [t_0, t_1], id)$ then $\mathsf{ID.Dec}(\text{TS-MPK}, \text{ID-MPK}, C, k_t, sk_{id})$ returns M when $t \in [t_0, t_1]$ and $\perp$ otherwise.*

We now model the security of an ID-TSE scheme. Since we are in an ID-based setting we will consider adversaries that interact with multiple users. We consider the following two types of adversaries:

- A curious TS who holds TS-MSK, and hence can derive TIKs for any time t, and wishes to break the confidentiality of the message.
- A curious TA who holds ID-MSK, and hence can derive private keys for any identity id, and wishes to break the confidentiality of the message.

We note that the latter adversary is more powerful than the natural analogue of the curious receiver in the ID setting, so we do not give a separate security definition for the curious receivers.

We first define IND-CPA security against a curious TS, which we denote IND-CPA$_{TS}$.

IND-CPA$_{TS}$ security game for ID-TSE

Consider the following game, which we call Game$_{\text{ID-TS}}$.

Setup. The challenger $\mathcal{C}$ runs $\mathsf{TS.Setup}(1^\lambda, T)$ to obtain the master public key TS-MPK and the master secret key TS-MSK, and runs $\mathsf{ID.Setup}(1^\lambda)$ to generate ID-MPK and ID-MSK. $\mathcal{C}$ gives (TS-MPK, TS-MSK, ID-MPK) to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can adaptively issue queries to a secret key extraction oracle to get the secret keys corresponding to identities id of its choice. $\mathcal{C}$ will respond with $sk_{id} \leftarrow \mathsf{ID.Key\text{-}Ext}(\text{ID-MPK}, \text{ID-MSK}, id)$.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and a time interval $[t_0, t_1] \subseteq \mathsf{TSp}$ and $id^\star \in \mathsf{IdSp}$ with the restriction that $id^\star$ was not queried to the secret key extraction oracle in Phase 1. $\mathcal{A}$ passes $M_0, M_1, [t_0, t_1], id^\star$ to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0, 1\}$ and computes $C^\star \leftarrow \mathsf{ID.Enc}(\text{TS-MPK}, \text{ID-MPK}, M_b, [t_0, t_1], id^\star)$. $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to make secret key extraction queries with the same restriction we have in the Challenge phase.

Guess. The adversary outputs its guess b' for b.

We can easily modify this game to obtain a *selective-id* security notion (as per Definition 2.6), by requiring that at the beginning of the game, before the Setup phase, the adversary will output the identity $id^\star$ on which it wishes to be challenged.

We define $\mathcal{A}$'s advantage in the above game as $\mathbf{Adv}^{\text{IND-CPA}_{TS}}_{\mathcal{A},ID}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 5.8 (IND-CPA$_{TS}$) *We say that an ID-TSE scheme is* IND-CPA$_{TS}$ secure *if all p.p.t. adversaries have at most negligible advantage in Game$_{ID\text{-}TS}$.*

We now define IND-CPA security against a curious TA (IND-CPA$_{TA}$).

IND-CPA$_{TA}$ security game for ID-TSE

Consider the following game, which we call Game$_{\text{ID-TA}}$.

Setup. The challenger $\mathcal{C}$ runs $\mathsf{TS.Setup}(1^\lambda, T)$ to obtain the master public key TS-MPK and the master secret key TS-MSK, and runs $\mathsf{ID.Setup}(1^\lambda)$ to generate ID-MPK and ID-MSK. $\mathcal{C}$ gives (TS-MPK, ID-MPK, ID-MSK) to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can adaptively issue TIK extraction queries for any time $t \in \mathsf{TSp}$. The challenger responds to each query with $k_t \leftarrow \mathsf{ID.TIK\text{-}Ext}$(TS-MPK, TS-MSK, t).

Challenge. $\mathcal{A}$ selects two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and a time interval $[t_0, t_1] \subseteq \mathsf{TSp}$, with the restriction that $t \notin [t_0, t_1]$ for all of the TIK extraction queries t in Phase 1, and $id^\star \in \mathsf{IdSp}$. $\mathcal{A}$ passes $M_0, M_1, [t_0, t_1], id^\star$ to $\mathcal{C}$. $\mathcal{C}$ chooses a random $b \leftarrow \{0, 1\}$ and computes $C^\star \leftarrow \mathsf{ID.Enc}$(TS-MPK, ID-MPK, $M_b, [t_0, t_1], id^\star$). $C^\star$ is called the *challenge ciphertext* and it is passed to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to make TIK extraction queries with the same restriction as in the Challenge phase.

Guess. The adversary outputs its guess b' for b.

We define $\mathcal{A}$'s advantage in the above game as $\mathbf{Adv}^{\text{IND}-\text{CPA}_{TA}}_{\mathcal{A},ID}(\lambda) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 5.9 (IND-CPA$_{TA}$) *We say that an ID-TSE scheme is* IND-CPA$_{TA}$ secure *if all p.p.t. adversaries have at most negligible advantage in Game$_{ID\text{-}TA}$.*

We finally give the unified notion of indistinguishability under chosen-plaintext attacks for ID-TSE next.

Definition 5.10 *We say that an ID-TSE scheme* $ID = (\mathsf{TS.Setup}, \mathsf{ID.Setup}, \mathsf{ID.TIK\text{-}Ext}, \mathsf{ID.Key\text{-}Ext}, \mathsf{ID.Enc}, \mathsf{ID.Dec})$ *is* IND-CPA secure *if it is both IND-CPA*$_{TS}$ *and IND-CPA*$_{TA}$ *secure.*

IND-CCA security for ID-TSE can be defined by giving the adversary suitably restricted access to a decryption oracle. However, we do not formalise this notion here, since we are mainly interested in using ID-TSE as a building block to obtain PK-TSE schemes.

5.3 Constructions for TSE Schemes

From the discussion on related work at the beginning of this chapter it is clear that TSE is achievable. For instance, we can build Plain TSE from attribute-based or broadcast encryption. In our work we explore alternative ways of achieving all flavours of TSE, with the aim of obtaining more efficient constructions through a dedicated approach.

5.3.1 Plain TSE

Our first step towards building TSE schemes is to focus on how to achieve Plain TSE. We have to find a way to express time so that the Time Server can extract time instant keys (TIKs) and the sender can specify a decryption time interval. Our approach will make use of a binary tree of depth d, where we denote with $\mathsf{parent}(x)$ and $\mathsf{child}(x)$ the standard notions of parent and child of a node x in a tree. The input T to $\mathsf{Plain.Setup}$, which represents the number of allowed time units, will be of the form $T = 2^d$. The root node of the tree is labeled with $\emptyset$ and the non-root nodes are labeled with binary strings of lengths between 1 and d, as illustrated for the case $d = 3$ in Figure 5.1. Hence each node is associated with a binary string $t_0 t_1 ... t_{l-1}$ of length $l \leq d$. In particular we will have that the leaves of the tree are binary strings of length d labeled from 0...0 (on the left) to 1...1 (on the right). Each leaf will represent a time instant $t = \Sigma_{i=0}^{d-1} t_i 2^{d-1-i}$ between 0 and $T - 1$.

We now define two particular sets of nodes.

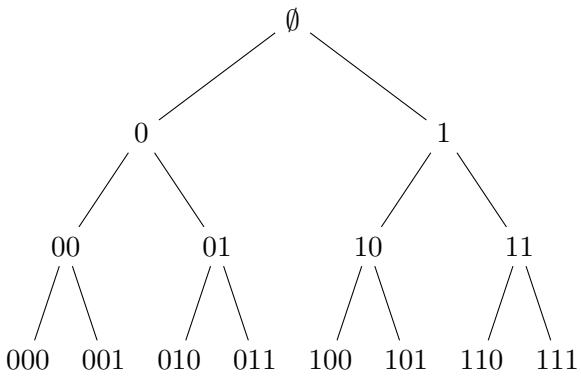

Figure 5.1: Example of binary tree of depth $d = 3$ used in our construction.

- **Path $\mathcal{P}_t$ to t.** Given a time instant $t = \Sigma_{i=0}^{d-1} t_i 2^{d-1-i}$ we construct the following path $\mathcal{P}_t$ in the tree, where the last node is the leaf corresponding to t:

$$\emptyset, t_0, t_0 t_1, ..., t_0 ... t_{d-1}.$$

- **Set $\mathcal{S}_{[t_0,t_1]}$ covering the interval $[t_0, t_1]$.** $\mathcal{S}_{[t_0,t_1]}$ is defined to be the minimal set of roots of subtrees that cover *exactly* the leaves representing time instants in $[t_0, t_1]$. We will call this the *cover set* for $[t_0, t_1]$. Such a set is unique and of size at most[2] $2d$. We can compute the labels of the nodes in $\mathcal{S}_{[t_0,t_1]}$ in a particular order by running Algorithm 1 on input $[t_0, t_1]$.

It is easy to see that $\mathcal{P}_t$ and $\mathcal{S}_{[t_0,t_1]}$ intersect in a unique node if and only if $t \in [t_0, t_1]$. This property will ensure that the correctness requirement holds for the Plain TSE scheme that we will construct. The key idea, then, is to view the nodes of the tree as identities and make use of identity-based encryption techniques to instantiate a Plain TSE scheme. Informally, the sender will encrypt under the nodes in the cover set for the decryption time interval (DTI), and the TIK for time t will be the set of private keys associated to the nodes on the path $\mathcal{P}_t$ to t.

More formally, we use an IBE scheme $I = (\mathsf{IBE.PG}, \mathsf{IBE.Setup}, \mathsf{IBE.KeyExt}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ with message space MsgSp and $\mathsf{IdSp} = \{0,1\}^{\leq d}$ to construct $P = (\mathsf{Plain.Setup}, \mathsf{Plain.TIK\text{-}Ext}, \mathsf{Plain.Enc}, \mathsf{Plain.Dec})$, a Plain TSE scheme with the same message space, in the following way.

[2] This can be proved by induction on d and it can be seen intuitively by considering the interval $[1, T-1]$, which gives rise to the largest possible cover set. Indeed, $\mathcal{S}_{[1,T-1]}$ has size $2d$.

Algorithm 1 Compute $\mathcal{S}_{[t_0,t_1]}$.

```
Let L be the binary expansion of t_0.
Let R be the binary expansion of t_1.
Let S = ∅.
while L < R do
   if L ≡ 0 mod 2 then
      L = parent(L)
   else
      S = S ∪ {L}
      L = parent(L) + 1
   end if
   if R ≡ 0 mod 2 then
      S = S ∪ {R}
      R = parent(R) − 1
   else
      R = parent(R)
   end if
end while
if L = R then
   S = S ∪ {L}
end if
return S
```

Plain.Setup$(1^\lambda, T)$: Run IBE.PG on input 1^λ to obtain $pars$ and IBE.Setup$(pars)$ to obtain TS-MPK and the master secret key TS-MSK. We set $T = 2^d$, where d is the depth of the tree used in our construction.

Plain.TIK-Ext(TS-MPK, TS-MSK, t): Construct $\mathcal{P}_t$ to obtain the list of nodes $\{\emptyset, p_1, ..., p_d\}$ on the path to t. Run IBE.KeyExt on all nodes p in $\mathcal{P}_t$ to obtain a set of private keys $\mathcal{D}_t = \{d_p : p \in \mathcal{P}_t\}$. Return $\mathcal{D}_t$ (we implicitly assume that t can be recovered from this set because $\mathcal{D}_t$ will be broadcast at the particular time t).

Plain.Enc(TS-MPK, $M, [t_0, t_1]$): Run Algorithm 1 on input $[t_0, t_1]$ to compute a list of nodes $\mathcal{S}_{[t_0,t_1]}$. For each $s \in \mathcal{S}_{[t_0,t_1]}$ run IBE.Enc(TS-MPK, M, s) to obtain a list of ciphertexts $\mathcal{CT}_{[t_0,t_1]} = \{c_s : s \in \mathcal{S}_{[t_0,t_1]}\}$. Output $C = (\mathcal{CT}_{[t_0,t_1]}, [t_0, t_1])$.

Plain.Dec(TS-MPK, $C, \mathcal{D}_t$): Here $C = (\mathcal{CT}, [t_0, t_1])$ denotes a list of ciphertexts for the scheme I together with a time interval. If $t \notin [t_0, t_1]$ return $\perp$. Otherwise run Algorithm 1 to generate an ordered list of nodes $\mathcal{S}_{[t_0,t_1]}$ and generate the set $\mathcal{P}_t$; the intersection of these sets is a unique node p. Obtain the key d_p corresponding to p from $\mathcal{D}_t$. Run IBE.Dec(TS-MPK, c_p, d_p), where $c_p \in \mathcal{CT}$ is in the same position in the list $\mathcal{CT}$ as p is in $\mathcal{S}_{[t_0,t_1]}$, to obtain either a message M or a failure symbol $\perp$. Output the result.

We provide a small example to illustrate our construction. Consider $d = 3$ as in Figure 1.

Example: Suppose we wish to decrypt a message that was encrypted using time interval $[2, 6]$. In the tree, these endpoints will correspond to nodes with labels 010 and 110, respectively. We compute $\mathcal{S}_{[2,6]} = \{110, 01, 10\}$. Suppose we obtain the TIK broadcast by TS at time 4 (corresponding to the leaf node labelled 100). This means that we obtain a list of private keys for nodes on the path $\mathcal{P}_4$ from the root to 100. In particular, we have the key corresponding to node 10, the unique intersection of $\mathcal{P}_4$ and $\mathcal{S}_{[2,6]}$. Hence, we are able to decrypt. We observe that for any time t outside of the interval $[2, 6]$, there is no intersection between $\mathcal{P}_4$ and $\mathcal{S}_{[2,6]}$.

For the above construction, the following result holds.

Theorem 5.11 *Let I be an IND-CPA secure IBE scheme. Then the Plain TSE scheme P constructed from I as above is IND-CPA secure, and is correct.*

Proof. The proof of IND-CPA security works in two steps. In the first step, we show that for any IND-CPA attacker $\mathcal{A}$ with non-negligible advantage against the Plain TSE scheme, there is a modified IND-CPA adversary $\mathcal{B}$ having non-negligible advantage against the IBE scheme. This modified adversary specifies in its challenge phase a list of ℓ identities, none of which are allowed to be queried to the ID-based private key extraction oracle at any point in the game, along with a pair of messages M_0, M_1. In return, $\mathcal{B}$ receives from its challenger the encryption of M_b under each of the identities in its list. As usual $\mathcal{B}$'s task is to find b. This reduction is straightforward, relying on the fact that the restriction $t \notin [t_0, t_1]$ imposed on the TIKs k_t that the Plain TSE adversary $\mathcal{A}$ can obtain from its TIK extraction oracle means that $\mathcal{B}$ can handle $\mathcal{A}$'s queries to its TIK extraction oracle by passing them to its ID-based private key extraction oracle.

In the second step, we use a standard hybrid argument to reduce the IND-CPA security of the IBE scheme against modified adversaries to its IND-CPA security against normal adversaries (who specify just one identity in the challenge phase). Indeed, we can define a sequence of games starting with Game_0, which is the original

game in which the challenger encrypts M_0 under all ℓ identities, and then slightly modify the game as usual up to Game_ℓ, the last game, in which the challenger encrypts M_1. It is easy to see that if an adversary can distinguish between any two successive games then this can be turned into an adversary that breaks the IND-CPA security of the underlying IBE scheme I.

The proof of correctness is straightforward, relying on the fact that $\mathcal{P}_t$ does not intersect $\mathcal{S}_{[t_0,t_1]}$ if $t \notin [t_0, t_1]$. □

In general, ciphertexts in the Plain TSE scheme P consist of up to $2d$ ciphertexts from the IBE scheme I, while private keys consist of at most d private keys from I. The public parameters of P are the same size as those of I. The cost of encryption for the scheme P is up to $2d$ times greater than its cost for the scheme I, while decryption for P costs the same as for I. This compares well with the naive solution of encrypting with a single private key to every time instant in the interval, as it allows for shorter ciphertexts.

A variety of IBE schemes can be used to instantiate the above construction, including Waters' [90] and Gentry's [50] schemes in the standard model, and the Boneh-Franklin scheme [16] and the Sakai-Kasahara scheme (as analysed in [31]) in the ROM. Each of them has various advantages and disadvantages in terms of efficiency and the sizes of public parameters and ciphertexts. For example, Waters' scheme has relatively large public parameters, compact ciphertexts, and depends for its security on the Bilinear Diffie-Hellman Problem, while Gentry's scheme has small public parameters and ciphertexts, but its security depends on a non-standard hardness assumption, the q-Truncated Decisional Augmented Bilinear Diffie-Hellman Exponent (q-TDABDHE) problem.

A potentially more efficient approach would be to use a multi-recipient, single key, ID-based Key Encapsulation Mechanism (MR-SK-IBKEM), as defined in [4], which would allow encapsulation of the same key for multiple recipients $id_1, ..., id_n$ in an efficient and secure manner. Using an approach similar to that in [87], we can combine an MR-SK-IBKEM with a (symmetric) Data Encapsulation Mechanism (DEM) to produce a multi-recipient IBE scheme in a standard way [11]; if the underlying KEM and DEM satisfy appropriate security notions, then the resulting

multi-recipient IBE scheme will be IND-CPA secure [11]. This primitive perfectly matches our requirement to be able to encrypt the same message to all nodes in a cover set simultaneously, and it is easy to see how to obtain IND-CPA secure Plain TSE from such a primitive. However, current instantiations for IND-CPA secure MR-SK-IBKEMs are only known in the random oracle model (ROM) (see for example [4]). To the best of our knowledge, it remains an open problem to find efficient instantiations that are secure in the standard model. We recall that the scheme in [87] actually solves the *dual* of our problem and can only handle intervals of the type $[t, T-1]$.

5.3.2 PK-TSE

Having presented a way to efficiently construct Plain TSE, we now wish to address the problem of obtaining PK-TSE. We first look at how to achieve IND-CPA security in the public-key setting.

5.3.2.1 IND-CPA Security

In our construction we will use a PKE scheme and a Plain TSE scheme as building blocks[3]. The idea is to *split* the message into two parts, each of which will be encrypted under a different primitive. To recover the message, a user will need both appropriate keys, as desired.

Let $\Pi = (\mathsf{PKE.PG}, \mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ be a PKE scheme with message space $\{0,1\}^l$. We will construct a PK-TSE scheme from a Plain TSE scheme $P = (\mathsf{Plain.Setup}, \mathsf{Plain.TIK\text{-}Ext}, \mathsf{Plain.Enc}, \mathsf{Plain.Dec})$ with $\mathsf{MsgSp} = \{0,1\}^l$ and Π in the following way:

$\mathsf{PK.Setup}(1^\lambda, T)$: Run $\mathsf{Plain.Setup}$ on the same input to obtain TS-MPK and the master secret key TS-MSK.

[3]Here, we will assume the schemes have the same message space consisting of bit-strings of a fixed length.

PK.TIK-Ext(TS-MPK, TS-MSK, t): Run Plain.TIK-Ext(TS-MPK, TS-MSK, t) to obtain k_t, broadcast by TS at time t.

PK.KeyGen(1^λ): Run PKE.PG(1^λ) to obtain $pars$ and PKE.KeyGen($pars$) to obtain a key-pair (pk, sk).

PK.Enc(TS-MPK, $M, [t_0, t_1], pk$): Pick a random $r \in \{0,1\}^l$ and set $M' = M \oplus r$. Then run Plain.Enc(TS-MPK, $r, [t_0, t_1]$) to obtain C_0 and PKE.Enc(M', pk) to get C_1. The ciphertext will be $C = (C_0, C_1)$.

PK.Dec(TS-MPK, C, k_t, sk): Parse C as C_0 and C_1. Run Plain.Dec(TS-MPK, C_0, k_t) which will output either a message r or a failure symbol $\perp$. Run PKE.Dec(C_1, sk) which will output either a message M' or $\perp$. If either of the decryption algorithms returns $\perp$, then output $\perp$; otherwise output $M = r \oplus M'$.

Lemma 5.12 *Let* $\Pi = (PKE.PG, PKE.KeyGen, PKE.Enc, PKE.Dec)$ *be an IND-CPA secure PKE scheme. Then the PK-TSE scheme PK constructed as above is IND-CPA$_{TS}$ secure.*

Proof. Suppose we have an IND-CPA$_{TS}$ adversary $\mathcal{A}$ against the PK-TSE scheme. We will construct an adversary $\mathcal{B}$ that will interact with $\mathcal{A}$ to break the IND-CPA security of the PKE scheme. The game proceeds as follows.

Setup. The challenger $\mathcal{C}$ runs PKE.PG(1^λ) to obtain $pars$ and PKE.KeyGen($pars$) to generate a pair (pk, sk). It gives pk to $\mathcal{B}$. $\mathcal{B}$ runs Plain.Setup($1^\lambda, T$) to generate (TS-MPK, TS-MSK) and gives (TS-MPK, TS-MSK, pk) to the adversary $\mathcal{A}$.

Challenge. $\mathcal{A}$ outputs two equal-length messages M_0 and $M_1 \in$ MsgSp and a time interval $[t_0, t_1] \subseteq \mathcal{T}$. $\mathcal{A}$ passes $M_0, M_1, [t_0, t_1]$ to $\mathcal{B}$. $\mathcal{B}$ picks a random $r \in \{0,1\}^l$ and passes $M_0 \oplus r, M_1 \oplus r$ to $\mathcal{C}$. $\mathcal{C}$ picks a random bit $b \leftarrow \{0,1\}$ and computes $C'' =$ PKE.Enc($M_b \oplus r, pk$). $\mathcal{B}$ runs Plain.Enc(TS-MPK, $r, [t_0, t_1]$) to obtain C'. Finally, $\mathcal{B}$ passes $C^\star = (C', C'')$ to $\mathcal{A}$.

Guess. The adversary outputs its guess b' for b and $\mathcal{B}$ outputs the same guess.

$\mathcal{B}$ perfectly simulates the IND-CPA$_{TS}$ game for $\mathcal{A}$. Hence $\mathcal{A}$'s advantage is as it would be in Game$_{\text{PK-TS}}$ and by construction $\mathcal{B}$ wins whenever $\mathcal{A}$ does. □

We can prove the following result in an analogous way.

Lemma 5.13 *Let P be an IND-CPA secure Plain TSE scheme. Then the PK-TSE scheme, constructed as above, is IND-CPA$_{CR}$ secure. Moreover, if P is correct, then so is the resulting PK-TSE scheme.*

Proof. Suppose we have an IND-CPA$_{CR}$ adversary $\mathcal{A}$ against the PK-TSE scheme. We will construct an adversary $\mathcal{B}$ that will interact with $\mathcal{A}$ to break the IND-CPA security of the Plain TSE scheme P. The game proceeds as follows.

Setup. The challenger $\mathcal{C}$ runs $\mathsf{Plain.Setup}(1^\lambda, T)$ to generate (TS-MPK, TS-MSK) and gives TS-MPK to $\mathcal{B}$. $\mathcal{B}$ runs $\mathsf{PKE.PG}(1^\lambda)$ to obtain $pars$ and $\mathsf{PKE.KeyGen}(pars)$ to generate a pair (pk, sk) and gives (TS-MPK, pk, sk) to the adversary $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can adaptively issue TIK extraction queries for any time $t \in \mathsf{TSp}$. When $\mathcal{B}$ receives such a query, it simply passes it on to its own challenger $\mathcal{C}$. $\mathcal{C}$ responds to each query with k_t which $\mathcal{B}$ forwards to $\mathcal{A}$ as a response.

Challenge. $\mathcal{A}$ outputs two equal-length messages M_0 and $M_1 \in \mathsf{MsgSp}$ and a time interval $[t_0, t_1] \subseteq \mathcal{T}$. $\mathcal{A}$ passes $M_0, M_1, [t_0, t_1]$ to $\mathcal{B}$. $\mathcal{B}$ picks a random $r \in \{0,1\}^l$ and passes $M_0 \oplus r, M_1 \oplus r$ to $\mathcal{C}$. $\mathcal{C}$ picks a random bit $b \leftarrow \{0,1\}$ and computes $C' = \mathsf{Plain.Enc}(\text{TS-MPK}, M_b \oplus r, [t_0, t_1])$. $\mathcal{B}$ runs $\mathsf{PKE.Enc}(r, pk)$ to obtain C''. Finally, $\mathcal{B}$ passes $C^\star = (C', C'')$ to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to make TIK extraction queries and $\mathcal{B}$ handles them as before.

Guess. The adversary outputs its guess b' for b and $\mathcal{B}$ outputs the same guess.

$\mathcal{B}$ perfectly simulates the IND-CPA$_{CR}$ game for $\mathcal{A}$. Hence $\mathcal{A}$'s advantage is as it would be in the real game and by construction $\mathcal{B}$ wins whenever $\mathcal{A}$ does. We observe that the correctness of PK follows directly from the correctness of P. □

Hence, the following theorem holds.

Theorem 5.14 *Let P be an IND-CPA secure Plain TSE scheme and Π be an IND-CPA secure PKE scheme. Then the PK-TSE scheme, constructed as above, is IND-CPA secure. Moreover, if P is correct, then so is the resulting PK-TSE scheme.*

5.3.2.2 IND-CPA secure ID-TSE

To achieve IND-CPA security in the ID-TSE setting we can adopt an approach similar to the one used above to build a PK-TSE scheme, where instead of a PKE scheme we employ an IBE scheme $I = (\mathsf{IBE.PG}, \mathsf{IBE.Setup}, \mathsf{IBE.KeyExt}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ in the obvious manner. In this setting we obtain an analogous result:

Theorem 5.15 *Let P be an IND-CPA secure Plain TSE scheme and I be an IND-CPA secure IBE scheme. Then the ID-TSE scheme, constructed analogously to the construction of the PK-TSE scheme above, is IND-CPA secure. Moreover, if P is correct, then so is the resulting ID-TSE scheme.*

In particular, we observe that if I is a *selective-id* IND-CPA secure IBE scheme, then it can be shown that the resulting ID-TSE scheme is also selectively secure.

5.3.2.3 IND-CCA Security

We will now address the problem of building IND-CCA$_{TS}$ secure PK-TSE schemes, using an approach similar to that of [25].

Consider a selective-id IND-CPA$_{TS}$ secure ID-TSE scheme $ID = (\mathsf{TS\text{-}Setup}, \mathsf{ID\text{-}Setup}, \mathsf{ID.TIK\text{-}Ext}, \mathsf{ID.Key\text{-}Ext}, \mathsf{ID.Enc}, \mathsf{ID.Dec})$, with $\mathsf{MsgSp} = \{0,1\}^l$ and $\mathsf{IdSp} = \{0,1\}^n$. We will construct from ID an IND-CCA$_{TS}$ secure PK-TSE scheme $PK = (\mathsf{PK.Setup}, \mathsf{PK.TIK\text{-}Ext}, \mathsf{PK.KeyGen}, \mathsf{PK.Enc}, \mathsf{PK.Dec})$. In the construction, we will also use a signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$, whose generation algorithm Gen outputs verification keys of length n. We construct the algorithms of PK as follows.

$\mathsf{PK.Setup}(1^\lambda, T)$: Run $\mathsf{TS\text{-}Setup}(1^\lambda, T)$ to get TS-MPK, TS-MSK.

PK.TIK-Ext(TS-MPK, TS-MSK, t): Run ID.TIK-Ext(TS-MPK, TS-MSK, t) to obtain TIK k_t.

PK.KeyGen(1^λ): Run ID-Setup(1^λ) to get (ID-MPK, ID-MSK), a key-pair.

PK.Enc(TS-MPK, $M, [t_0, t_1]$, ID-MPK): Run Gen(1^λ) and obtain $(sigk, vk)$. Compute $c \leftarrow$ ID.Enc(TS-MPK, ID-MPK, $M, [t_0, t_1], vk$). Produce $\sigma \leftarrow$Sign$(sigk, c)$. The final ciphertext will be $C = (vk, c, \sigma)$.

PK.Dec(TS-MPK, C, k_t, ID-MSK): Parse C as (vk, c, σ). Check if Ver$(vk, c, \sigma) = 1$. If not, output $\perp$. Otherwise, run ID.Key-Ext(ID-MPK, ID-MSK, vk) to obtain sk_{vk} and decrypt c by running ID.Dec with inputs k_t, sk_{vk}.

The following result holds.

Theorem 5.16 *Let ID be a correct, selective-id IND-CPA$_{TS}$ secure ID-TSE scheme and Σ a strongly unforgeable one-time signature scheme. Then PK, as constructed above, is an IND-CCA$_{TS}$ secure PK-TSE scheme.*

Proof. Our proof follows closely the proof of [25, Theorem 1], with suitable modifications. Given an IND-CCA$_{TS}$ adversary $\mathcal{A}$ against PK we construct an adversary $\mathcal{B}$ that will interact with $\mathcal{A}$ to break the selective-id IND-CPA$_{TS}$ security of ID. Before presenting the game, we make the following important definitions.

First, we denote by $\Pr_{\mathcal{A},S}[\mathsf{Event}]$ the probability that Event occurs when an adversary $\mathcal{A}$ interacts with a scheme S in a specified security game. By $\neg$Event we denote the complement of Event. In particular, we denote by Succ the event that $b' = b$ in the games played in this proof.

We say that a ciphertext $C = (vk, c, \sigma)$ is valid if Ver$(vk, c, \sigma) = 1$. Let $C^\star = (vk^\star, c^\star, \sigma^\star)$ denote the challenge ciphertext received by $\mathcal{A}$ during the game, and let F denote the event that in this game $\mathcal{A}$ submits a valid ciphertext $C = (vk^\star, c, \sigma)$ to its decryption oracle. It can be shown that $\mathcal{A}$ can be used to break the underlying one-time signature scheme Σ with probability exactly $\Pr_{\mathcal{A},PK}[\mathsf{F}]$. Since Σ is a strongly unforgeable one-time signature (as per Definition 2.15), we can assume that $\Pr_{\mathcal{A},PK}[\mathsf{F}]$ is negligible in the security parameter λ.

We now describe the simulation.

Initialize. $\mathcal{B}$ runs $\mathsf{Gen}(1^\lambda)$ to get $(sigk^\star, vk^\star)$. The string $vk^\star$ will be the target identity used by $\mathcal{B}$.

Setup. $\mathcal{C}$ runs TS-Setup, ID-Setup to get (TS-MPK, TS-MSK, ID-MPK, ID-MSK). It gives $\mathcal{B}$ (TS-MPK, TS-MSK, ID-MPK). $\mathcal{B}$ passes the same information to $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can issue queries of the form (C, t) to its decryption oracle, where C is of the form (vk, c, σ). $\mathcal{B}$ responds as follows:

- if $\mathsf{Ver}(vk, c, \sigma) \neq 1$ then $\mathcal{B}$ returns $\perp$;
- if $\mathsf{Ver}(vk, c, \sigma) = 1$ and $vk = vk^\star$ then $\mathcal{B}$ aborts and outputs a random bit (event F just occurred);
- if $\mathsf{Ver}(vk, c, \sigma) = 1$ and $vk \neq vk^\star$ then $\mathcal{B}$ makes a query vk to its secret key extraction oracle to obtain sk_{vk}. $\mathcal{B}$ can compute the TIK k_t as it obtained TS-MSK from its challenger. $\mathcal{B}$ then computes $\mathsf{ID.Dec}$(TS-MPK, ID-MPK, k_t, sk_{vk}) and obtains either a message M or failure symbol $\perp$, which is passed to $\mathcal{A}$.

Challenge. $\mathcal{A}$ outputs M_0, M_1 and $[t_0, t_1]$ and passes the tuple $(M_0, M_1, [t_0, t_1])$ to $\mathcal{B}$, who then sends $(M_0, M_1, [t_0, t_1], vk^\star)$ to $\mathcal{C}$ as its challenge query. $\mathcal{C}$ picks a random bit b and computes $c^\star \leftarrow \mathsf{ID.Enc}$(TS-MPK, ID-MPK, $M_b, [t_0, t_1], vk^\star$). $\mathcal{C}$ gives $c^\star$ to $\mathcal{B}$, who computes $\sigma^\star = \mathsf{Sign}(sigk^\star, c^\star)$ and returns $C^\star = (vk^\star, c^\star, \sigma^\star)$ to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ can continue issuing queries of the form (C, t), where $C = (vk, c, \sigma)$, with the restriction that $(C, t) \neq (C^\star, t')$, where $t' \in [t_0, t_1]$. If $\mathsf{Ver}(vk, c, \sigma) \neq 1$ then $\mathcal{B}$ returns $\perp$. Otherwise, $\mathcal{B}$ will respond as described in the following cases:

- Case 1: $C \neq C^\star$.
  - if $c \neq c^\star$ and $vk = vk^\star$ then $\mathcal{B}$ aborts and outputs a random bit (event F just occurred);
  - if $c = c^\star$ and $vk = vk^\star$ then $\mathcal{B}$ aborts and outputs a random bit (event F just occurred);

- if $c = c^\star$ and $vk \neq vk^\star$ then $\mathcal{B}$ passes the oracle query vk to its challenger to obtain sk_{vk}. $\mathcal{B}$ then computes the TIK k_t using TS-MSK. $\mathcal{B}$ then computes $\mathsf{ID.Dec}(\text{TS-MPK}, \text{ID-MPK}, k_t, sk_{vk})$ and obtains either a message M or failure symbol $\perp$, which it passes to $\mathcal{A}$.

- Case 2: $C = C^\star$ and $t' \notin [t_0, t_1]$.

  In this case, the correctness of the scheme ID implies that the decryption algorithm $\mathsf{ID.Dec}$ applied to $c^\star$ with key k_t and identity $vk^\star$ should output $\perp$, so $\mathcal{B}$ returns $\perp$.

Guess. $\mathcal{A}$ outputs b' as its guess for b. $\mathcal{B}$ outputs the same bit.

We note that $\mathcal{B}$ provides a perfect simulation for $\mathcal{A}$ as well as a legal strategy for attacking scheme ID, provided that $\mathcal{B}$ is not forced to abort (a situation that occurs only when the event F occurs). In particular $\mathcal{B}$ never queries its challenger for the secret key corresponding to the target identity $vk^\star$. We hence have that $\mathcal{B}$ wins if $\mathcal{A}$ does, and this can only happen when the event F does not occur. In that case, $\mathcal{B}$ is forced to abort and outputs a random bit. We therefore have

$$|\Pr_{\mathcal{B},ID}[\mathsf{Succ}] - \tfrac{1}{2}| = |\Pr_{\mathcal{A},PK}[\mathsf{Succ} \wedge \neg\mathsf{F}] + \tfrac{1}{2}\Pr_{\mathcal{A},PK}[\mathsf{F}] - \tfrac{1}{2}| \ .$$

We observe that the RHS of above the equation is negligible since both $\Pr_{\mathcal{A},PK}[\mathsf{F}]$ and $\left|\Pr_{\mathcal{B},ID}[\mathsf{Succ}] - \frac{1}{2}\right|$ are negligible, assuming the security of schemes ID and Σ. Finally we have:

$$\begin{aligned}
&|\Pr_{\mathcal{A},PK}[\mathsf{Succ}] - \tfrac{1}{2}| \\
&= |\Pr_{\mathcal{A},PK}[\mathsf{Succ} \wedge \mathsf{F}] + \Pr_{\mathcal{A},PK}[\mathsf{Succ} \wedge \neg\mathsf{F}] - \tfrac{1}{2}\Pr_{\mathcal{A},PK}[\mathsf{F}] + \tfrac{1}{2}\Pr_{\mathcal{A},PK}[\mathsf{F}] - \tfrac{1}{2}| \\
&\leq |\Pr_{\mathcal{A},PK}[\mathsf{Succ} \wedge \mathsf{F}] - \tfrac{1}{2}\Pr_{\mathcal{A},PK}[\mathsf{F}]| + |\Pr_{\mathcal{A},PK}[\mathsf{Succ} \wedge \neg\mathsf{F}] + \tfrac{1}{2}\Pr_{\mathcal{A},PK}[\mathsf{F}] - \tfrac{1}{2}| \\
&\leq \tfrac{1}{2}\Pr_{\mathcal{A},PK}[\mathsf{F}] + |\Pr_{\mathcal{A},PK}[\mathsf{Succ} \wedge \neg\mathsf{F}] + \tfrac{1}{2}\Pr_{\mathcal{A},PK}[\mathsf{F}] - \tfrac{1}{2}| \ ,
\end{aligned}$$

where on the RHS we have two negligible quantities. We have hence proved that $\mathcal{A}$'s advantage in winning the IND-CCA$_{TS}$ game is negligible. □

We have therefore provided a way to generically and efficiently achieve IND-CCA secure PK-TSE. We could also use a variant of the more complex Boneh-Katz transform [19] to again construct PK-TSE schemes that are IND-CCA secure in the standard model. The resulting schemes would generally have improved efficiency.

5.4 Extensions

There are several other possible extensions to this area of research. These include alternative approaches to the TSE problem as well as the development of additional capabilities provided to the system. We give an overview of the ones we have considered and some that have arisen after our initial work.

5.4.1 Plain TSE from BE

As mentioned in the introduction of this chapter, Plain TSE can be seen as a special case of broadcast encryption (BE), where time instants are the *users* in the BE setting and the DTI is the *target set*. We formalize this idea next.

Let $B = (\mathsf{BE.PG}, \mathsf{BE.Setup}, \mathsf{BE.KeyGen}, \mathsf{BE.Enc}, \mathsf{BE.Dec})$ be a BE scheme with message space MsgSp, and let $U = \{0, ..., T-1\}$ be the universe of users in the system. We will construct $P = (\mathsf{Plain.Setup}, \mathsf{Plain.TIK\text{-}Ext}, \mathsf{Plain.Enc}, \mathsf{Plain.Dec})$, a Plain TSE scheme with the same message space, using B in the following way.

$\mathsf{Plain.Setup}(1^\lambda, T)$: Run $\mathsf{BE.PG}(1^\lambda, T)$ to obtain $pars$ and run $\mathsf{BE.Setup}(pars)$ to obtain BE-MPK, BE-MSK. Set TS-MPK = BE-MPK and TS-MSK = BE-MSK.

$\mathsf{Plain.TIK\text{-}Ext}$(TS-MPK, TS-MSK, t): Run $\mathsf{BE.Key\text{-}Gen}$ on the same inputs to obtain the TIK k_t for t.

$\mathsf{Plain.Enc}$(TS-MPK, M, $[t_0, t_1]$): Run $\mathsf{BE.Enc}$(TS-MPK, M, S), where S is the set $\{t \ : \ t \in [t_0, t_1]\}$, to obtain a ciphertext C.

$\mathsf{Plain.Dec}$(TS-MPK, C, k_t). Run $\mathsf{BE.Dec}$(TS-MPK, C, k_t) to obtain either a message M or a failure symbol $\perp$.

For the above construction, the following result holds.

Theorem 5.17 *Let B be a fully collusion resistant, adaptively IND-CPA secure BE scheme. Then the Plain TSE scheme P constructed from B as above is IND-CPA secure.*

Proof. Suppose we have an adversary $\mathcal{A}$ against the Plain TSE scheme P. We will construct an adversary $\mathcal{B}$ that will interact with $\mathcal{A}$ to break the IND-CPA security of the BE scheme B used in the construction. The game proceeds as follows.

Setup. The challenger $\mathcal{C}$ runs $\mathsf{BE.PG}(\lambda, T)$ to obtain $pars$ and $\mathsf{BE.Setup}$ to obtain a pair (BE-MPK, BE-MSK). It gives BE-MPK to $\mathcal{B}$, who passes it on to $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can adaptively issue TIK extraction queries for any time $t \in T$. Such a query is passed on to $\mathcal{B}$, who gives it to $\mathcal{C}$ as secret key query for user t. $\mathcal{C}$ will respond to each query with the private key sk_t, which is passed to $\mathcal{B}$, who then forwards it to $\mathcal{A}$ as the TIK for time t.

Challenge. $\mathcal{A}$ selects two equal-length messages M_0, $M_1 \in \mathsf{MsgSp}$ and a challenge interval $[t_0, t_1] \subseteq T$, such that $t \notin [t_0, t_1]$ for any of the queries t in made in Phase 1. $\mathcal{A}$ passes $M_0, M_1, [t_0, t_1]$ to $\mathcal{B}$, who passes them to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit b and computes $C^\star = \mathsf{BE.Enc}(\text{BE-MPK}, M_b, [t_0, t_1])$. $C^\star$ is passed to $\mathcal{B}$, who in turn passes it to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to issue TIK extraction queries with the same restriction as in the Challenge phase. These queries are dealt with by $\mathcal{B}$ as in Phase 1.

Guess. The adversary $\mathcal{A}$ outputs its guess b' for b and $\mathcal{B}$ outputs the same guess.

$\mathcal{B}$ perfectly simulates the IND-CPA game for $\mathcal{A}$. Hence $\mathcal{A}$'s advantage is as it would be in the real security game for the BE scheme B and by construction $\mathcal{B}$ wins whenever $\mathcal{A}$ does. □

While the above result provides an alternative way of achieving secure Plain TSE, namely from BE, this approach has some limitations.

First of all, to meet our TSE security requirement, we need the BE scheme to be *fully* collusion resistant. This condition immediately rules out many of the existing schemes. Secondly, satisfying the TSE correctness property, as we have defined it in Section 5.2, requires the underlying BE scheme to have appropriate robustness guarantees, which may incur in additional computational and communication costs. Furthermore, if adaptive security is our aim, then the currently most efficient adaptively secure scheme is that of Gentry and Waters in [51]. As we know from Chapter

4, this scheme requires the specification of the target set (in our case, the DTI) as an input to the decryption algorithm, *inherently* preventing the resulting Plain TSE scheme from having the DTI confidentiality property. In terms of efficiency, the BE scheme in [51] has constant size secret keys, but public parameters and ciphertexts of size $O(\sqrt{T})$. The resulting Plain TSE scheme inherits these sizes. If we are willing to give up on adaptive security, then other schemes may be considered. In particular, the BE scheme of Boneh et al. in [17], proved only statically secure, results in a Plain TSE scheme with again constant size secret keys, public parameters and ciphertexts of size $O(\sqrt{T})$. In [51] Gentry and Waters introduce the new notion of semi-static security and provide a BE scheme achieving this notion which can be used to construct a Plain TSE scheme with constant size secret keys and ciphertexts but public parameters whose size is linear in T.

On the other hand, our tree-based solution results in schemes which are fully secure, with constant size public parameters, and secret keys and ciphertexts of size $O(\log T)$. This brief comparison illustrates the value of a dedicated approach when realising Plain TSE.

5.4.2 Decryption time interval confidentiality

We could consider TSE schemes (in all three settings) that have the property of hiding the decryption time interval of ciphertexts from adversaries. Our current constructions do not offer this. We call such a property DTIC (decryption time interval confidentiality). We can model the DTIC-IND-CPA/CCA security of a TSE scheme by allowing the adversary to select two messages and *two* time intervals in the challenge phase and requiring him to guess which message was encrypted under which interval. We distinguish between the plain setting, where the decryption time interval is hidden from all users, making successful decryption a kind of *proof of work* (since every user will need to attempt decryption under possibly many keys), and the public-key and identity-based settings, where the decryption time interval is hidden from everyone except the intended recipient.

In light of the results in Section 5.4.1 and our work on anonymous broadcast encryption (ANOBE) in Chapter 4, it seems very natural to consider using an ANOBE

scheme to build TSE having decryption time interval confidentiality. While this approach would guarantee the desired security properties, currently known instantiations of ANOBE do not provide a significant efficiency improvement over the naive solution to the TSE problem. Indeed, the spirit of state-of-the-art ANOBE schemes [62], even if highly optimized, is essentially that of encrypting the same message multiple times, and therefore the resulting TSE ciphertext would have size linear in that of the time interval.

Achieving better performing TSE schemes with DTIC, perhaps exploring possible extensions of the key-privacy/recipient-anonymity properties in the public-key and identity-based settings, represents an interesting direction for future work.

5.4.3 Time and parameters

The model for TSE we put forth in this chapter deals with intervals of the form $[t_0, t_1]$, where $0 \leq t_0 \leq t_1 < T$. In certain scenarios it could be useful to be able to address more elaborate decryption time sets (we may want to grant access to information only between 6 and 7 pm on Thursdays, for instance). This is clearly a more complex problem and our current tree-based approach does not generically guarantee an efficient solution to this: indeed, in the case of the time set being every other time unit, our approach would degenerate to the naive solution of encrypting to each time unit. In the plain setting, BE could achieve the desired functionality since, by definition, in BE we can address an *arbitrary* subset of a universe. An interesting extension is to consider potential approaches in the public-key and identity-based setting as well.

In terms of practical considerations, relevant issues concerning time are for example how to allow for a dynamic selection of time-granularity (such a consideration was made in [87] for TIBE), or how to securely extend the maximum time value T without resetting the parameters of the time server. Furthermore, in view of designing a real-world system, one could envisage schemes supporting multiple time servers, as well as revocation and delegation mechanisms. We observe that, in our constructions, we used the same security parameter across the components of the system, i.e. the TSE time server, the key generation algorithms and the IBE trusted

authority: in reality, these may not share the same security concerns, and therefore require different security levels.

5.4.4 Follow-up work

In [71], the published version of this chapter, we left as an interesting direction for future work that of constructing TSE schemes allowing the capability of *opening* the message outside of the decryption time interval, a useful feature supporting *break the glass* policies [3]. This extension has already been considered in the setting of TRE [56, 41]. In [64], the authors address this problem in the public-key setting and provide a generic construction for a PK-TSE scheme with pre-open capability which is IND-CCA$_{TS}$ and IND-CPA$_{CR}$ secure.

In this thesis, we have constructed PK-TSE/ID-TSE schemes in the IND-CPA setting by combining Plain TSE schemes and PKE/IBE schemes. We have then used the CHK technique to obtain CCA security for PK-TSE. It might be worth investigating an alternative approach in which one first obtains an IND-CCA secure Plain TSE scheme, and then applies a Dodis-Katz style construction [44] to combine this with IND-CCA secure PKE/IBE schemes. This may lead to efficiency gains as compared to our CHK-based constructions. It would also be useful to solve the open problem of constructing MR-SK-IBKEMs that are provably secure in the standard model, in order to improve the efficiency of our Plain TSE constructions.

Our focus has been on achieving IND-CCA security of PK-TSE in the standard model. This leaves the problem of constructing IND-CCA secure ID-TSE schemes, in either the standard model or the ROM. The former, we believe, should be achievable by introducing hierarchical ID-TSE notions and further extending the CHK-style transformation to this setting. The latter can be achieved by developing Fujisaki-Okamoto style transformations for the TSE setting.

Thinking more broadly, one can envisage the development of the wider concept of time-specific cryptography. This could include, for example, time-specific signatures (where signatures can only be created within certain time intervals). We believe there is much interesting work still to be done in this area.

Bibliography

[1] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. *J. Cryptology*, 21(3):350–391, 2008.

[2] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust encryption. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *Lecture Notes in Computer Science*, pages 480–497. Springer, 2010.

[3] Claudio Agostino Ardagna, Sabrina De Capitani di Vimercati, Tyrone Grandison, Sushil Jajodia, and Pierangela Samarati. Regulating exceptions in healthcare using policy spaces. In Vijay Atluri, editor, *DBSec*, volume 5094 of *Lecture Notes in Computer Science*, pages 254–267. Springer, 2008.

[4] Manuel Barbosa and Pooya Farshim. Efficient identity-based key encapsulation to multiple parties. In Smart [86], pages 428–441.

[5] Manuel Barbosa and Pooya Farshim. Randomness reuse: Extensions and improvements. In Steven D. Galbraith, editor, *IMA Int. Conf. 2007*, volume 4887 of *Lecture Notes in Computer Science*, pages 257–276. Springer, 2007.

[6] Adam Barth, Dan Boneh, and Brent Waters. Privacy in encrypted content distribution using private broadcast encryption. In Giovanni Di Crescenzo and Aviel D. Rubin, editors, *Financial Cryptography 2006*, volume 4107 of *Lecture Notes in Computer Science*, pages 52–64. Springer, 2006.

[7] Amos Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.

[8] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer, 2001.

[9] Mihir Bellare, Alexandra Boldyreva, Kaoru Kurosawa, and Jessica Staddon. Multirecipient encryption schemes: How to save on bandwidth and computation without sacrificing security. *IEEE Trans. on Information Theory 2007*, 53(11):3927–3943, 2007.

[10] Mihir Bellare, Alexandra Boldyreva, and Jessica Staddon. Randomness re-use in multi-recipient encryption schemes. In *PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 85–99. Springer, 2003.

[11] Kamel Bentahar, Pooya Farshim, John Malone-Lee, and Nigel P. Smart. Generic constructions of identity-based and certificateless KEMs. *J. Cryptology*, 21(2):178–199, 2008.

[12] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In Pfitzmann and McDaniel [75], pages 321–334.

[13] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 2004.

[14] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Cramer [35], pages 440–456.

[15] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 506–522. Springer, 2004.

[16] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil Pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.

[17] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor,

CRYPTO 2005, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275. Springer, 2005.

[18] Dan Boneh and Michael Hamburg. Generalized identity based and broadcast encryption schemes. In Josef Pieprzyk, editor, *ASIACRYPT 2008*, volume 5350 of *Lecture Notes in Computer Science*, pages 455–470. Springer, 2008.

[19] Dan Boneh and Jonathan Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In Alfred Menezes, editor, *CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 87–103. Springer, 2005.

[20] Dan Boneh and Moni Naor. Timed commitments. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 236–254. Springer, 2000.

[21] Dan Boneh, Amit Sahai, and Brent Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In Vaudenay [89], pages 573–592.

[22] Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 290–307. Springer, 2006.

[23] Jan Camenisch, Markulf Kohlweiss, Alfredo Rial, and Caroline Sheedy. Blind and anonymous identity-based encryption and authorised private searches on public key encrypted data. In Stanislaw Jarecki and Gene Tsudik, editors, *Public Key Cryptography 2009*, volume 5443 of *Lecture Notes in Computer Science*, pages 196–214. Springer, 2009.

[24] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 255–271. Springer, 2003.

[25] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222. Springer, 2004.

[26] Julien Cathalo, Benoît Libert, and Jean-Jacques Quisquater. Efficient and non-interactive timed-release encryption. In Sihan Qing, Wenbo Mao, Javier Lopez, and Guilin Wang, editors, *ICICS 2005*, volume 3783 of *Lecture Notes in Computer Science*, pages 291–303. Springer, 2005.

[27] Aldar C.-F. Chan and Ian F. Blake. Scalable, server-passive, user-anonymous timed release cryptography. In *ICDCS*, pages 504–513. IEEE Computer Society, 2005.

[28] David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO 1982*, pages 199–203, 1982.

[29] David Chaum. Security without identification: Transaction systems to make Big Brother obsolete. *Commun. ACM 1985*, 28(10):1030–1044, 1985.

[30] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *EUROCRYPT 1991*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer, 1991.

[31] Liqun Chen and Zhaohui Cheng. Security proof of Sakai-Kasahara's identity-based encryption scheme. In Smart [86], pages 442–459.

[32] Ling Cheung and Calvin C. Newport. Provably secure ciphertext policy ABE. In Ning et al. [68], pages 456–465.

[33] Sherman S. M. Chow, Volker Roth, and Eleanor G. Rieffel. General certificateless encryption and timed-release encryption. In Rafail Ostrovsky, Roberto De Prisco, and Ivan Visconti, editors, *SCN 2008*, volume 5229 of *Lecture Notes in Computer Science*, pages 126–143. Springer, 2008.

[34] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In Bahram Honary, editor, *IMA Int. Conf. 2001*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363. Springer, 2001.

[35] Ronald Cramer, editor. *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*. Springer, 2005.

[36] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In Hugo Krawczyk, editor, *CRYPTO 1998*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer, 1998.

[37] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *IACR Cryptology ePrint Archive*, 2001:108, 2001.

[38] Jason Crampton. Trade-offs in cryptographic implementations of temporal access control. In Audun Jøsang, Torleiv Maseng, and Svein J. Knapskog, editors, *NordSec 2009*, volume 5838 of *Lecture Notes in Computer Science*, pages 72–87. Springer, 2009.

[39] Cécile Delerablée. Identity-based broadcast encryption with constant size ciphertexts and private keys. In Kaoru Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 200–215. Springer, 2007.

[40] Cécile Delerablée, Pascal Paillier, and David Pointcheval. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In Tsuyoshi Takagi, Tatsuaki Okamoto, Eiji Okamoto, and Takeshi Okamoto, editors, *Pairing 2007*, volume 4575 of *Lecture Notes in Computer Science*, pages 39–59. Springer, 2007.

[41] Alexander W. Dent and Qiang Tang. Revisiting the security model for timed-release encryption with pre-open capability. In Juan A. Garay, Arjen K. Lenstra, Masahiro Mambo, and René Peralta, editors, *ISC 2007*, volume 4779 of *Lecture Notes in Computer Science*, pages 158–174. Springer, 2007.

[42] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[43] Yevgeniy Dodis and Nelly Fazio. Public key broadcast encryption for stateless receivers. In Joan Feigenbaum, editor, *Digital Rights Management Workshop 2002*, volume 2696 of *Lecture Notes in Computer Science*, pages 61–80. Springer, 2002.

[44] Yevgeniy Dodis and Jonathan Katz. Chosen-ciphertext security of multiple encryption. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 188–209. Springer, 2005.

[45] Pooya Farshim, Benot Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. Robust encryption, revisited. Preprint, 2012.

[46] Nelly Fazio and Irippuge Milinda Perera. Outsider-anonymous broadcast encryption with sublinear ciphertexts. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 225–242. Springer, 2012.

[47] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *CRYPTO 1993*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491. Springer, 1993.

[48] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory 1985*, 31(4):469–472, 1985.

[49] Roxana Geambasu, Tadayoshi Kohno, Amit A. Levy, and Henry M. Levy. Vanish: Increasing data privacy with self-destructing data. In *USENIX Security Symposium 2009*, pages 299–316. USENIX Association, 2009.

[50] Craig Gentry. Practical identity-based encryption without random oracles. In Vaudenay [89], pages 445–464.

[51] Craig Gentry and Brent Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 171–188. Springer, 2009.

[52] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[53] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security 2006*, pages 89–98. ACM, 2006.

[54] Javier Herranz, Fabien Laguillaumie, and Carla Ràfols. Constant size ciphertexts in threshold attribute-based encryption. In Phong Q. Nguyen and David Pointcheval, editors, *Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 19–34. Springer, 2010.

[55] Dennis Hofheinz and Eike Kiltz. Secure hybrid encryption from weakened key encapsulation. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 553–571. Springer, 2007.

[56] Yong Ho Hwang, Dae Hyun Yum, and Pil Joong Lee. Timed-release encryption with pre-open capability and its application to certified e-mail system. In Jianying Zhou, Javier Lopez, Robert H. Deng, and Feng Bao, editors, *ISC 2005*, volume 3650 of *Lecture Notes in Computer Science*, pages 344–358. Springer, 2005.

[57] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC Press, 2007.

[58] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, 2008.

[59] Kaoru Kurosawa. Multi-recipient public-key encryption with shortened ciphertext. In *Public-Key Cryptography 2002*, volume 2274 of *Lecture Notes in Computer Science*, pages 48–63. Springer, 2002.

[60] Kaoru Kurosawa and Yvo Desmedt. A new paradigm of hybrid encryption scheme. In *CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 426–442. Springer, 2004.

[61] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 62–91. Springer, 2010.

[62] Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. Anonymous broadcast encryption: Adaptive security and efficient constructions in the standard model. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors,

Public Key Cryptography, volume 7293 of *Lecture Notes in Computer Science*, pages 206–224. Springer, 2012.

[63] Wenbo Mao. Timed-release cryptography. In Serge Vaudenay and Amr M. Youssef, editors, *Selected Areas in Cryptography 2001*, volume 2259 of *Lecture Notes in Computer Science*, pages 342–358. Springer, 2001.

[64] Takahiro Matsuda, Yasumasa Nakai, and Kanta Matsuura. Efficient generic constructions of timed-release encryption with pre-open capability. In Marc Joye, Atsuko Miyaji, and Akira Otsuka, editors, *Pairing 2010*, volume 6487 of *Lecture Notes in Computer Science*, pages 225–245. Springer, 2010.

[65] Timothy C. May. Time-release crypto, 1993. Manuscript.

[66] Payman Mohassel. A closer look at anonymity and robustness in encryption schemes. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 501–518. Springer, 2010.

[67] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. *Electronic Colloquium on Computational Complexity (ECCC) 2002*, (043), 2002.

[68] Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors. *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007*. ACM, 2007.

[69] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 191–208. Springer, 2010.

[70] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In Ning et al. [68], pages 195–203.

[71] Kenneth G. Paterson and Elizabeth A. Quaglia. Time-specific encryption. In Juan A. Garay and Roberto De Prisco, editors, *SCN 2010*, volume 6280 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2010.

[72] Kenneth G. Paterson and Sriramkrishnan Srinivasan. Security and anonymity of identity-based encryption with multiple trusted authorities. In Steven D.

Galbraith and Kenneth G. Paterson, editors, *Pairing 2008*, volume 5209 of *Lecture Notes in Computer Science*, pages 354–375. Springer, 2008.

[73] Kenneth G. Paterson and Sriramkrishnan Srinivasan. Building key-private public-key encryption schemes. In Colin Boyd and Juan Manuel González Nieto, editors, *ACISP 2009*, volume 5594 of *Lecture Notes in Computer Science*, pages 276–292. Springer, 2009.

[74] Radia Perlman. The ephemerizer: Making data disappear. *Journal of Information System Security*, Vol. 1(1), 2005.

[75] Birgit Pfitzmann and Patrick McDaniel, editors. *IEEE Symposium on Security and Privacy (S&P 2007), 20-23 May 2007, Oakland, California, USA*. IEEE Computer Society, 2007.

[76] Duong Hieu Phan, David Pointcheval, and Mario Strefler. Security notions for broadcast encryption. In Javier Lopez and Gene Tsudik, editors, *ACNS 2011*, volume 6715 of *Lecture Notes in Computer Science*, pages 377–394, 2011.

[77] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In Joan Feigenbaum, editor, *CRYPTO*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer, 1991.

[78] Ron Rivest, Adi Shamir, and David Wagner. Time-lock puzzles and timed-release crypto. MIT LCS Tech. Report MIT/LCS/TR-684,1996.

[79] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In Cramer [35], pages 457–473.

[80] Ryuichi Sakai, K Ohgishi, and Masao Kasahara. Cryptosystems based on pairings. *The 2000 Symposium on Cryptography an Information Security, Okinawa, Japan*, pages 26–28, 2000. (In Japanese).

[81] Kazue Sako. An auction protocol which hides bids of losers. In Hideki Imai and Yuliang Zheng, editors, *Public-Key Cryptography 2000*, volume 1751 of *Lecture Notes in Computer Science*, pages 422–432. Springer, 2000.

[82] Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO 1984*, pages 47–53, 1984.

[83] Claude Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, Vol. 28(4):656–715, 1949.

[84] Elaine Shi, John Bethencourt, Hubert T.-H. Chan, Dawn Xiaodong Song, and Adrian Perrig. Multi-dimensional range query over encrypted data. In Pfitzmann and McDaniel [75], pages 350–364.

[85] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. *IACR Cryptology ePrint Archive*, 2004:332, 2004.

[86] Nigel P. Smart, editor. *Cryptography and Coding, 10th IMA International Conference, Cirencester, UK, December 19-21, 2005, Proceedings*, volume 3796 of *Lecture Notes in Computer Science*. Springer, 2005.

[87] Mudhakar Srivatsa, Shane Balfe, Kenneth G. Paterson, and Pankaj Rohatgi. Trust management for secure information flows. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM Conference on Computer and Communications Security 2008*, pages 175–188. ACM, 2008.

[88] Jacques Stern. *La Science du secret*. Odile Jacob edition, 1998.

[89] Serge Vaudenay, editor. *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*. Springer, 2006.

[90] Brent Waters. Efficient identity-based encryption without random oracles. In Cramer [35], pages 114–127.