Université Catholique de Louvain
Faculté des Sciences Appliquées
Laboratoire de Microélectronique

New Secure Applications of Bilinear Maps in Cryptography

Benoît Libert

Thèse soutenue en vue de l'obtention du grade de
Docteur en Sciences Appliquées

Composition du jury:

Pr. Jean-Jacques Quisquater (UCL - DICE) – Promoteur
Pr. Paulo S.L.M. Barreto (Univ. de São Paulo - Brasil)
Pr. Philippe Delsarte (UCL - INFO)
Dr. Andreas Enge (LIX – France)
Dr. Marc Girault (France Telecom)
Pr. Luc Vandendorpe (UCL - TELE) – Président

Louvain-la-Neuve, Belgique
2006

ABSTRACT

Abstract

Nowadays, the design of new cryptographic schemes follows the modern methodology of provable security. This approach requires to first clearly set out a formal model for the security of the scheme. Next, a "reductionist" proof is needed to show that the only way to break the scheme with a significant probability is either to attack an insecure component or to solve a computationally hard mathematical problem.

Pairings are bilinear mappings defined over groups wherein the discrete logarithm problem is hard. They are usually instantiated with carefully chosen elliptic curves. For the last couple of years, they have been found to provide plenty of applications in the design of cryptographic protocols. The most salient examples were probably the appearance of tripartite key agreement protocols, identity-based encryption schemes, where any arbitrary public identifier can be used as a public key, and digital signature schemes producing short signatures.

This thesis deals with pairing-based cryptographic protocols ensuring the same functionalities as the latter two kinds of primitives. In a first part, we deal with efficiency and (provable) security issues in identity based encryption (IBE) schemes and related concepts. We show two efficient variants of the Boneh-Franklin IBE and discuss the feasibility of using such schemes to construct of certificateless public key encryption systems.

A second part describes new digital signatures supporting identity-based public keys. One of these new constructions can be modified at a very low cost to simultaneously ensure the confidentiality of authenticated messages. The resulting signature/encryption scheme is supported by security proofs and enjoys an impressive efficiency for such a scheme.

The last part of this work considers several constructions to jointly achieve signature and encryption in a traditional public-key setting. For each of them, we give security reductions in a suitable model of security.

Acknowledgements

First of all, I would like to thank Jean-Jacques Quisquater for having trusted me and having accepted to be my advisor for this thesis. His suggestions, encouragements were always a precious support, as well as the freedom of action he allowed me.

I also would like to thank Andreas Enge and the Crypto team of the Laboratoire d'Informatique de l'école polytechnique that welcomed me in January 2004. I am also grateful to Marc Girault and Paulo Barreto, for having accepted to be members of the jury, as well as to Philippe Delsarte, for his participation in my advisoring committee, and Luc Vandendorpe for having accepted to preside the jury.

Working within the UCL Crypto Group members was a real pleasure. I would like to thank all of its current or former members and especially Sylvie Baudine for her availability, efficiency, and patient proofreading of this text. I have to also express my gratitude to Paulo Barreto and Noel McCullagh with whom I enjoyed to collaborate.

This research greatly benefitted from the excellent working conditions offered by the DGTRE's First Europe program that I would like to thank for its financial support.

As always, my family and friends offered me continuous support, each of them in their own way. My parents and my brothers were all, knowingly or not, a permanent support for me and deserve special thanks. I am also grateful to all of those who came to Paris to visit me during my 6 month-stay at LIX. Many acknowledgements also go to Damien Vergnaud, for his comments on the draft of a paper, and to Gregory Neven for the helpful discussions we had. I am finally indebted to Fabien Laguillaumie who kindly reviewed some chapters of this thesis.

Contents

List of Figures

List of Tables

Notations

Notations and abbreviations

Notations

$\mathbb{G}$	Cyclic group of prime order (depending on the context, cyclic groups will be noted additively $(\mathbb{G}, +)$ or multiplicatively $(\mathbb{G}, \cdot)$)
$1_{\mathbb{G}}$	Identity element for the group operation of $\mathbb{G}$
$\mathbb{G}^*$	Set of non-zero elements in group $\mathbb{G}$ (or $\mathbb{G} \backslash \{1_{\mathbb{G}}\}$ for short)
$\mathbb{Z}$	Set of integers
$\mathbb{N}$	Set of natural numbers
$\mathbb{Z}_q$	Set of integers $\{0, 1, \ldots, q-1\}$
$\mathbb{Z}_q^*$	Set of elements having a modular multiplicative inverse in $\mathbb{Z}_q$
$\mathbb{F}_r$	Finite field of r elements
$\in$	Membership of a set
$\in_R$	Membership of a randomly and uniformly distributed sample from a set
$:=$	Assignment of a value to a variable
$\xleftarrow{R}$	Assignment of a random and uniformly chosen value from a set
$\equiv$	Congruence modulo an integer
$\oplus$	Bitwise exclusive OR
e	Asymmetric pairing mapping
$\hat{e}$	Symmetric pairing mapping
$\{0,1\}^t$	Set of strings of t bits
$\{0,1\}^*$	Set of strings of arbitrary but finite length
$\perp$	Rejection symbol returned by a decryption algorithm to indicate that a ciphertext is invalid
$\|\|$	denotes the concatenation operation: $a\|\|b \in \{0,1\}^{n_0+n_1}$ stands for the concatenation of $a \in \{0,1\}^{n_0}$ and $b \in \{0,1\}^{n_1}$

Abbreviations

$1m$-CDHP	One more Computational Diffie-Hellman problem
ANON-IBSC-CCA	Ciphertext anonymity against chosen-ciphertext attacks for identity based signcryption
AUTH-IBSC-CMA	Ciphertext authentication against chosen-message attacks for identity-based signcryption
AUTH-SC-CMA	Ciphertext authentication against chosen-message attacks for signcryption
BDHP	Bilinear Diffie-Hellman problem
CA	Certification Authority
co-CDHP	co-Diffie-Hellman problem
co-BDHP	co-Bilinear Diffie-Hellman problem
co-DDHP	co-Decision Diffie-Hellman problem
CDHP	Computational Diffie-Hellman problem
CBE	Certificate-Based Encryption
CLE	Certificateless Encryption
CL-PKC	Certificateless Public Key Cryptography
CRL	Certificate revocation list
DLP	Discrete logarithm problem
DBDHP	Decision Bilinear Diffie-Hellman problem
DDHP	Decision Diffie-Hellman problem
DEM	Data Encapsulation Mechanism
ECDLP	Elliptic curve discrete logarithm problem
ECUF-IBSC-CMA	Existential ciphertext unforgeability against chosen-message attacks for identity-based signcryption
ESUF-IBSC-CMA	Existential signature unforgeability against chosen-message attacks for identity-based signcryption
ESUF-SC-CMA	Existential signature unforgeability against chosen-message attacks for signcryption
EUF-CMA	Existential unforgeability against chosen-message attacks
Gap-BDHP	Gap Bilinear Diffie-Hellman problem
GDHP	Gap Diffie-Hellman problem
HIBE	Hierarchical Identity-Based Encryption
IBE	Identity-Based Encryption
IBI	Identity-Based Identification
IBS	Identity-Based Signature
IBSC	Identity-Based Signcryption

IBUS	Identity-Based Undeniable Signature
ID-PKC	Identity-Based Public Key Cryptography
iff	if and only if
IND-CCA	Indistinguishability against chosen-ciphertext attacks
IND-CPA	Indistinguishability against chosen-plaintext attacks
IND-IBSC-CCA	Indistinguishability against chosen-ciphertext attacks for identity-based signcryption
IND-ID-CCA	Indistinguishability against chosen identity and ciphertext attacks
IND-ID-CPA	Indistinguishability against chosen identity and plaintext attacks
IND-SC-CCA	Indistinguishability against chosen-ciphertext attacks for signcryption
IND-sID-CCA	Indistinguishability against selective identity and chosen-ciphertext attacks
IND-sID-CPA	Indistinguishability against selective identity and chosen-plaintext attacks
NMA	No-message attack
KEM	Key Encapsulation Mechanism
KGC	Key Generation Center
p-BDHIP	p-Bilinear Diffie-Hellman Inversion problem
p-DHIP	p-Diffie-Hellman Inversion problem
OAEP	Optimal Asymmetric Encryption Padding
PKE	Public Key Encryption
PKCS	Public Key Cryptography Standards
PKG	Private Key Generator
PKI	Public Key Infrastructure
PPT	Probabilistic Polynomial Time
PSS-R	Probabilistic Signature Scheme with message Recovery
p-SDHP	p-Strong Diffie-Hellman problem
RMA	Random Message Attacks
SC-SUF-CMA	Strong ciphertext unforgeability against chosen-message attacks for signcryption
SC-INDK-CCA	Key indistinguishability against chosen-ciphertext attacks for signcryption
SSL	Secure Socket Layer
TA	Trusted authority

Introduction

1. Motivations and scope of this thesis

Over the last decades, the expansion of the Internet has significantly simplified the exchange of information between remote users. However, this in turn has led to an explosive growth in threats over the networks such as electronic eavesdropping, fraud, and identity thefts. In order to be immune against such threats, valuable resources must be protected in several ways: when two parties are communicating, they want to ensure that the data they exchange are not eavesdropped (confidentiality), that they are not altered by a third-party (data integrity), and that the sender is actually who he claims to be (entity authentication). Cryptography is the science that addresses these various concerns.

Two kinds of cryptographic schemes can be distinguished. The first one falls in the area of *secret key* or *symmetric* cryptography where remote entities communicate using a shared secret key that they both use to encrypt or decrypt messages. The second one pertains to the area of *public key* or *asymmetric* cryptography where each user has a public key made available on the Internet and a matching private key that should be computationally infeasible to recover given the public key. Asymmetric schemes enjoy the advantage that remote users do not have to first agree on a shared secret key before communicating: the sender only has to obtain his correspondent's public key to send him an encrypted message or verify his digital signature on a message. These properties are really attractive in networks of numerous users where shared symmetric keys would have to be distributed between each pair of user willing to communicate with each other if only symmetric cryptography primitives were available.

This thesis focuses on the design and the security of public key cryptographic schemes for privacy, authentication and sometimes simultaneously for both of these functionalities. Our work was more precisely

dedicated to the study of the applications of bilinear mappings that, although originally used for cryptanalytic purposes [147, 86], found a couple of applications in the design of public key cryptographic protocols [19]. Among those "positive applications", the most salient ones were perhaps the design of tripartite key exchange protocols [118], digital signature algorithms producing short signatures [43, 37, 227] and the construction of *identity-based encryption* schemes by Boneh-Franklin [40] and subsequently by other authors [35, 36, 38, 217]. An important part of our work was related to this kind of public key scheme wherein any arbitrary string such as email addresses can be used as public keys.

Many cryptographic protocols involve public key as well as secret key operations. The design of such reliable cryptographic protocols has turned out to be a highly non-trivial task. It must not only employ sturdy cryptographic primitives, but also has to integrate them in such a way that their security is preserved. A sound approach to evaluate the security of cryptographic schemes or protocols already exists. This approach is called "provable security" and stems from pioneering works by Goldwasser and Micali [90] and, later on, by Goldwasser, Micali and Rivest [105] respectively for public key encryption schemes and digital signatures. In the provable security approach, one ensures the security of a given cryptographic scheme by presenting a "reduction" between the properly defined security notion for the scheme and the intractability of some well-studied problem (such as the discrete logarithm or the integer factoring problem) that is thought to be difficult. Since Goldwasser and Micali's work [90], numerous achievements have followed this approach which has become a paradigm of cryptographic research. In accordance with it, we strove to provide security proofs in a reasonable model of computation for the various schemes that we studied.

2. Organization of the work

This thesis is divide into four distinct parts.

- **Part 1** contains material that will be used in the three other parts of the thesis.
  - Chapter 1 gives definitions for various discrete logarithm-related hard problems and intractability assumptions on which the security of our cryptographic schemes provably

relies. It also summarizes the properties of bilinear mappings that are used as building blocks of our schemes. In order to intuitively justify the relevance of the security models that we consider for our protocols, this chapter finally recalls the strongest formal security models for public key encryption and digital signatures.

- Chapter 2 introduces the concept of identity-based cryptography that is investigated in parts 2 and 3. It contains a brief summary of the main outstanding results that have been achieved in this striding out area.

- **Part 2** contains new results on identity-based encryption and the related certificateless encryption paradigm of Al-Ryami and Paterson [6].

  - Chapter 3 describes two variants of the Boneh-Franklin identity based encryption schemes. It shows how to speed up the decryption operation and/or shorten ciphertexts at the expense of having the security rely on a stronger assumption. Those results were published in the ACNS 2005 conference [136].
  - Chapter 4 describes recent unpublished results regarding generic constructions of certificateless encryption schemes using identity based cryptosystems. It also proposes a new pairing-based scheme that is slightly faster than the ones proposed so far.

- **Part 3** is about identity-based signatures (IBS).

  - Chapter 5 first proposes a new IBS scheme which is more efficient than all previous pairing-based proposals. It also gives a new security proof for an existing scheme already proven secure by Bellare, Namprempre and Neven [28]. The new proof enjoys a much tighter reduction than the original one. The first result is taken from a joint paper with Paulo Barreto and Noel McCullagh that was accepted at Asiacrypt 2005 [21]. The second result was never published and is only available on the Cryptology ePrint Archive [133].

- Chapter 6 describes a special purpose IBS which was the first example of an identity-based undeniable signature (IBUS) with a security proof in the random oracle model. It was published at the Cryptography Track of the RSA 2004 conference [134].
- Chapter 7 studies several schemes jointly performing signature and encryption in an identity-based public key setting. It summarizes results from a paper published at the 2003 Information Theory Workshop [131] and another paper (written with Paulo Barreto and Noel McCullagh) recently accepted at Asiacrypt 2005 [21]. We compare our schemes to other existing ones from efficiency and security points of view. Our second construction requires stronger computational assumptions than previous ones in its security proofs but happens to be the fastest scheme proposed so far.

• **Part 4** is devoted to the study of other cryptosystems for joint signature and encryption in a traditional public key setting.

- Chapter 8 describes three schemes based on various Diffie-Hellman related assumptions. The first two ones are constructed on signature schemes that make use of bilinear maps. They are taken from papers published respectively at PKC 2004 [132] and SCN 2004 [135]. The third scheme is constructed on the Schnorr signature scheme and has never been published. It does not rely on bilinear maps and can be instantiated with more general groups as we will see.

Part 1

Preliminary topics

CHAPTER 1

Generalities

Abstract. This chapter provides some formal definitions as well as technical background that will be used in the forthcoming chapters. These preliminary definitions are mainly related to bilinear maps, security notions for public key cryptography primitives and provable security.

1. Public key cryptography

The most spectacular development in the history of cryptography came in 1976 when W. Diffie and M. Hellman published their seminal paper [78] introducing the concept of public key cryptography where each entity has a public key e and a corresponding private key d which should be computationally infeasible to compute given e. This paper also provided a new and ingenious method for key exchange, the security of which is based on the intractability of the discrete logarithm problem. Such a paradigm avoids remote entities wishing to confidentially communicate to meet each other to agree on a secret encryption key.

Although Diffie and Hellman had no practical realization of a public-key encryption scheme at that time, the idea was clear and it enjoyed an extensive interest from the cryptographic research community. In 1978, Rivest, Shamir, and Adleman [187] discovered the first practical method, now referred to as RSA, to obtain public-key encryption and signature schemes. The RSA scheme is based on another hard mathematical problem, the intractability of factoring large integers. More precisely, the RSA assumption is the hardness of inverting the trapdoor permutation $f_{RSA}(x) = x^e \bmod N$, where $N = pq$ is the product of two large primes and e is an integer which is prime to $\varphi(N) = (p-1)(q-1)$, without knowing the prime factors p and q. The basic "textbook" RSA encryption of a message $x \in \mathbb{Z}_N^*$ was $c = x^e \bmod N$ and could be undone

by the legitimate recipient computing $x = c^d \bmod N$ using a private exponent d such that $ed \equiv 1 \pmod{\varphi(N)}$. Besides, the same trapdoor function also yields a technique to digitally sign a message $x \in \mathbb{Z}_N^*$: the signer simply uses his private exponent d to compute a signature $\sigma = f_{RSA}^{-1}(x) = x^d \bmod N$ that can be subsequently verified by checking that $x = \sigma^e \bmod N$. The discovery of RSA was quickly followed by the appearance of the Rabin [184] trapdoor function $f_{Rab}(x) = x^2 \bmod N$ which consists of a modular squaring using a composite modulus of secret factorization. This function is no longer a permutation but enjoys interesting advantages. Unlike RSA, inverting it is provably as hard as factoring the modulus $N = pq$ and it also turned out to be an interesting number theoretic primitive for encryption and signature.

Those applications of hard mathematical problems to cryptography revitalized efforts to find more efficient methods to factor large integers. The eighties saw major advances in this area but none which rendered the RSA system insecure. Another class of powerful and practical public-key schemes was found by El Gamal [96] in 1985. These are based on the discrete logarithm problem. The search for new public-key schemes, improvements to existing cryptographic mechanisms, and proofs of security is continuing at a rapid pace. The feasibility of secure cryptosystems based on other hard problems than factoring or computing discrete logarithms was studied with more or less success in the late seventies [145, 149] and more recently in [173, 4, 104, 112, 11]. Other more encouraging constructions based on assumptions related to factorization or discrete logarithms were also studied in the late nineties [159, 160, 169, 172].

2. Discrete logarithms and Diffie-Hellman problems

Since all cryptosystems studied in this thesis rely on groups where the discrete logarithm and Diffie-Hellman problems are supposed to be hard, we begin by first recalling their definition.

Definition 1.1 *For a security parameter $k \in \mathbb{N}$, let $(\mathbb{G}, \cdot)$ be a cyclic group of order $q > 2^k$ and let g be a generator of $\mathbb{G}$. The* ***discrete logarithm problem (DLP)*** *is, given a random $y \in_R \mathbb{G}$, to find the unique $x \in \mathbb{Z}_q$ such that $y = g^x$.*

Although easy in some particular cases (such as $(\mathbb{Z}_q, +)$), solving this problem is known to require exponential time in $k < \log_2|\mathbb{G}|$ in black-box groups [163, 197]. In prime order subgroups $\mathbb{G}$ of multiplicative groups $\mathbb{Z}_p^*$, the best known algorithms [148] take sub-exponential time in k. On subgroups of randomly chosen elliptic curves, the best known algorithms are exponential and elliptic curves are thus especially interesting for the implementation of cryptographic protocols as they allow shorter key sizes for the same security level w.r.t. implementations in finite fields $\mathbb{Z}_p^*$. By now, except for weaker kinds of curves such as 'anomalous' [193, 201] or 'supersingular' curves, 160-bit elliptic curve public keys offer about the same security as a 1024-bit RSA modulus.

In their founding paper [78] proposing the idea of public key cryptography, Diffie and Hellman introduced what is now usually called "the Diffie-Hellman problem" (DHP).

Definition 1.2 *Let $(\mathbb{G}, \cdot)$ be a cyclic group of order $q > 2^k$ and a generator g. The **computational Diffie-Hellman problem (CDHP)** is, given (g, g^a, g^b) for $a, b \in_R \mathbb{Z}_q^*$, to compute g^{ab}.*

Although not harder than DLP, this problem is commonly accepted as a hard problem: Nechaev [163] and Shoup [197] showed its exponential complexity for generic[1] algorithms. Salient results of Maurer and Wolf [142] additionally showed that in certain groups and under particular conditions, there exists a polynomial reduction from DLP to DHP.

The hardness of the computational Diffie-Hellman problem is referred to as "computational Diffie-Hellman assumption". A stronger assumption is formalized by the next definition that considers the problem of deciding whether or not a given group element is the solution of a Diffie-Hellman instance.

Definition 1.3 *Let $(\mathbb{G}, \cdot)$ be a cyclic group of order $q > 2^k$ and a generator g. The **decision Diffie-Hellman problem (DDHP)** is, given (g, g^a, g^b, g^c) for $a, b, c \in_R \mathbb{Z}_q^*$, to decide whether $c \equiv ab \pmod q$. In other words, the problem is to distinguish the distribution $D_1 =$*

[1] The generic model is an idealized computational model wherein an attacker may only access the group operations as black-box function calls, and may not meaningfully operate on encodings of group elements. While this model cannot capture the most efficient attacks on the discrete logarithm in specific instantiations of groups, it does provide evidence that the considered computational problems are hard.

$\{(g, g^a, g^b, g^{ab})|a, b \stackrel{R}{\leftarrow} \mathbb{Z}_q^*\}$ *of "Diffie-Hellman quadruples" from the distribution* $D_2 = \{(g, g^a, g^b, g^c)|a, b, c \stackrel{R}{\leftarrow} \mathbb{Z}_q^*\}$ *of "random quadruples".*

Both of these assumptions are necessary to establish the security (against passive attacks) of the famous Diffie-Hellman key exchange protocol which enable remote parties A et B to agree on a secret. The protocol consists for A in sending B (resp. for B in sending A) a group element g^a for a randomly chosen $a \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ (resp. a group element g^b for a random $b \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$) in such a way that A can eventually compute $K_{AB} = (g^b)^a$ (resp. B eventually computes $K_{AB} = (g^a)^b$).

Although Shoup [197] showed that there exists no generic algorithm to solve the DDHP in black box groups (and that a reasonable confidence can be invested into the DDH assumption), the decision Diffie-Hellman problem is believed to be strictly easier than its computational counterpart. Maurer and Wolf indeed showed in 1998 [141] that no generic algorithm reduces CDHP to DDHP. In other terms, there is a gap between the hardness of the computational Diffie-Hellman problem and the complexity of its decisional variant.

This observation led Okamoto and Pointcheval [168] to introduce the *Gap* Diffie-Hellman assumption that informally states that the CDH problem remains hard even in the presence of a black box algorithm solving DDH.

Definition 1.4 ([168]) *The* ***Gap Diffie-Hellman problem (GDHP)*** *in a cyclic group* $(\mathbb{G}, \cdot)$ *of order* q *is, given* (g, g^a, g^b) *for* $a, b \in_R \mathbb{Z}_q^*$*, to compute* g^{ab} *with the help of an oracle distinguishing the distributions* $D_1 = \{(g, g^a, g^b, g^{ab})|a, b \stackrel{R}{\leftarrow} \mathbb{Z}_q^*\}$ *and* $D_2 = \{(g, g^a, g^b, g^c)|a, b, c \stackrel{R}{\leftarrow} \mathbb{Z}_q^*\}$ *with probability* 1*.*

Although the intractability of the Gap Diffie-Hellman problem was much less studied than that of the computational problem, it is believed to be a reasonable assumption. The results of Maurer and Wolf [141] show that it holds in a generic model of computation [197].

3. Pairings

This subsection formally defines the tools that will be used by protocols described throughout this thesis. Those tools have enjoyed a tremendous interest from the research community. Since the seminal

paper published by Joux [118] who showed how they yield a one-round tripartite Diffie-Hellman protocol, pairings have provided a couple of applications (referenced in [19]) such as identity based encryption [40], short signatures [43, 37, 227],...

3.1. Formal definition of bilinear map groups

Concretely, a pairing is a bilinear map defined over elliptic curve subgroups. These groups were first thought to be unsuitable for cryptographic purposes because of the Menezes-Okamoto-Vanstone [147] and Frey-Rück reductions [86] that transform the ECDL problem on a curve $E(\mathbb{F}_r)$ into to a discrete logarithm problem in a small extension $\mathbb{F}_{r^\alpha}$ of the base field.

It indeed turns out that pairings are infeasible to compute over subgroups of random curves where the parameter α, called "embedding degree of the group"(i.e. the smallest integer α for which the order q of the group divides $r^\alpha - 1$), is usually huge. Any elliptic curve group must have a reasonably small embedding degree α in order for a bilinear map to be computable on it. Therefore, the MOV and Frey-Rück reductions are both known [146, 86] to take probabilistic polynomial time over pairing-friendly curves and to eventually provide probabilistic subexponential time algorithms for the ECDLP problem.

Nevertheless, using such curves is now believed to be reasonable if security parameters are adapted to render infeasible the computation of logarithms in the related extension field $\mathbb{F}_{r^\alpha}$. In 2000, Joux [118] showed how to employ pairing-friendly curves in his tripartite protocol and was followed by Boneh and Franklin [40] who showed a concrete implementation of their identity based encryption scheme using them.

Definition 1.5 *Let k be a security parameter and q be a k-bit prime number. Let us consider groups $(\mathbb{G}_1, +)$, $(\mathbb{G}_2, +)$ and $(\mathbb{G}_T, \cdot)$ of the same prime order q and let P, Q be generators of respectively $\mathbb{G}_1$ and $\mathbb{G}_2$. We say that $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ are **asymmetric bilinear map groups** if there exists a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ satisfying the following properties:*

1. *Bilinearity:* $\forall\ (S,T) \in \mathbb{G}_1 \times \mathbb{G}_2,\ \forall\ a, b \in \mathbb{Z}$, *we have the relation* $e(aS, bT) = e(S,T)^{ab}$.
2. *Non-degeneracy:* $\forall\ S \in \mathbb{G}_1,\ e(S,T) = 1\ \forall\ T \in \mathbb{G}_2$ *iff* $S = 1_{\mathbb{G}_1}$.

3. *Computability:* $\forall\ (S,T) \in \mathbb{G}_1 \times \mathbb{G}_2,\ e(S,T)$ *is efficiently computable.*
4. *There exists an efficient, publicly computable (but not necessarily invertible) isomorphism* $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ *such that* $\psi(Q) = P$.

Such bilinear map groups are known to be instantiable with ordinary elliptic curves such as MNT curves [155] or a kind of curves studied by Barreto and Naehrig [23]. In practice, $\mathbb{G}_1$ is a q-order cyclic subgroup of such a curve $E(\mathbb{F}_r)$ while $\mathbb{G}_2$ is a subgroup of $E(\mathbb{F}_{r^\alpha})$, where α is the embedding degree of the curve. The group $\mathbb{G}_T$ is the set of q^{th} roots of unity in the finite field $\mathbb{F}_{r^\alpha}$. In this case, the trace map can be used as an efficient isomorphism ψ as long as $\mathbb{G}_2$ is properly chosen [203] within $E(\mathbb{F}_{r^\alpha})$. It should be noted that some pairing based cryptographic schemes (e.g. [43, 37, 39]) do not explicitly require a computable isomorphism ψ but their security proof does.

The property of *computability* is ensured by Miller's famous and for a long time unpublished algorithm [152, 153] which will not be detailed here. In q-order cyclic subgroups of curves of embedding degree α, its complexity is dominated by $O(\log q)$ operations in the extension field $\mathbb{F}_{r^\alpha}$ containing the group $\mathbb{G}_T$. Computing a pairing is thus significantly more expensive than computing an elliptic curve scalar multiplication. Using a naive implementation of Miller's algorithm, a pairing computation is more than α^2 slower than a scalar multiplication on $E(\mathbb{F}_r)$. On the other hand, a recent paper by Scott [192] showed that most optimized algorithms for an embedding degree $\alpha = 2$ end up with a running time which is from twice to four times as long as an RSA decryption. However, pairing-based cryptographic protocols usually strive to minimize the number of pairing calculations they involve.

Some protocols (such as [99]) need symmetric pairings where $\mathbb{G}_1 = \mathbb{G}_2$ and ψ is the identity mapping.

Definition 1.6 *Let us consider groups* $(\mathbb{G}_1, +)$ *and* $(\mathbb{G}_2, \cdot)$ *of the same prime order* $q > 2^k$ *for a security parameter* $k \in \mathbb{N}$. *We say that* $(\mathbb{G}_1, \mathbb{G}_2)$ *are* ***symmetric bilinear map groups*** *if there exists a bilinear map* $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ *satisfying the following properties:*

1. *Bilinearity:* $\forall\ P, Q \in \mathbb{G}_1,\ \forall\ a, b \in \mathbb{Z},\ \hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$.
2. *Non-degeneracy:* $\forall\ P \in \mathbb{G}_1,\ \hat{e}(P, Q) = 1\ \forall\ Q \in \mathbb{G}_1$ *iff* $P = 1_{\mathbb{G}_1}$.
3. *Computability:* $\forall\ P, Q \in \mathbb{G}_1,\ \hat{e}(P, Q)$ *is efficiently computable.*

Those symmetric pairings have the additional commutativity property: for any pair $P, Q \in \mathbb{G}_1$, $\hat{e}(P,Q) = \hat{e}(Q,P)$. Admissible symmetric mappings of this kind can be derived from the Weil and Tate pairings using special endomorphisms called 'distortion maps' that are known [212] to only exist on a particular kind of curve termed 'supersingular' in the literature[2].

While supersingular curves probably allow the most efficient implementations of several protocols (such as [99] for instance), they may be more susceptible to attacks than ordinary curves. Indeed, several optimization tricks for them [20] require to use fields of small characteristic. The problem is that MOV and Frey-Rück reductions end up with a discrete logarithm problem in a finite field that is much easier to solve [67] than in fields of large characteristic and similar size. Such a threat is usually thwarted by increasing field sizes to maintain a sufficient level of security. That is why protocols where bandwidth requirements have to be minimized [43, 37, 39] usually avoid supersingular curves when possible.

This led several research papers (see [43, 41, 37, 39, 24, 93] for a few examples) to describe new cryptographic protocols in terms of asymmetric pairings fitting definition 1.5 and instantiable with ordinary curves. Although some of these scheme [43, 41, 37, 39, 93] do not explicitly require the existence of an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$, their security proofs need such a mapping to rely on a fairly natural assumption. Recent results of Smart and Vercauteren [203] highlighted that, if one is willing to benefit from the most efficient pairing calculation algorithms for ordinary curves [22], there is no known choice of groups $\mathbb{G}_2$ for which a computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ exists if a hash function $H : \{0,1\}^* \to \mathbb{G}_2^*$ should be able to efficiently map arbitrary strings onto $\mathbb{G}_2^*$ (without trivially multiplying a generator of $\mathbb{G}_2$ by a random multiplier) at the same time.

As a result, unless one is willing to accept somewhat unnatural computational assumptions for their security, it is likely that the most practical way to implement several pairing based protocols (including those

[2]In fact, a curve $E(\mathbb{F}_r)$ is said to be supersingular if its number of points $\#E(\mathbb{F}_r)$ is such that $t = q + 1 - \#E(\mathbb{F}_r)$ is a multiple of the characteristic of $\mathbb{F}_r$.

of [45, 24, 58, 131, 93]) is to use symmetric pairings. Unless stated otherwise, the protocols studied in the forthcoming chapters will be described using the kind of bilinear map groups that we deemed to be the most appropriate.

3.2. Consequences of pairings

Pairings have important consequences on the hardness of certain variants of the Diffie-Hellman problem. For instance, symmetric pairings lead to a strict separation between the intractability of the Computational Diffie-Hellman problem and the hardness of the corresponding decision problem.

Definition 1.7 *The* ***Computational Diffie-Hellman (CDH)*** *problem in symmetric bilinear map groups* $(\mathbb{G}_1, \mathbb{G}_2)$ *is to compute* $abP \in \mathbb{G}_1$ *given* $(P, aP, bP) \in \mathbb{G}_1^3$. *The* ***Decision Diffie-Hellman (DDH)*** *problem in symmetric bilinear map groups* $(\mathbb{G}_1, \mathbb{G}_2)$ *is, given* $(P, aP, bP, cP) \in \mathbb{G}_1^4$, *to decide whether* $c \equiv ab \bmod q$.

As noted by Joux and Nguyen [119], the DDH problem is easy in any symmetric bilinear map group. Indeed, to decide whether $c \equiv ab \bmod q$ given (P, aP, bP, cP), it suffices to check whether $\hat{e}(P, cP) = \hat{e}(aP, bP)$. Galbraith and Rotger [92] recently gave an algorithm to construct distortion maps and thus groups fitting definition 1.6 for any supersingular curve. It thus turns out that the DDH problem is easy on all such curves.

On the other hand, the CDH problem is still assumed to be hard on them where it becomes equivalent to Okamoto and Pointcheval's *Gap Diffie-Hellman* problem [168].

In [43], similar problems were formalized in the setting of asymmetric bilinear map groups. We note that the hardness of these problems does not depend on the existence of a computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$.

Definition 1.8 ([43]) *The* ***Computational co-Diffie-Hellman (co-CDH)*** *problem in asymmetric bilinear map groups* $(\mathbb{G}_1, \mathbb{G}_2)$ *is to compute* $abP_1 \in \mathbb{G}_1$ *given* $(P_1, P_2, aP_1, bP_2) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \times \mathbb{G}_2$. *The* ***Decision co-Diffie-Hellman (co-DDH)*** *problem in asymmetric bilinear map groups* $(\mathbb{G}_1, \mathbb{G}_2)$ *is, given* $(P_1, P_2, aP_1, bP_2, cP_1) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1$, *to decide whether* $c \equiv ab \bmod q$.

Similarly to what happens in symmetric pairing groups, the co-DDH problem is easy in asymmetric bilinear map groups: given as input $(P_1, P_2, aP_1, bP_2, cP_1)$, one can decide whether $c = ab$ by checking if $e(aP_1, bP_2) = e(cP_1, P_2)$. According to the terminology of [168], those groups are called *Gap co-Diffie-Hellman groups*. In chapter 8, we present two protocols taking advantage of the separation between the co-DDH and co-CDH problems.

We finally note that, while the hardness of the co-CDH and co-DDH problems does not depend on whether an isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ is computable, the Decision Diffie-Hellman problem in $\mathbb{G}_2$ (which is to decide whether (P_2, aP_2, bP_2, cP_2) satisfies $c \equiv ab \bmod q$) is easy whenever such an isomorphism exists. In contrast, very few things are known about the hardness of the DDH problem in $\mathbb{G}_1$ (i.e. distinguishing (P_1, aP_1, bP_1, abP_1) from (P_1, aP_1, bP_1, cP_1)) in asymmetric bilinear map groups. Although it appears to be hard as long as $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ is computationally one-way (such a requirement may hold for known instantiations of ψ with the trace map over MNT curves), its possible intractability was deemed 'a risky assumption' in [39]. Leaving the security of a cryptosystem rely on it would be hazardous as nothing is really known on the hardness of the DDH problem on MNT curves defined on their base field.

3.3. Bilinear map problems

Pairings also have their own related problems. The bilinear Diffie-Hellman problem was implicitly suggested by Joux in [118] and formalized by Boneh and Franklin [40].

Definition 1.9 ([118, 40]) *The **Bilinear Diffie-Hellman problem (BDHP)** in symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ is to compute $\hat{e}(P, P)^{abc} \in \mathbb{G}_2$ given $(P, aP, bP, cP) \in {\mathbb{G}_1}^4$.*
*The **co-Bilinear Diffie-Hellman problem (co-BDH)** in asymmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ is to compute $e(P_1, P_2)^{abc} \in \mathbb{G}_T$ given $(P_1, aP_1, bP_1) \in {\mathbb{G}_1}^3$ and $(P_2, aP_2, cP_2) \in {\mathbb{G}_2}^3$.*

These problems have both decisional counterparts (called DBDHP and co-DBDHP for short) which respectively consist in distinguishing

between the distributions

$$\begin{aligned} D_1 &:= \{(P, aP, bP, cP, \hat{e}(P,P)^{abc}) | a, b, c \stackrel{R}{\leftarrow} \mathbb{Z}_q^*\} \\ D_2 &:= \{(P, aP, bP, cP, \hat{e}(P,P)^{z}) | a, b, c, z \stackrel{R}{\leftarrow} \mathbb{Z}_q^*\} \end{aligned}$$

in symmetric bilinear map groups and

$$\begin{aligned} D_1 &:= \{(P_1, aP_1, bP_1, P_2, aP_2, cP_2, e(P_1,P_2)^{abc}) | a, b, c \stackrel{R}{\leftarrow} \mathbb{Z}_q^*\} \\ D_2 &:= \{(P_1, aP_1, bP_1, P_2, aP_2, cP_2, e(P_1,P_2)^{z}) | a, b, c, z \stackrel{R}{\leftarrow} \mathbb{Z}_q^*\} \end{aligned}$$

in asymmetric ones. The BDHP was used by Boneh and Franklin [40] to prove the security of their identity based encryption scheme. Its intractability as well as the hardness of the DBDHP are also necessary conditions for the security of Joux's tripartite protocol. Both assumptions were used to prove the security of many cryptographic constructions such as [7, 49, 35, 36, 45, 217].

4. Security models for public key cryptography primitives

The present section intends to recall formal models that are commonly believed to be required for public key encryption schemes and digital signatures. The first known methods to construct those primitives were the RSA [187] and Rabin [184] trapdoor functions that were recalled in the introduction of this chapter. As we will see, those mathematical tools alone do not suffice to provide cryptographic constructions reaching the required security levels.

4.1. Security notions for public key encryption

Formally, a public key encryption scheme is a triple $(\mathcal{K}, \mathcal{E}_{\mathsf{pk}}, \mathcal{D}_{\mathsf{sk}})$ where $\mathcal{K}(.)$ is a probabilistic key generation algorithm returning key pairs $(\mathsf{pk}, \mathsf{sk})$ on input of a security parameter k, $\mathcal{E}_{\mathsf{pk}}(.)$ is an (ideally probabilistic[3]) encryption algorithm producing ciphertexts from plaintexts (and optionally some randomness) while $\mathcal{D}_{\mathsf{sk}}(.)$ is the deterministic decryption algorithm that, on input of a ciphertext, either returns a plaintext or the special symbol $\perp$ if no plaintext corresponds to the ciphertext.

[3]As the rest of this thesis always considers probabilistic encryption algorithms, the notation $\mathcal{E}_{\mathsf{pk}}(m, r)$ shall always denote the result of encrypting the message m using the randomness r

The weakest security requirement for a public key encryption scheme is to be computationally one-way: given an encryption $c = \mathcal{E}_{\mathsf{pk}}(m)$ under a public key pk of some message m, it should be infeasible to find m without the private decryption key sk. Encryption schemes satisfying this (very weak) criterion are said to be *one-way against chosen-plaintext attacks* (OW-CPA) and the plain RSA encryption scheme where $\mathcal{E}_{\mathsf{pk}}(m) = m^e \bmod N$ is believed to be so. This security level is definitely not sufficient in practice: from an information theoretic point of view, ciphertexts are generally required not to reveal any bit of information about the clear message. It should even be infeasible to decide whether or not a ciphertext is an encryption of a given plaintext. That is why schemes such as the one of Goldwasser and Micali [90] wherein encryption algorithms are probabilistic provide more security than deterministic ones.

We here recall the definitions of *semantic security* [90] and *adaptive chosen-ciphertext security* [185] for probabilistic public key encryption schemes. Both notions are formalized by a so-called "find-then-guess" game where an adversary runs in two stages.

Definition 1.10 *A public key encryption scheme $(\mathcal{K}, \mathcal{E}_{pk}, \mathcal{D}_{sk})$ is secure against chosen-plaintext attacks (or has the IND-CPA security also called semantic security) if no probabilistic polynomial time (PPT) adversary has a non-negligible advantage in the game below:*

1. *Given k, the challenger $\mathcal{CH}$ runs the key generation algorithm $\mathcal{K}(k)$ to obtain a key pair $(\mathsf{pk}, \mathsf{sk})$. The adversary $\mathcal{A}$ is given the public key pk while the private key sk is kept secret.*
2. *(find stage) $\mathcal{A}$ outputs two messages m_0, m_1 and gets a ciphertext $C^* = \mathcal{E}_{pk}(m_b)$, for a random $b \stackrel{R}{\leftarrow} \{0,1\}$ chosen by $\mathcal{CH}$.*
3. *(guess stage) $\mathcal{A}$ eventually outputs a bit b' and wins if $b' = b$. Her advantage is*

$$Adv^{\mathtt{ind\text{-}cpa}}(\mathcal{A}) := 2 \times Pr[b' = b] - 1$$

where the probability is taken over the random choices of the challenger and the adversary.

The above definition intuitively captures the requirement that a ciphertext should reveal no information (not even a single bit) about the plaintext to the adversary in the sense that she should be unable to

distinguish which element of a very small plaintext set (of say two elements) corresponds to a given ciphertext. The basic "textbook" RSA encryption scheme obviously fails to reach this security level because it is deterministic and deciding which plaintext is encrypted by a given ciphertext is easy.

In realistic situations, adversaries might obtain information by observing plaintexts associated with particular ciphertexts. The previous model is thus believed to need some enhancements consisting in providing the adversary with a black box $\mathcal{D}_{\mathsf{sk}}(.)$ which is a decryption oracle extracting plaintexts from adversarially chosen ciphertexts. The resulting strengthened model is formalized by the following definition.

Definition 1.11 ([185]) *A public key encryption scheme $(\mathcal{K}, \mathcal{E}_{pk}, \mathcal{D}_{sk})$ is secure against adaptive chosen-ciphertext attacks (or has the IND-CCA2 security) if no probabilistic polynomial time (PPT) adversary has a non-negligible advantage in the game below:*

1. *Given k, the challenger $\mathcal{CH}$ runs the key generation algorithm $\mathcal{K}(k)$ to obtain a key pair $(\mathsf{pk}, \mathsf{sk})$. The adversary $\mathcal{A}$ is given the public key pk while the private key sk is kept secret.*
2. *(find stage) $\mathcal{A}$ is given access to a decryption oracle $\mathcal{D}_{\mathsf{sk}}(.)$ which, given a ciphertext C returns a plaintext m or a rejection message $\perp$. She may present her queries adaptively: each decryption request may depend on the obtained answers for previous ones. At some point, she outputs two messages m_0, m_1 and gets a ciphertext $C^* = \mathcal{E}_{\mathsf{pk}}(m_b)$, for $b \stackrel{R}{\leftarrow} \{0, 1\}$.*
3. *(guess stage) $\mathcal{A}$ issues new decryption queries on any ciphertext but C^*. She eventually outputs a bit b' and wins if $b' = b$. Her advantage is*

$$Adv^{\mathtt{ind\text{-}cca}}(\mathcal{A}) := 2 \times Pr[b' = b] - 1$$

where the probability is taken over the random choices of both $\mathcal{A}$ and the challenger.

The above definition obviously only makes sense if the adversary is prohibited from requiring the decryption of the challenge ciphertext C^* during the guess stage. This model was introduced by Rackoff and Simon [185] as an extension of the weaker notion of security against non-adaptive attacks (or IND-CCA1 security) that were initially considered

by Naor and Yung [162]. In these weaker attacks, adversaries are restricted to only issue decryption queries *before* receiving the challenge ciphertext. The latter non-adaptive security level was studied in 1990 as an enhancement of the notion of semantic security.

In order to give a concrete example of a cryptosystem that is secure in the sense of definition 1.10 but not in the scenario of definition 1.11, we mention the El Gamal encryption scheme [96] where ciphertexts have the form $\langle g^r, m \cdot y^r \rangle$, where g is the generator of a prime order group G and $(y = g^x, x)$ is the public/private key pair. Its semantic security is known [210] to be equivalent to the decision Diffie-Hellman (DDH) assumption if the message m is encoded as a group element (if $G = \langle g \rangle$ is a q-order subgroup of $\mathbb{Z}_p^*$, leaving plaintexts lie in $Z_p^* \backslash G$ would be insecure as $\mathcal{A}$ could simply win the game of definition 1.10 by choosing m_0 as a square in $\mathbb{Z}_p^*$ and m_1 as a non-square). On the other hand, the system is trivially insecure against adaptive chosen-ciphertext attacks. Indeed, a given challenge ciphertext $C = \langle c_1, c_2 \rangle$ can be turned into another encryption of the same message $C' = \langle c_1 \cdot g^{r'}, c_2 \cdot y^{r'} \rangle$ (for a random $r' \stackrel{R}{\leftarrow} \mathbb{Z}$) that an IND-CCA2 adversary can submit as a legal decryption query during the guess stage. This example illustrates the power granted to adversaries by leaving them access to the decryption oracle throughout the guess stage. The actual status of El Gamal against IND-CCA1 attacks is unknown.

In 1998, Bellare et al. [27] gave evidence that, in an *adaptive* chosen-ciphertext scenario where attackers may issue post-challenge decryption queries, the *indistinguishability* (IND-CCA2) property is equivalent to the *non-malleability* [83] (NM-CCA2) which is the computational infeasibility of turning a ciphertext into another one encrypting the same plaintext (as in the aforementioned attack against El Gamal) or a different plaintext which satisfies some known relation w.r.t. the first one.

4.2. Security notions for digital signatures

Digital signatures are the electronic counterpart of handwritten signatures for digital documents. They should be devised in such a way that changing a single bit to either m or s in a message-signature pair (m, s) renders this pair invalid.

More formally, a digital signature is a triple $(\mathcal{K}, \mathcal{S}_{\mathsf{sk}}, \mathcal{V}_{\mathsf{pk}})$ where $\mathcal{K}(.)$

is a probabilistic key generation algorithm returning key pairs $(\mathsf{pk}, \mathsf{sk})$ on input of a security parameter k, $\mathcal{S}_{\mathsf{sk}}(.)$ is a (possibly probabilistic) signature issuing algorithm and $\mathcal{V}_{\mathsf{pk}}(.)$ is the deterministic signature verification algorithm that, on input of a message m and a purported signature σ, returns either 1 or 0 depending on whether the signature is accepted or not.

Attacks against digital signature schemes can be classified according to the goals of the adversary and to the resources that she can use. The goals are diverse and include:

- Disclosing the private key of the signer. This is the most drastic attack. It is called a *total break*.
- Constructing an efficient algorithm that is able to sign any message without the private key with a significant probability. This is called a *universal forgery attack*.
- Finding a single message/signature pair. This is called the *existential forgery*.

Proper digital signature algorithms should be immune even to existential forgeries. The basic RSA signature scheme where signatures are computed as $\sigma = m^d \bmod N$ using the private exponent d are not secure against existential forgeries as anyone can simply choose $\sigma \xleftarrow{R} \mathbb{Z}_N^*$ and compute a matching message $m = \sigma^e \bmod N$ to obtain an existential forgery (m, σ). In order to obtain existentially unforgeable signatures from the RSA permutation, a suitable pre-processing (such as the Full Domain Hash and Probabilistic Signature Schemes described in [31]) involving hash functions should be applied to the message.

Regarding adversarial resources, three different attack scenarii exist. The weakest one is the *no-message attack* (NMA) wherein attackers only know the public key pk of the signer. In the second, the attacker has access to a list of valid message/signature pairs which she does not choose: the list contains messages randomly and uniformly chosen. This attack is termed *random-message attack* (RMA). The strongest attack scenario provides an attacker with an oracle producing signatures on arbitrary messages of her choice. Her endeavour, termed *adaptive chosen-message attack*, is to produce an existential forgery on a message for which she never obtains a signature from the oracle. This next definition formalizes

the de facto model of *existential unforgeability against chosen-message attacks* introduced by Goldwasser, Micali and Rivest [105].

Definition 1.12 *A digital signature scheme* $(\mathcal{K}, \mathcal{S}_{\mathsf{sk}}, \mathcal{V}_{\mathsf{pk}})$ *is secure against existential forgery against chosen-message attacks (or has the EUF-CMA secure) if no PPT adversary* $\mathcal{F}$ *has a non-negligible advantage in the game below:*

1. *Given* k*, the challenger* $\mathcal{CH}$ *runs the key generation algorithm* $\mathcal{K}(k)$ *to obtain a key pair* $(\mathsf{pk}, \mathsf{sk})$*. The forger* $\mathcal{F}$ *is given the public key* pk *while the challenger keeps* sk *to itself.*
2. $\mathcal{F}$ *is given access to a signature oracle* $\mathcal{S}_{\mathsf{sk}}(.)$ *which, given a message* m *returns a signature computed on* m *using the private key* sk*. She may present her queries adaptively: each signing query may depend on the results of previous ones.*
3. $\mathcal{F}$ *outputs a pair made of a message* m *and a purported signature* σ *on* m*. She wins the game if the verification algorithm* $\mathcal{V}_{\mathsf{pk}}(.)$ *returns* 1 *for the pair* (m, s) *and* m *was never the input of a signing query during the game. Her advantage is defined as her probability of victory taken over her coin tosses and those of the challenger.*

We also mention the existence of a stronger form of existential unforgeability introduced in [9] and only useful for some specific applications. In this model, the forger is allowed to produce a signature σ on a message previously queried to the signing oracle. The restriction is that the pair (m, σ) must differ from all pairs produced by the signing oracle at stage 2. However, the standard form of existential unforgeability against chosen-message attacks is widely believed to be sufficient for most practical applications.

5. Provable security

Along the ever-increasing interest of the research community in public key cryptography, there has been a graduate evolution tending to a necessity to provide security proofs for asymmetric cryptosystems in the sense that the existence of an attacker against them would imply a probabilistic polynomial time algorithm to solve a hard number theoretic

problem. The commonly assumed intractability of the latter implies the non-existence of the attacker.

5.1. Hash functions

Hash functions are one of the most fundamental tools in cryptography. They essentially allow to produce digests of fixed length from arbitrary sequences of bits. They can very efficiently map strings of (ideally) arbitrary length onto elements of particular encodings such as finite field elements or elliptic curve points. Except when their range has a special algebraic structure (such as the cyclic subgroup of a finite field or an elliptic curve), they can be implemented much more efficiently than even simple arithmetic operations such as a modular exponentiation.

Those functions are obviously not injective (their range being usually much smaller than their domain). Nevertheless, they must be devised in such a way that finding collisions (that is two domain elements having identical images) is computationally infeasible. More precisely, a hash function $h : \{0,1\}^* \to \{0,1\}^\ell$ must have the following properties, the strongest of those being the last one.

(1) Pre-image resistance: given $x \in \{0,1\}^\ell$ it should be infeasible to find an element $m \in \{0,1\}^*$ such that $x = h(m)$.
(2) Second pre-image resistance: given $m \in \{0,1\}^*$, it must be computationally infeasible to find $m' \neq m \in \{0,1\}^*$ such that $h(m) = h(m')$.
(3) Collision resistance: it should be computationally infeasible to find a pair $m, m' \in \{0,1\}^*$ such that $h(m) = h(m')$.

The design of hash functions fitting the above constraints is highly non-trivial. The last few months saw the discovery of several attacks against widely employed hash functions. The very popular MD-5 [188], and SHA-1 [164] hashing algorithms were very recently proved [216, 215] not to be collision resistant.

5.2. The random oracle model

In 1993, motivated by the perspective of proving the security of efficient protocols, Bellare and Rogaway introduced the random oracle

model [29] that was previously implicitly suggested in [84]. In this computational model, hash functions are used as black box by attackers for whom they are indistinguishable from perfectly random functions. For completeness, we here recall a definition taken from [29].

Definition 1.13 *Formally, a random oracle R is a map from $\{0,1\}^*$ to $\{0,1\}^\infty$ chosen by selecting each bit of $R(x)$ uniformly and independently, for every x. Of course no actual protocol uses infinitely long outputs, this just saves us from having to say how long "sufficiently long" is. When restricting ourselves to functions of fixed-length ranges, a random oracle $H : \{0,1\}^* \to \{0,1\}^k$ is a random hash function of range $\{0,1\}^k$ and the domain of which is a set of strings of variable lengths.*

In reductionist security proofs, the output of a random oracle is unpredictable by an adversary. The latter is thus provided with an oracle access to random oracles: whenever she wants to know the value of a hash function at some point of its domain, she has to explicitly query an oracle producing a random output for each new input[4].

In security reductions, the behaviour of these oracles is simulated by an algorithm attempting to solve a *random* instance of a hard number theoretic problem. Random oracles are indeed "programmed" in such a way that parts of the random-looking inputs of the hard problem are passed to the adversary with the hope that the latter's result will yield the searched solution. This model of computation thus allows additional degrees of freedom in security proofs.

Although it is well known that security in the random oracle model does not imply security in the real world as showed by several papers (see [48] or [26] for instance) exhibiting pathological cases of provably secure schemes for which no secure implementation exists, it still seems to be "a good engineering principle" to give security proofs 'at least' in the random oracle model when proposing a new asymmetric cryptosystem. It gives heuristic arguments rather than formal proofs that the protocol is not inherently flawed.

[4]When multiple random oracles are involved in a protocol, all of these have to be independently selected. Simple natural encodings provide as many independent random oracles as needed from a single one.

6. The standard model

Several cryptographic protocols do not employ random oracles and have security proofs "in the real-world", but their security usually relies on stronger assumptions and/or they are computationally more expensive than random oracle-using schemes. The most famous and practical secure constructions in the standard model are the Cramer-Shoup public key encryption [71, 74] and signature schemes [72] and the short signatures of Boneh and Boyen [37]. The early nineties saw other cryptosystems such as the one of Dolev, Dwork and Naor [83] that are also provably IND-CCA2 in the standard model but they are too inefficient to be used in practice. The Cramer-Shoup proposal was the first fairly efficient one but it is still more than twice slower than El Gamal [96]. Its generalization [73] and the variant given by Kurosawa and Desmedt [123] are also expensive. The Cramer-Shoup signature [72] has a security proof under a stronger assumption than the hardness of inverting the RSA function and it is also twice slower than RSA-based signature schemes in the random oracle model [31]. Finally, the Boneh-Boyen signature scheme [37] enjoys a great efficiency for a construction that is provably secure in the standard model (a signature generation requires a single elliptic curve scalar multiplication whereas a verification entails a single pairing computation). Nevertheless, its security relies on a new strong Diffie-Hellman-related assumption which we also use in chapter 8.

Regarding provable security in the standard model, we also mention a certain form of identity based encryption schemes [195, 40] that can be transformed into public key encryption which are provably secure against chosen-ciphertext attacks in the standard model. More details about them will be given in the next chapter.

CHAPTER 2

Identity based cryptography

Abstract. We give formal definitions of identity based encryption and signature schemes. We provide motivations for the concept as well as several examples of constructions and discuss their efficiency.

1. Introduction

Since the appearance of public key cryptography in 1976, the threat of "man-in-the-middle attacks" has been a great concern for the research community that was led to devise certification methods to provide users with confidence in the authenticity of public keys they are using. These guarantees took the form of digital certificates signed by trusted entities called Certification Authorities (CAs) which aimed at vouching for the fact that a given public key actually belongs to its alleged owner. It was the birth of Public Key Infrastructures (PKIs) that deployed mechanisms to manage these digital certificates throughout the lifetime of their corresponding keys. Unfortunately, these certificate-based infrastructures turned out to be very heavy to deploy, cumbersome to use and non-transparent for the end-user. Indeed, trust problems arise when a digital certificate is signed by an authority whose public key is not already trusted by the user of the certified public key: in such a case, the user is led to validate an entire chain of digital certificates before acquiring confidence in the authenticity of a given key and, furthermore, finding such a chain of certificates between the enquired key and a trusted one is not a trivial problem. The treatment of certification paths has also been a critical issue in PKIs and softwares like web browsers are sometimes unable to deal with it. This might incur real security concerns at the establishment of an SSL connection, when the certificate of the server has to be validated by the client's browser.

Other problems with PKIs is the fact that certificates are not delivered for free in many situations and their cost is making public key owners reluctant to enquire for them and, from a robustness point of view, their lack of fault tolerance: when a private key exposure happens, no easy solution allows to repair it nor to limit the damage it involves: the corresponding certificate must be invalidated by using black lists called Certificate Revocation Lists (CRLs) that must be periodically checked by users who want to ensure that the key they are about to use has not been compromised.

In order to bypass the trust problems encountered in conventional Public Key Infrastructures, Shamir introduced in 1984 [195] the concept of identity based cryptography where a public key can be a binary string identifying its owner non-ambiguously (e.g. an e-mail address, an IP address combined to a user-name, a social security number,...). The motivation of this kind of scheme was to simplify key management and remove the need of public key certificates as much as possible: since a key is the identity of its owner, there is no need to bind them by a digital certificate and a public repository containing a list of user names and their associated public keys becomes useless since public keys are human-memorizable. End users do not have to enquire for a certificate for their public key. The only things that still must be certified are the public keys of trusted authorities called private key generators (PKGs) that have to generate private keys associated to users' identities thanks to their secret key (unlike conventional public key schemes, users do not generate their key pair themselves). This does not completely remove the need of certificates but, since many users depend on the same authority, this need is drastically reduced.

One inconvenience of these systems is their inherent key escrow feature. Indeed, since trusted authorities called private key generators (PKGs) have to deliver private keys to users after having computed them from their identity information and from a master secret key, these PKGs are able to sign messages on behalf of or to decrypt ciphertexts intended to any user depending on them. This key escrow property led the research community to investigate alternative paradigms keeping the advantages of identity based cryptography without involving an authority in which a too great amount of confidence must be invested. Among others, we mention the works of Girault [101], Brown et al. [46] and,

more recently, Gentry [98] and Al-Riyami-Paterson [6]. The last model will be discussed further in chapter 4.

Several practical solutions for identity based signatures (IBS) rapidly appeared after 1984 [84, 106, 190] but finding a practical identity based encryption scheme (IBE) remained an open challenge until 2001 despite some attempts [77, 143, 115, 211, 209]. The latter proposals either require tamper-proof hardware, expensive private key generation operations for PKGs or end-users who are assumed not to collude to expose the authority's master key. The first practical construction came in 2001 when Boneh and Franklin [40] proposed to use pairings to achieve an elegant identity based encryption method. Another one was suggested by Cocks [66] the same year. This second method relies on simpler mathematics but, as we will see, it turns out to be much less practical regarding the size of ciphertexts.

Other identity based signature and key agreement schemes based on pairings were proposed after 2001 ([51],[202],[111],...). We also mention the existence of many other proposals of identity based cryptographic protocols from pairings. They are not discussed here but are referenced in [19].

In this chapter, section 2 formally defines the concepts IBE and IBS schemes. The famous Boneh-Franklin pairing based IBE scheme is described in section 3. Section 4 then gives two example of identity based protocols which do not use pairings but rather simple modular arithmetic operations. For completeness, section 5 then discuss the hierarchical extension of the concept of identity based encryption. Some recent results about provably secure identity based encryption in the standard model are finally summarized in section 6.

2. Components of identity based cryptosystems

We here recall the formalism introduced in [40] for identity based encryption. Such a primitive consists of the following algorithms.

Setup: is a probabilistic algorithm run by a private key generator (PKG) that takes as input a security parameter to output a public/private key pair (P_{pub}, mk) for the PKG (P_{pub} is its public key and mk is its master key that is kept secret).

Keygen: is a key generation algorithm run by the PKG on input of a master key mk and a user's identity ID to return the user's private key d_{ID}.

Encrypt: this probabilistic algorithm takes as input a plaintext M, a recipient's identity ID and the PKG's public key P_{pub} to output a ciphertext C.

Decrypt: is a deterministic decryption algorithm that takes as input a ciphertext C and the private decryption key d_{ID} to return a plaintext M or a distinguished symbol $\perp$ if C is not a valid ciphertext.

In the case of identity based signatures, the last two algorithms are

Sign: given a message M, the PKG's public key and a private key d_{ID}, the signature generation algorithm generates a signature on M.

Verify: is a signature verification algorithm that, given an alleged signature σ on a message M for an identity ID, outputs either 1 or 0 depending on whether the signature is acceptable or not.

3. The Boneh-Franklin identity based encryption scheme

Figure 2.1 describes the simplest version of the scheme. This version is only provably secure against chosen-plaintext attacks and has some similarities with El Gamal's cryptosystem [96]. Boneh and Franklin showed that applying the Fujisaki-Okamoto generic transformation [89] allows turning this basic scheme into a chosen-ciphertext secure one in an extended security model (detailed in chapter 3).

3.1. The scheme

The consistency is easy to check: if the sender correctly encrypted the message, we have $U = rP$ and

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

Boneh and Franklin [40] proved in the random oracle model that, when padded with the Fujisaki-Okamoto transformation [89], the above scheme is secure against adaptive chosen-ciphertext attacks if the Bilinear Diffie-Hellman problem is hard (although a flaw in the security reduction was recently discovered and fixed in [93]).

The crucial information is the PKG's master key: all the system's

Setup: given a security parameter k, the PKG chooses symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of prime order $q > 2^k$, a generator P of $\mathbb{G}_1$, a master key $s \in \mathbb{Z}_q^*$ and the associated public key $P_{pub} = sP$. The plaintext space is $\mathcal{M} = \{0,1\}^n$ for a fixed n while $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n$ is the ciphertext space. Hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1^*$ and $H_2 : \mathbb{G}_2 \to \{0,1\}^n$ are also chosen. The system public parameters are

$$\mathsf{params} = (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2).$$

Keygen: given identity $\mathsf{ID} \in \{0,1\}^*$, the PKG computes $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in \mathbb{G}_1$ and $d_{\mathsf{ID}} = sQ_{\mathsf{ID}} \in \mathbb{G}_1$ that is given to the user as a private key.

Encrypt: to encrypt a message $M \in \mathcal{M}$, the sender uses the recipient's public identifier $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in \mathbb{G}_1$ as follows:

(1) He/she picks a random $r \in \mathbb{Z}_q^*$.
(2) He/she computes $g_{\mathsf{ID}} = \hat{e}(P_{pub}, Q_{\mathsf{ID}}) \in \mathbb{G}_2$.

The ciphertext $C = \langle rP, M \oplus H_2(g_{\mathsf{ID}}^r) \rangle$ is sent to the receiver.

Decrypt : given a ciphertext $C = \langle U, V \rangle$, the receiver uses his/her private key d_{ID} to decrypt by computing $M = V \oplus H_2(\hat{e}(U, d_{\mathsf{ID}}))$.

FIGURE 2.1. The BasicIdent scheme

privacy is compromised if that master key is ever stolen by an attacker. In order to avoid having a single point of failure and remove the built-in key escrow, Boneh and Franklin showed it was possible to split the PKG into several partial PKGs in such a way that these partial PKGs jointly generate a discrete logarithm key pair in a threshold fashion and each of them eventually holds a share of the master key. Users then have to visit a minimum of t-out-of-n honest PKGs to obtain a share of their private decryption key. These shares can then be recombined into a full decryption key thanks to a Lagrange interpolation as in Shamir's secret sharing scheme [194]. An alternative to this approach was suggested by Chen et al. [57] who imagined a setting with n different PKGs, each having their own master key/public key pair and issuing private keys associated to users' identities independently: a user's full private key was simply the sum of the n received independent private keys while the full public key of the scheme was the sum of the n PKGs' public keys.

3.2. Any IBE implies a digital signature

As explained in [40], any IBE system can be generically turned into a signature scheme where a signature σ on a message M is simply the private key (computed according to the Keygen algorithm) for the identity M (the public key being the system-wide public key P_{pub}). It can be verified by simply encrypting a random message M' under P_{pub} for the identity M and the signature is accepted if the obtained ciphertext decrypts into M' when using σ as a decryption key.

The verification algorithm is thus somewhat surprisingly probabilistic. Nevertheless, the description of the signature scheme can here be re-written in such a way that a signature $\sigma = sH_1(M) \in \mathbb{G}_1$ can be verified by checking that $\hat{e}(P, \sigma) = \hat{e}(P_{pub}, H_1(M))$ where s is the private key and P_{pub} plays the role of the public key. When implemented with asymmetric bilinear map groups over MNT curves [155], the latter signature algorithm is nothing but the Boneh-Lynn-Shacham signature [43] that provides signatures as short as 171 bits for the same security level as 320-bit Schnorr signatures [191].

3.3. Another pairing-based IBE

In 2003, Sakai and Kasahara [189] suggested a different IBE scheme using bilinear maps. In the simplest description of their scheme, the private key associated to an identity ID is $d_{\mathsf{ID}} = (1/(s+h_1(\mathsf{ID})))P$, where s is the master key and $h_1 : \{0,1\}^* \to \mathbb{Z}_q^*$ denotes a hash function. The key extraction algorithm implicitly defines a signature scheme where signatures can be verified by checking that $\hat{e}(h_1(\mathsf{ID})P + P_{pub}, d_{\mathsf{ID}}) = \hat{e}(P, P)$. The security of this signature scheme was proved by Boneh and Boyen [37] and independently by Zhang et al. [227].

Their IBE system encrypts ciphertexts having the form

$$C = \langle U, V \rangle = \langle rh_1(\mathsf{ID}) + rP_{pub}, M \oplus h_2(\hat{e}(P, P)^r) \rangle$$

for a random $r \in \mathbb{Z}_q^*$. The plaintext can be recovered by computing $M = V \oplus h_2(U, d_{\mathsf{ID}})$. The scheme has the advantage not to involve a pairing calculation in the encryption algorithm (as $\hat{e}(P, P)$ can be included among the system-wide public parameters) nor to hash identities onto a cyclic elliptic curve subgroup: it only uses (more efficient) standard hash functions having a finite field as a range.

On the other hand, the security of the Sakai-Kasahara IBE relies on a stronger computational assumption than the BDH assumption. This stronger assumption will also be used in chapters 4 and 7 of this thesis. A security proof was given by Chen and Cheng [55] who adapted the techniques of Boneh-Boyen [37, 35] in a fairly straightforward manner.

4. Identity based schemes from simple modular arithmetic

This section shows that identity based schemes can be devised without pairings by presenting two simple examples: the Guillou-Quisquater [106] IBS and Cocks's IBE scheme [66]. Both are obtained from modular arithmetic and their security relies on the intractability of factoring large integers. The first one uses the RSA trapdoor permutation while the second one is based on quadratic residues.

4.1. The Guillou-Quisquater signature scheme

This scheme is derived from a three round identification scheme. It was proposed in 1988 and is depicted in figure 2.2.

To verify the consistency of the scheme, we note that

$$u \equiv s^e I^\ell \equiv (ka^\ell)^e I^\ell \equiv k^e (a^e I)^\ell \equiv k^e \equiv r \pmod{n}.$$

Hence $u = r$ and then $h(m\|u) = h(m\|r)$. This signature scheme is obtained from the Guillou-Quisquater identification protocol (GQ) using the Fiat-Shamir heuristic [84][1]. That is why the output of the hash function h must be smaller than e (the set of challenges is $\mathbb{Z}_e$ in the underlying identification protocol).

The public exponent is taken as a prime for provable security purposes. The redundancy function R can be instantiated with a hash function. It aims at preventing attacks that could take advantage of multiplicative relations between identities. In order to avoid birthday

[1] A canonical 3-move identification scheme is a protocol (between a prover and a verifier) producing transcripts $(\mathsf{Cmt}, \mathsf{Ch}, \mathsf{Rsp})$ which are "proofs of knowledge" of a private key sk. They are made of a commitment Cmt computed by the prover using a secret value r, a challenge Ch sent by a verifier and a response Rsp computed by the prover using the secret value r, the challenge and his private key sk. The transcript is validated (or rejected) by the verifier using the public key pk of the prover. The Fiat-Shamir heuristic turns such an interactive proof into a signature scheme by setting the challenge Ch as a hash value of the message to sign and the commitment Cmt.

Setup: given a security parameter k_0, the private key generator (PKG) picks two $k_0/2$-bit primes p and q and computes $n = pq$. It also picks a prime number $e \in \mathbb{Z}_{\varphi(n)}$ such that $\gcd(e, \varphi(n)) = 1$ and chooses a cryptographic hash function $h :\rightarrow \mathbb{Z}_e$ and a redundancy function $R : \{0,1\}^* \rightarrow \mathbb{Z}_n^*$. The pair (n, e) is its public key while the pair (p, q) is kept secret and is its master key. The public parameters are

$$\mathsf{params} := \{k_0, n, e, R, h\}$$

Keygen: given a user's identity ID, the PKG computes $I = R(\mathsf{ID}) \in \mathbb{Z}_n^*$ and $a \in \mathbb{Z}_n^*$ such that $Ia^e \equiv 1 \pmod{n}$. The obtained a is returned to the user as a private key.

Sign : given a message M, the signer does the following:

(1) Pick a random $k \leftarrow_R \mathbb{Z}_n^*$ and compute $r = k^e \bmod n$
(2) Compute $\ell = h(M\|r) \ \in \mathbb{Z}_e$
(3) Calculate $s = ka^\ell \bmod n$

The signature on m is the pair (s, ℓ)

Verify: : to verify a signature (s, ℓ) on M,

(1) Compute $I = R(\mathsf{ID})$ from the signer's identity ID.
(2) Compute $u = s^e I^\ell \bmod n$.
(3) Accept the signature if $\ell = h(M\|u)$.

FIGURE 2.2. The GQ-IBS scheme

attacks on the hash function, it is recommended to use public exponents e of at least 160 bits (in the corresponding identification scheme, shorter exponents are allowed). The security parameters should be at least 1024 or 2048 to avoid attacks trying to factor the modulus.

The GQ signature scheme can be proved to be existentially unforgeable provided it is hard to invert the RSA function by using the proof technique of Pointcheval and Stern [182, 183]. The security of the underlying identity based identification (IBI) protocol was studied by Bellare, Namprempre and Neven [28] in a suitable model of security for IBI schemes.

From a general point of view, all identity based signatures can be viewed as non-interactive *proofs of knowledge* of a signature on a message which is nothing but the signer's identity. In the present case, the GQ signature scheme is a non-interactive proof of knowledge of an RSA signature. Many pairing-based IBS schemes are rather proofs of knowledge

of a Boneh-Lynn-Shacham [43] signature on the signer's identity.

The GQ scheme is one of the most efficient known identity based signatures. Several other known IBS (see [33, 84, 170, 85, 28] for instance) have a comparable efficiency as they only use simple arithmetic operations. In contrast, the only known reasonable proposal of IBE system using such simple mathematics was discovered by Cocks [66] in 2001. It unfortunately suffers from being highly bandwidth-consuming as bits of plaintext have to be separately encrypted. This prevents it from being really usable in practice. This illustrates that IBE systems are generally harder to construct than IBS schemes.

4.2. Cocks's IBE

This encryption scheme was discovered by Cocks [66] roughly at the same time as the Boneh-Frankin IBE. It is based on quadratic residues and on the properties of the Jacobi symbol for Blum integers (i.e. composite numbers n that are a product of two primes p and q such that $p \equiv q \equiv 3 \pmod 4$). The signature scheme that is implicitly used in the private key extraction algorithm is a Rabin-type [184] signature.

For encrypting 128-bit symmetric keys, the scheme is reasonably computationally cheap: the sender's computing time is dominated by 2×128 Jacobi symbol evaluations and 2×128 modular inversions. The receiver just has to compute 128 Jacobi symbols since he/she knows which of a or $-a$ is the square of his/her private key. The drawback of the scheme is its bandwidth overhead: for a 1024-bit modulus n and a 128-bit symmetric transport key, at least 2×16 Kb need to be transmitted if all encrypted key bits are sent together. Ciphertexts are eventually 2048 times longer than encrypted symmetric keys.

Cocks did not give a security proof in a formal model but informally showed that his construction is secure against chosen-plaintext attacks under the Quadratic Residuosity Assumption (i.e. the hardness of deciding whether or not a random integer a such that $\left(\frac{a}{n}\right) = 1$ is a square or not).

Setup: the PKG picks prime numbers p and q such that $p \equiv q \equiv 3 \pmod 4$, computes their product $n = pq$ that is made public together with a hash function $H : \{0,1\}^* \to \mathbb{Z}_n^*$. The PKG's master key is (p, q).

Keygen: given an identity ID, the PKG computes a chain of hash values starting from ID until obtaining $a = H(H(H \dots (\mathsf{ID}))) \in \mathbb{Z}_n^*$ such that $\left(\frac{a}{n}\right) = 1$. For any such $a \in \mathbb{Z}_n^*$, either a or $-a$ is a square in $\mathbb{Z}_n^*$. It is easy to verify that $r = a^{\frac{n+5-(p+q)}{8}} \mod n$ satisfies $a = r^2 \bmod n$ or $a = -r^2 \bmod n$ depending on whether $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = 1$ or $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = -1$. The obtained r is returned to the user as a private key.

Encrypt: the sender A ignores which of a or $-a$ is a square in $\mathbb{Z}_n^*$. We first assume we are in the case $a = r^2 \bmod n$. A generates a symmetric transport key K and encrypts the plaintext M with it. Each bit x of that symmetric key is then encrypted before being sent to the receiver B. To do this, A encodes x in $\{-1, 1\}$ rather than in $\{0, 1\}$ and does the following.

(1) Pick a random $t \in \mathbb{Z}_n^*$ such that $\left(\frac{t}{n}\right) = x$.
(2) Compute $s = (t + \frac{a}{t}) \bmod n$ (since $\left(\frac{t}{n}\right) \neq 0$, t is invertible in $\mathbb{Z}_n$) and send it to B.

Since A does not know which of a or $-a$ is the square of B's decryption key, A has to repeat the above process for a new t and, this time, send $s = (t - a/t) \bmod n$. Hence, $2|n|$ bits, where $|x|$ denotes the bitlength of x, have to be transmitted for each bit of the symmetric key.

Decrypt: B recovers x as follows. Given that

$$t(1 + r/t)^2 \equiv t + 2r + \frac{r^2}{t} \equiv t + 2r + \frac{a}{t} \equiv s + 2r \pmod n,$$

B can compute $\left(\frac{s+2r}{n}\right) = \left(\frac{t}{n}\right) = x$ and recover x using his/her private key r thanks to the multiplicative properties of the Jacobi symbol. Once the symmetric key K is obtained in clear, the ciphertext can be decrypted.

FIGURE 2.3. The Cocks IBE scheme

5. Hierarchical identity based cryptography

A shortcoming of the Boneh-Franklin IBE is that in a large network, the PKG's key generation task rapidly becomes a bottleneck when many private keys have to be computed and secure channels have to be established to transmit them to their legitimate owner. To overcome this

problem, a solution is to set up a hierarchy of PKGs in which each PKG only computes private keys for entities (other PKGs or end-users) immediately below it in the hierarchy. In such hierarchies, entities are represented by a vector of identifiers (i.e. a concatenation of their identifier to those of all their ancestors' ones: for example a child of $\langle \mathsf{ID}_1, \ldots, \mathsf{ID}_i \rangle$ has an address $\langle \mathsf{ID}_1, \ldots, \mathsf{ID}_i, \mathsf{ID}_{i+1} \rangle$) instead of a single identifier as in the Boneh-Franklin scheme.

5.1. The Gentry-Silverberg hierarchical scheme

In figure 2.4, we give an example, proposed by Gentry and Silverberg [99], of such a hierarchical IBE (HIBE) that can be viewed as a scalable extension of Boneh and Franklin's proposal (both schemes are identical if the hierarchy has a single level). Unlike another 2-level hierarchical scheme proposed by Horwitz and Lynn [113], this one supports multiple levels. Lower-level PKGs (i.e. PKGs other than the Root PKG located at the top of the hierarchy) generate private keys for their children by using some information coming from their ancestors together with a private information that is only known to them. Each of them then adds some information to the secret parameters of their children.

In our notation, $Level_i$ is the set of entities at level i, $Level_0$ denotes the sole Root PKG. The simplified version of the scheme is made of the following algorithms.

The consistency of the scheme follows from the following equations:

$$\begin{aligned} \hat{e}(U_0, S_t) &= \hat{e}(P_0, P_1)^{rs_0} \hat{e}(P_0, P_2)^{rs_1} \ldots \hat{e}(P_0, P_t)^{rs_{t-1}} \\ &= \hat{e}(Q_0, P_1)^{r} \hat{e}(Q_1, P_2)^{r} \ldots \hat{e}(Q_{t-1}, P_t)^{r} \\ &= g^r \hat{e}(Q_1, U_2) \ldots \hat{e}(Q_{t-1}, U_t) \end{aligned}$$

and $\frac{\hat{e}(U_0, S_t)}{\prod_{i=2}^{t} \hat{e}(Q_{i-1}, U_i)} = g^r$ for the g computed by Alice at the encryption.

The present version of the scheme is a simplified one reaching only the chosen-plaintext security level. To convert it into a chosen-ciphertext secure one, the Fujisaki-Okamoto generic transformation [89] is simply applied to it. Unlike the 2-level solution proposed by Horwitz and Lynn in 2002, the resulting scheme provably resists to a collusion between any

Root Setup: given a security parameter k, the root PKG chooses symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of prime order $q > 2^k$, a generator $P_0 \in \mathbb{G}_1$, a master key $\mathsf{mk} := s_0 \xleftarrow{R} \mathbb{Z}_q^*$ and sets $Q_0 = s_0 P_0$. It selects hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1$ and $H_2 : \mathbb{G}_2 \to \{0,1\}^n$ for some n denoting the size of plaintexts. The public parameters are

$$\mathtt{params} := (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P_0, Q_0, H_1, H_2).$$

Lower Level Setup: An entity E_t at level $Level_t$ randomly picks $s_t \xleftarrow{R} \mathbb{Z}_q^*$ and keeps it secret.

Keygen$_{t-1}$: at level $Level_t$, we consider an entity E_t of address $(\mathsf{ID}_1, \ldots, \mathsf{ID}_t)$, where $(\mathsf{ID}_1, \ldots, \mathsf{ID}_i)$, for $1 \leq i \leq t-1$, is the address of its ancestor at $Level_i$. Let S_0 be the unit element of $\mathbb{G}_1$. At $Level_{t-1}$, the father E_{t-1} of E_t generates E_t's private key as follows:

(1) It computes $P_t = H_1(\mathsf{ID}_1, \ldots, \mathsf{ID}_t) \in \mathbb{G}_1$.
(2) Let $S_t = S_{t-1} + s_{t-1} P_t = \sum_{i=1}^{t} s_{i-1} P_i$ be E_t's secret point which then depends on secret elements S_{t-1} and $s_{t-1} \in \mathbb{Z}_q$ of E_{t-1}.
(3) E_{t-1} also transmits $Q_i = s_i P_0 \in \mathbb{G}_1$ to E_t for $1 \leq i \leq t-1$ (it has computed $Q_{t-1} = s_{t-1} P_0$ itself and has received $Q_0, \ldots, Q_{t-2}$ from its ancestors).

E_t's private key is $(S_t, Q_1, \ldots, Q_t, s_t)$. The part $(S_t, Q_1, \ldots, Q_{t-1})$ is received from its father E_{t-1} and it generates the components s_t and $Q_t = s_t P_0$ itself.

Encrypt: to encrypt a message $M \in \mathcal{M}$ for an entity E_t of address $(\mathsf{ID}_1, \ldots, \mathsf{ID}_t)$,

(1) Compute $P_i = H_1(\mathsf{ID}_1, \ldots, \mathsf{ID}_i) \in \mathbb{G}_1$ for $1 \leq i \leq t$.
(2) Randomly pick $r \in \mathbb{Z}_q$ and compute the ciphertext

$$C = [rP_0, rP_2, \ldots, rP_t, M \oplus H_2(g^r)] \quad \text{with} \quad g = \hat{e}(Q_0, P_1).$$

Decrypt: E_t receives $C = [U_0, U_2, \ldots, U_t, V] \in \mathcal{C}$. To decrypt it, he/she computes

$$V \oplus H_2\left(\frac{\hat{e}(U_0, S_t)}{\prod_{i=2}^{t} \hat{e}(Q_{i-1}, U_i)}\right) = M.$$

FIGURE 2.4. The Gentry-Silverberg HIBE scheme

number of dishonest users (i.e. a set of users pooling their private information in an attempt to threaten the confidentiality of messages sent

to a honest user). It is showed in [99] how to turn the above encryption scheme into a hierarchical identity based signature and how to shorten the ciphertexts produced by the scheme (as well as the signatures produced by the derived hierarchical signature issuing protocol).

The drawback of the Gentry-Silverberg scheme is that its computational cost and the size of ciphertexts are proportional to the depth of the receiver in the hierarchy. A solution to this problem was recently found by Boneh, Boyen and Goh [38] who showed an example of an HIBE system with constant size ciphertexts and constant encryption and decryption costs.

6. Identity based encryption in the standard model

Recent works of Boneh-Boyen [36] and Waters [217] showed that it was possible to devise identity based encryption schemes that have security proofs without random oracles.

These results followed from observations of Canetti, Halevi and Katz [49] who introduced a weaker model (called 'selective-ID' model) of chosen-plaintext and chosen-ciphertext attacks for (hierarchical) IBE schemes than the model originally considered by Boneh and Franklin [40][2]. Canetti et al. [49] showed the existence of a certain form of hierarchical IBE schemes that are provably secure against chosen-plaintext and 'selective-ID' attacks (also termed 'IND-sID-CPA attacks') without random oracles. The same authors [50] subsequently showed a generic transformation that turns any 'selective-ID chosen-plaintext secure' IBE in the standard model into a traditional public key encryption scheme that is fully IND-CCA2 in the standard model. Another generic conversion ending up with shorter ciphertexts was subsequently given by Boneh and Katz [42].

Both generic transformations were further used by Boneh and Boyen [35] to convert their 'IND-sID-CPA' HIBE of ℓ levels into a provably 'selective-ID chosen-ciphertext secure' (or 'IND-sID-CCA2' secure) HIBE scheme of $\ell - 1$ levels. At Crypto'04, Boneh and Boyen put forward the

[2]Intuitively, in the 'selective-ID' attack model, the adversary has to announce at the beginning of the game (even before seeing the public parameters) the target identity under which her challenge ciphertext will be encrypted at the challenge phase. As we will see in chapter 3, the stronger model of Boneh and Franklin allows adversaries to adaptively choose the target identity at the challenge phase.

first random oracle-free construction of IBE system that is provably secure against chosen plaintext attacks (or IND-ID-CPA secure) in the strongest model of Boneh-Franklin. Their scheme can be scaled into a 2-level hierarchical construction which yields (through the techniques of Canetti et al. or Boneh-Katz) a fully secure IND-ID-CCA2 (see chapter 3 for a formal definition of this security notion) IBE in the standard model. Unfortunately, their scheme is much too inefficient to be used in practice: for recommended parameters, an encryption requires about 160 elliptic curve scalar multiplications. A more efficient construction was provided by Waters [217] at Eurocrypt'05: encrypting a message only entails a few group operations but this scheme features a very long sequence of public parameters. For the security level of a 1024-bit RSA modulus and using supersingular curves, the system-wide public key is made of 42000 bits (more than 40 times the size of an RSA modulus). This shows again that provably secure cryptosystems in the standard model tend to be more expensive than random oracle-using ones. Both the Boneh-Boyen [36] and Waters systems have security proofs under the Decisional Bilinear Diffie-Hellman (DBDH) assumption defined in chapter 1.

Part 2

New Results on Identity-Based Encryption and Related Concepts

CHAPTER 3

Variants of the Boneh-Franklin IBE

Abstract. This chapter presents two methods to improve the efficiency of the Boneh-Franklin identity-based encryption scheme without affecting its security against chosen-ciphertext attacks. The first one enables a faster decryption at the expense of using a stronger assumption to prove the security of the scheme. The second one additionally allows reducing the length of ciphertexts and requires a strongly secure symmetric encryption scheme as a building block. The latter scheme has the noticeable feature that no validity checking must be performed at decryption as all elements of the ciphertext space are valid ciphertexts. In other words the encryption mapping is surjective as for the variants of OAEP recently studied by Phan and Pointcheval.

1. Introduction

As mentioned in chapter 2, finding a practical identity-based encryption scheme (IBE) remained a long-standing open challenge until two independent works of Boneh-Franklin [40] and Cocks [66] issued in 2001. Among those two solutions, Boneh and Franklin's one happens to be the most practical one, especially from a bandwidth point of view.

In their paper, they extended the usual notions of chosen-plaintext and chosen-ciphertext security to the context of identity-based encryption through a model where adversaries may adaptively choose the identity on which they want to be challenged after having obtained private keys for other arbitrary identities. This extension of usual security models was motivated by the need to prove that any collusion of dishonest end-users does not threaten the master information stored by authorities and does not harm the secrecy of messages intended to honest users.

In the model of security against chosen-ciphertext attacks, a maximal power is granted to adversaries who additionally have access to a decryption oracle returning plaintexts or rejection messages on input of

ciphertext-identity pairs. In order to enhance the security of their construction in such a way that it provably withstands chosen-ciphertext attacks, Boneh and Franklin applied a random oracle-using generic transformation due to Fujisaki and Okamoto [89] which is well known to convert weakly secure traditional public key encryption schemes into cryptosystems that are secure against chosen-ciphertext attacks.

Since their enhanced IBE system (named FullIdent in [40]) falls into a context which is slightly different from a traditional public key setting, Boneh and Franklin had to formally establish in the random oracle model that the Fujisaki-Okamoto conversion also applies to their context. A flaw was very recently discovered by Galindo [93] in one of the steps of their security proof. He explained how to fix the problem and showed that the security result provided by the generic transformation of [89] remains correct. In the same paper [93], Galindo used another transformation due to Fujisaki and Okamoto [88] to obtain another enhancement of the Boneh-Franklin construction. He then pinpointed that his new enhanced identity-based cryptosystem features a tighter security reduction (but from a stronger assumption) than the repaired reduction for the Boneh-Franklin scheme.

The contribution of the present chapter is to extend the results of [40] and [93] by putting forward two other chosen-ciphertext secure enhancements of Boneh and Franklin's basic IBE (called BasicIdent in [40] and in the previous chapter) that avoid some overheads which are present in the constructions of [40] and [93]. Indeed, these works employ the Fujisaki-Okamoto transformations [88, 89] that both imply a significant additional computational effort for receivers w.r.t. the weakly secure primitive to which they are applied. This overhead is actually induced by the necessity of re-encrypting decrypted ciphertexts using some specific randomness in order to check their validity in chosen-ciphertext security concerns.

In our first proposal, the decryption operation is essentially as efficient as its counterpart in BasicIdent and ciphertexts have the same length as in the FullIdent scheme. The validity checking of a ciphertext is performed by simply computing a hash function and no re-encryption is required. As explained further in this chapter, other generic security enhancing transformations could be applied to BasicIdent without entailing a re-encryption but ciphertexts would be significantly longer

than those of our scheme.

Our second proposal is a hybrid construction involving a strongly secure symmetric encryption scheme. If the latter is length-preserving (that is if symmetrically encrypted ciphertexts are not longer than plaintexts), ciphertexts are as short as those of BasicIdent. In this case, the hybrid construction has the particularity that all elements of the ciphertext space are accepted as valid ciphertexts and have a corresponding plaintext. Other public key encryption schemes with the same surprising feature were previously discovered by Phan and Pointcheval [176, 177] but our construction happens to be the first example of an identity-based scheme of this kind.

The rest of this chapter is organized as follows: section 2 formally defines the hard problem on which the security of our schemes provably relies. It then recalls the definition of Boneh and Franklin's identity-based extension of the notion of chosen-ciphertext security. Section 3 recalls the specification of the Fujisaki-Okamoto transformation and the description of Boneh and Franklin's FullIdent scheme. Our two constructions are detailed in section 4: subsection 4.1 shows the first one which may be regarded as an extension of a construction originally designed by Bellare and Rogaway [29] whereas the hybrid extension of BasicIdent is described in subsection 4.2.

2. Preliminaries

2.1. Underlying hard problem

This section defines a hard problem that is a variant of the Bilinear Diffie-Hellman problem recalled in chapter 1 and on which the security of our scheme is showed to rely.

Definition 3.1 *Given symmetric bilinear map groups* $(\mathbb{G}_1, \mathbb{G}_2)$ *of prime order* $q > 2^k$ *for a security parameter* $k \in \mathbb{N}$*, the* ***Gap Bilinear Diffie-Hellman Problem*** *(Gap-BDH) in* $(\mathbb{G}_1, \mathbb{G}_2)$ *consists in, given* (P, aP, bP, cP)*, computing* $\hat{e}(P,P)^{abc}$ *with the help of a decision oracle distinguishing the distribution* $D_1 := \{(P, aP, bP, cP, \hat{e}(P,P)^{abc}) | a, b, c \stackrel{R}{\leftarrow} \mathbb{Z}_q^*\}$*, that is called distribution of "BDH tuples", from the distribution of random tuples* $D_2 := \{(P, aP, bP, cP, h) | a, b, c \stackrel{R}{\leftarrow} \mathbb{Z}_q^*,\ h \stackrel{R}{\leftarrow} \mathbb{G}_2\}$.

This problem is an intuitively natural generalization of the Gap Diffie-Hellman problem [168] recalled in chapter 1. It was used in [134]

for the first time and subsequently in [127, 128, 14]. In this chapter, we two IBE schemes which are secure in the strongest sense provided the Gap-BDH problem is hard.

2.2. Security notions for identity-based encryption

The model of adaptive chosen-ciphertext security was extended by Boneh and Franklin themselves [40] to the setting of identity-based encryption. Their model also considers a "find-then-guess" game between a challenger and an adversary who may adaptively choose the identity on which she will be challenged after having corrupted several arbitrary identities by asking a key extraction oracle for the private key associated to them. The model obviously imposes the restriction that such adversaries are disallowed to request the private key of the target identity.

Definition 3.2 *An identity-based encryption scheme (IBE) is said to be* ***adaptively chosen-ciphertext secure*** *(IND-ID-CCA2) if no probabilistic polynomial time (PPT) adversary has a non-negligible advantage in the following game.*

1. *The challenger runs the Setup algorithm on input of a security parameter k and sends the domain-wide parameters to the CCA-adversary $\mathcal{A}$.*
2. *In a find stage, $\mathcal{A}$ starts probing the following oracles:*
   - *Key extraction oracle: given an identity ID, it returns the extracted private key associated to it.*
   - *Decryption oracle: given an identity $\mathsf{ID} \in \{0,1\}^*$ and a ciphertext C, it generates the private key d_{ID} associated to ID and returns either a plaintext $M \in \mathcal{M}$ or a distinguished symbol $\perp$ indicating that the ciphertext was not correctly formed.*

   $\mathcal{A}$ can present her queries adaptively in the sense that each query may depend on the answer to previous ones. At some point, she produces two plaintexts $M_0, M_1 \in \mathcal{M}$ and a target identity ID^* for which she has not corrupted the private key in stage 2. The challenger computes $C = \mathsf{Encrypt}(M_b, \mathsf{ID}^*)$, for a random hidden bit $b \overset{R}{\leftarrow} \{0,1\}$, which is sent to $\mathcal{A}$.
3. *In the guess stage, $\mathcal{A}$ asks new queries as in the find stage but is restricted not to issue a key extraction request on the target*

identity ID^* and cannot submit C to the decryption oracle for the identity ID^*. Eventually, $\mathcal{A}$ outputs a bit b' and wins if $b' = b$.

$\mathcal{A}$'s advantage is defined as $Adv(\mathcal{A}) := 2 \times Pr[b' = b] - 1$.

The above definition formalizes the strongest notion of security for IBE systems. A strictly weaker one called *chosen-plaintext security* (or IND-ID-CPA security in this context) is formalized by an identical game except that adversaries are not equipped with a decryption oracle but are still provided with a private key extraction oracle.

3. Identity-based encryption with chosen-ciphertext security

The simplest version of the Boneh-Franklin IBE, that was called BasicIdent in [40] and in the previous chapter, does not meet the security level captured by definition 3.2. Indeed, from a given challenge ciphertext $C = \langle A, B\rangle = \langle rP, m_b \oplus h_2(\hat{e}(P_{pub}, Q_{\mathsf{ID}})^r)\rangle$ where $m_b \in \{0,1\}^n$ is the message chosen by the challenger among m_0, m_1 at step 3, a straightforward attack consists in computing an encryption $C' = \langle A, B'\rangle$ of the logical negation of m_b (by simply setting B' as the bitwise exclusive OR of B and the bitstring $\mathbf{1}^n = 11\ldots1$) and asking for the decryption of C' at some moment of the guess stage[1]. The latter attack can be prevented by appending a checksum[2] to the ciphertext or through a generic transformation such as Fujisaki and Okamoto's one [89] as done in the paper by Boneh and Franklin [40].

The Fujisaki-Okamoto conversion is the following: given a public key encryption scheme $\Pi := (\mathcal{K}, \mathcal{E}_{pk}, \mathcal{D}_{sk})$ that satisfies the very weak requirement of *one-wayness against chosen-plaintext attacks* (OW-CPA: that is the infeasibility of recovering the plaintext when observing a ciphertext), a hybrid chosen-ciphertext secure scheme $\Pi^{hy} = (\mathcal{K}, \mathcal{E}^{hy}_{pk}, \mathcal{D}^{hy}_{sk})$ is obtained through the following transformation:

$$\mathcal{E}^{hy}_{\mathsf{pk}}(m, \sigma) = \langle \mathcal{E}_{\mathsf{pk}}(\sigma, H(\sigma, m)), H'(\sigma) \oplus m\rangle$$

and

[1] Such an attack is actually a form of malleability as the decryption of C' is related to the plaintext corresponding to C in a known manner.

[2] This is basically the idea of the REACT generic conversion [167] and the construction put forward by Bellare and Rogaway [29] for trapdoor permutations.

$$\mathcal{D}^{hy}_{\mathsf{sk}}(A,B) \quad = \quad B \oplus H'(\sigma) \;\text{ if }\; \mathcal{E}_{\mathsf{pk}}(\sigma, H(\sigma, B \oplus H'(\sigma))) \\ \text{and } \perp \text{ otherwise}$$

where $\sigma = \mathcal{D}_{\mathsf{sk}}(A)$ and H, H' are random oracles of appropriate domain and range. The transformation is very powerful as it provably turns a very weak primitive into a strongly secure one. Unfortunately, it involves a computational penalty for the receiver who has to re-encrypt the result of the decryption operation (perfomed using the decryption algorithm of the weakly secure scheme) in order to check the integrity of the ciphertext.

When applied to the BasicIdent IBE, the generic transformation produces an IBE scheme called FullIdent which is provably IND-ID-CCA2 secure under the bilinear Diffie-Hellman assumption as shown in [40, 93].

Setup: given security parameters k and k_1 so that k_1 is polynomial in k, the PKG runs this algorithm to output a prime q such that $2^{k-1} < q < 2^k$, symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of order q, hash functions $h_1 : \{0,1\}^* \to \mathbb{G}_1$, $h_2 : \mathbb{G}_2 \to \{0,1\}^{k_1}$, $h_3 : \{0,1\}^{n+k_1} \to \mathbb{Z}_q^*$ and $h_4 : \{0,1\}^{k_1} \to \{0,1\}^n$. It finally picks a master key $\mathsf{mk} := s \xleftarrow{R} \mathbb{Z}_q^*$ and the public key $P_{pub} := sP \in \mathbb{G}_1$. The system-wide public key is then

$$\mathsf{params} := \{q, k, k_1, \mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, e, h_1, h_2, h_3, h_4, n\}$$

where n denotes a bound on the size of plaintexts.

Keygen: given a user's identity $\mathsf{ID} \in \{0,1\}^*$, the PKG computes $Q_{\mathsf{ID}} = h_1(\mathsf{ID}) \in \mathbb{G}_1$ and returns a private key $d_{\mathsf{ID}} = sQ_{\mathsf{ID}} \in \mathbb{G}_1$.

Encrypt: to encrypt a message M under the system-wide public key P_{pub} and an identity $\mathsf{ID} \in \{0,1\}^*$, compute $Q_{\mathsf{ID}} = h_1(\mathsf{ID}) \in \mathbb{G}_1$, pick a random $\sigma \xleftarrow{R} \{0,1\}^{k_1}$ and output the ciphertext

$$C = \left\langle rP, \sigma \oplus h_2(g^r_{\mathsf{ID}}), m \oplus h_4(\sigma) \right\rangle$$

where $g_{\mathsf{ID}} = \hat{e}(P_{pub}, Q_{\mathsf{ID}}) \in \mathbb{G}_2$ and $r = h_3(\sigma, m)$.

Decrypt: given $C = \langle U, V, W \rangle \in \mathbb{G}_1 \times \{0,1\}^{n+k_1}$, compute $\omega = \hat{e}(U, d_{\mathsf{ID}}) \in \mathbb{G}_2$, $\sigma = V \oplus h_2(\omega) \in \{0,1\}^{k_1}$ and $m = W \oplus h_4(\sigma) \in \{0,1\}^n$. The algorithm outputs $m \in \{0,1\}^n$ if $U = rP$ with $r = h_3(\sigma, m)$ and $\perp$ otherwise.

FIGURE 3.1. The FullIdent scheme

Fortunately, in the present case as well as in some particular probabilistic cryptosystems such as El Gamal's one, receivers do not need

to perform a complete re-encryption upon decryption but may simply check whether $U = rP$ with $r = h_3(\sigma, M)$. A simple additional scalar multiplication in $\mathbb{G}_1$ is thus needed w.r.t. to the decryption algorithm of BasicIdent. Nevertheless, avoiding this additional elliptic curve multiplication would be interesting as the speed of the decryption operation would be significantly increased: for example, if we believe the implementation measurements given in [171] for supersingular curves in characteristic 3 using a Pentium 4 2.4 GHz, an elliptic curve scalar multiplication in projective coordinates requires about 25% of the time to compute a Tate Pairing. Removing the final scalar multiplication in the decryption operation can offer an efficiency improvement of 20%. The relative gain is smaller for ordinary curves. However, as discussed in chapter 7, a recent work [203] tends to show that the best way to implement the Boneh-Franklin IBE is to use supersingular curves if the security is required to rely on a fairly natural assumption.

4. Avoiding the re-encryption in IBE

This section presents two methods to ensure the chosen-ciphertext security of Boneh and Franklin's system [40] without requiring a re-encryption for validity checking upon decryption and without having to encode a piece of ciphertext as a long element of $\mathbb{G}_2$. The price to pay for this efficiency improvement is the need of a stronger computational assumption in the security proof of the enhanced scheme: while the original FullIdent construction is secure under the Bilinear Diffie-Hellman assumption, the security of our constructions is proved assuming the intractability of the Gap Bilinear Diffie-Hellman problem.

We have to mention that other generic transformations such as REACT [167] or GEM [70] could be applied to BasicIdent or to some of its variants to turn them into fully secure identity-based encryption schemes without requiring the receiver to perform a re-encryption for validity checking concerns. Unfortunately, these two transformations should be applied to an OW-PCA[3] variant of BasicIdent for which a part of the

[3]More precisely, this notion would be an identity-based flavored extension of the One-Wayness against Plaintext-Checking Attacks characterizing schemes that remain computationally one-way even in the presence of an oracle deciding whether a given ciphertext encrypts a given message. See [167] for a more formal definition.

ciphertext is obtained by multiplying the message with a $\mathbb{G}_2$ element that has a representation of at least 1024 bits for recommended parameters (see [40] or [43] for details). More precisely, REACT or GEM would have to be applied to a variant of the BasicIdent scheme where plaintexts are encoded as $\mathbb{G}_2$ elements and ciphertexts have the form

$$C := \langle rP, M \cdot \hat{e}(P_{pub}, Q_{\mathsf{ID}})^r \rangle.$$

Although it is possible to compress the representation of those $\mathbb{G}_2$ elements to third of their length on supersingular curves using the techniques of Barreto and Scott [24], ciphertexts would remain significantly longer than in our schemes even if a compression technique with a ratio of 1/3 was applied in variants of the scheme padded with REACT or GEM. Our bandwidth improvements remain significant w.r.t. to implementations allowed by REACT/GEM and pairing compressions for BN curves [23] (which shorten pairing values from 1920 to 320 bits).

4.1. An extension of the Bellare-Rogaway construction

This first method introduces a minimal amount of redundancy in ciphertexts (only 160 additional bits are needed w.r.t to BasicIdent) and is actually an extension of a construction originally designed by Bellare and Rogaway [29] for trapdoor permutations. This construction produces ciphertexts of the form $E(m, r) = \langle f(r), m \oplus G(r), H(m, r) \rangle$, where $f : D \to D$ is a trapdoor permutation over some domain D, r is a random element of D and G, H are random oracles. Actually, this construction (that was previously generalized in [167] into a generic conversion from a weakly secure encryption scheme to a chosen-ciphertext secure one) can be instantiated with more general number theoretic primitives. For example, it can protect the El Gamal [96] cryptosystem from chosen-ciphertext attacks. The latter enhanced encryption scheme is then very similar to the repaired version of the Zheng-Seberry [229] cryptosystem that was proven secure in [17] under the Gap Diffie-Hellman assumption. The construction may be applied to the Boneh-Franklin identity-based encryption scheme as well. The resulting scheme is called XBR-IBE as a shorthand for eXtended Bellare-Rogaway like IBE.

Including elements U and ID among the inputs of the h_2 hash function is not mandatory but allows a more efficient reduction in the security proof detailed hereafter. The scheme remains secure if the rightmost

Setup: given security parameters k and k_1 so that k_1 is polynomial in k, the PKG runs this algorithm to output a prime q such that $2^{k-1} < q < 2^k$, chooses symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of order q, a generator $P \in \mathbb{G}_1$, hash functions $h_1 : \{0,1\}^* \to \mathbb{G}_1$, $h_2 : \{0,1\}^* \to \{0,1\}^{k_1}$ and $h_3 : \mathbb{G}_2 \to \{0,1\}^n$. It finally picks a master key $\mathsf{mk} := s \xleftarrow{R} \mathbb{Z}_q^*$ and the public key $P_{pub} := sP \in \mathbb{G}_1$. The system-wide public key is then

$$\mathsf{params} := \{q, k, k_0, \mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, e, h_1, h_2, h_3, n\}$$

where n denotes a bound on the size of plaintexts.

Keygen: given a user's identity $\mathsf{ID} \in \{0,1\}^*$, the PKG computes $Q_{\mathsf{ID}} = h_1(\mathsf{ID}) \in \mathbb{G}_1$ and returns a private key $d_{\mathsf{ID}} = sQ_{\mathsf{ID}} \in \mathbb{G}_1$.

Encrypt: to encrypt a message m under the system-wide public key P_{pub} and an identity $\mathsf{ID} \in \{0,1\}^*$, compute $Q_{\mathsf{ID}} = h_1(\mathsf{ID}) \in \mathbb{G}_1$, pick a random $r \xleftarrow{R} \mathbb{Z}_q^*$ and output the ciphertext

$$C = \left\langle rP, m \oplus h_3(g_{\mathsf{ID}}^r), h_2(m||rP||\mathsf{ID}||g_{\mathsf{ID}}^r) \right\rangle$$

where $g_{\mathsf{ID}} = \hat{e}(P_{pub}, Q_{\mathsf{ID}}) \in \mathbb{G}_2$.

Decrypt: given $C = \langle U, V, W \rangle \in \mathbb{G}_1 \times \{0,1\}^{n+k_1}$, compute $\omega = \hat{e}(U, d_{\mathsf{ID}}) \in \mathbb{G}_2$ and $m = V \oplus h_3(\omega) \in \{0,1\}^n$. The algorithm outputs $m \in \{0,1\}^n$ if $W = h_2(m||U||\mathsf{ID}||\omega)$ and $\perp$ otherwise.

FIGURE 3.2. The XBR-IBE scheme

part of the ciphertext is computed as $W = h_2(m||g_{\mathsf{ID}}^r)$ but the reduction then involves a number of calls to the decision oracle which is quadratic (instead of linear) in the number of adversarial queries.

Theorem 3.1 *Let us assume that an IND-ID-CCA2 adversary $\mathcal{A}$ has an advantage ϵ over XBR-IBE when running in a time τ, asking q_{h_i} queries to oracles h_i ($i = 1, 2, 3$), q_D decryption queries and q_{KE} key extraction queries. Then there exists a PPT algorithm $\mathcal{B}$ solving the Gap-BDH problem with an advantage*

$$Adv^{\mathsf{Gap\text{-}BDH}}(\mathcal{B}) \geq \frac{1}{e(q_{KE}+1)} \left(\epsilon - \frac{q_D}{2^{k-1}}\right)$$

within time $\tau' \leq \tau + (q_{h_1} + q_{KE})\tau_{mult} + 2(q_{h_2} + q_{h_3})\Phi$ where τ_{mult} is the cost of a scalar multiplication in $\mathbb{G}_1$, Φ denotes the cost of a call to the DBDH oracle and e is the base of the natural logarithm.

PROOF. Let $(aP, bP, cP, \mathcal{O}_{DBDH})$ be a random instance of the Gap-BDH problem where $\mathcal{O}_{DBDH}(.)$ denotes a decision oracle[4] that, given (aP, bP, cP, ω), answers 1 if $\omega = \hat{e}(P,P)^{abc}$ and 0 otherwise. We set out the description of an algorithm $\mathcal{B}$ using the latter oracle to extract $\hat{e}(P,P)^{abc}$ from its interaction with $\mathcal{A}$.

$\mathcal{B}$ initializes the adversary $\mathcal{A}$ with the system-wide public key $P_{pub} = aP$ and simulates her view as explained below. W.l.o.g., we assume that h_1 queries on identities are distinct and that any key extraction, decryption or h_2 query involving an identity is preceded by an h_1 query on the same identity.

- h_1 queries on an identity ID: $\mathcal{B}$ flips a $coin \in \{0,1\}$ taking the value 0 with probability $q_{KE}/(q_{KE}+1)$ and the value 1 with probability $1/(q_{KE}+1)$. If $coin = 0$, $\mathcal{B}$ returns $uP \in \mathbb{G}_1$ for some $u \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and answers $u(bP) \in \mathbb{G}_1$ if $coin = 1$. In both cases, a triple $(\mathsf{ID}, u, coin)$ is stored in a list L_1.

- Private key queries: when the private key associated to an identity $\mathsf{ID} \in \{0,1\}^*$ is requested, $\mathcal{B}$ recovers the triple $(\mathsf{ID}, u, coin)$ from L_1. If $coin = 1$, $\mathcal{B}$ aborts since it is unable to coherently answer the query. Otherwise, it returns uP_{pub} as a private key.

- h_2 queries: for such queries $(m_i||U_i||\mathsf{ID}_i||\omega_i)$ indexed by integers $i \in \{1, \ldots, q_{h_2}\}$, $\mathcal{B}$ does the following:

  - if oracle h_2 was already queried on the same input, the previously defined value is returned.
  - Otherwise, $\mathcal{B}$ recovers Q_{ID_i} from list L_1 and probes its decision oracle $\mathcal{O}_{DBDH}(.)$:

    - if $\mathcal{O}_{DBDH}(P, P_{pub}, U_i, Q_{\mathsf{ID}_i}, \omega_i) = 1$,
      - if $U_i = cP$ and $Q_{\mathsf{ID}_i} = h_1(\mathsf{ID}_i) = u_i(bP)$ for some known $u_i \in \mathbb{Z}_q^*$, then $\mathcal{B}$ halts and outputs $\omega_i^{1/u_i} \in \mathbb{G}_2$ as a result.
      - Otherwise, a random string $W_i \stackrel{R}{\leftarrow} \{0,1\}^{k_1}$ is sampled.

[4]It is actually a restricted decision oracle as one of its inputs never changes between all queries. The actual assumption is thus slightly weaker than the Gap-BDH one.

· $\mathcal{B}$ obtains the value $\mu_i = h_3(\omega_i) \in \{0,1\}^n$ by issuing an h_3 query on its own.

· $\mathcal{B}$ computes $C_i = \langle U_i, V_i, W_i \rangle = \langle U_i, m_i \oplus \mu_i, W_i \rangle$ and stores a record $\langle (m_i||U_i||\mathsf{ID}_i||\omega_i), W_i, V_i \rangle$ into list L_2 in order to anticipate subsequent decryption queries. The random string W_i is finally returned to $\mathcal{A}$.

- if $\mathcal{O}_{DBDH}(P, P_{pub}, U_i, Q_{\mathsf{ID}_i}, \omega_i) = 0$, $\mathcal{B}$ simply returns a random $W_i \xleftarrow{R} \{0,1\}^{k_1}$ and stores a 4-uple $\langle (m_i||U_i||\mathsf{ID}_i)||\omega_i), W_i, - \rangle$ in L_2 where the symbol $-$ means that no potential ciphertext is associated to this random oracle query.

- h_3 queries are simply responded with the previously defined value if it exists and with a new random uniformly sampled string $\mu_i \xleftarrow{R} \{0,1\}^n$ otherwise. A list L_3 is then updated to store the input of the query and its answer.

- Decryption queries: at any moment, $\mathcal{B}$ can also ask for the decryption of a ciphertext $C = \langle U, V, W \rangle \in \mathbb{G}_1 \times \{0,1\}^{n+k_1}$ for an identity ID. To simulate the behavior of the decryption oracle, $\mathcal{B}$ checks whether list L_2 contains a tuple of the form $\langle (m||U||\mathsf{ID}||\omega), W, V \rangle$ for some $m \in \{0,1\}^n$, $\omega \in \mathbb{G}_2$. If so, the corresponding $m \in \{0,1\}^n$ is returned as a plaintext. Otherwise, the ciphertext is declared invalid and a rejection message $\perp$ is returned. Clearly, in the whole simulation, the probability to wrongly reject a ciphertext is at most $q_D/2^{k_1}$. Indeed, the probability that $\mathcal{A}$ produces a ciphertext $\langle U, V, W \rangle$ and an identity ID for which $W = h_2(V \oplus h_3(\omega)||U||\mathsf{ID}||\omega)$, where ω denotes the relevant $\mathbb{G}_2$ element $\hat{e}(U, ah_1(\mathsf{ID}))$, without asking for the hash value of $V \oplus h_3(\omega)||U||\mathsf{ID}||\omega$ is at most $1/2^{k_1}$.

When $\mathcal{A}$ decides that phase 1 is over, she outputs an uncorrupted identity ID^* together with a pair of messages (m_0, m_1). At that moment, if $h_1(\mathsf{ID}^*)$ was not defined as a known multiple $u^*(bP) \in \mathbb{G}_1$ of bP, $\mathcal{B}$ fails as any subsequent interaction with $\mathcal{A}$ is useless.

Otherwise, it constructs the challenge ciphertext $C^* = \langle cP, V^*, W^* \rangle$

for randomly sampled strings $V^* \xleftarrow{R} \{0,1\}^n$ and $W \xleftarrow{R} \{0,1\}^*$. The simulation fails if the pair (C^*, ID^*) was queried to the decryption oracle at the first stage (but the probability for this to happen is smaller than $q_D/2^k$). Otherwise C^* is sent to $\mathcal{A}$ that is unable to realize that C^* is not an encryption of m_0 nor m_1 as long as she does not query h_2 on the input $(m_d||cP||\mathsf{ID}||\omega^*)$ or h_3 on ω^* for $d \in \{0,1\}$ and $\omega^* = \hat{e}(P,P)^{abcu^*}$. If such an event occurs, the simulation is not perfect anymore but it does not matter. Indeed, in this case, the relevant $\mathbb{G}_2$ element $\hat{e}(P,P)^{abcu^*}$ is made available to $\mathcal{B}$ that can then extract the Gap-BDH solution (as it knows u^*).

On the other hand, if the simulation does not fail, $\mathcal{A}$'s view is indistinguishable from a real attack environment and, in the latter case, ω^* is very likely to be submitted to oracles h_2 or h_3 at some point of the game. Indeed, let Fail denote the event that $\mathcal{B}$ fails in providing a consistent simulation of $\mathcal{A}$'s environment and, for any event E, we call $\mathrm{pr}[E]$ the conditional probability $\Pr[E|\neg\mathsf{Fail}]$. In a real game, we have $\Pr[d' = d] = (\epsilon+1)/2$ and thus $\mathrm{pr}[d' = d] = (\epsilon+1)/2$. If AskH_3 denotes the event that the hash value of ω^* is asked to h_3 and AskH_2 the event that $(m_d||cP||\mathsf{ID}||\omega^*)$ is queried to h_2, we have

$$\begin{aligned}
\mathrm{pr}[d' = d] &= \mathrm{pr}[d' = d|\mathsf{AskH}_2 \vee \mathsf{AskH}_3]\mathrm{pr}[\mathsf{AskH}_2 \vee \mathsf{AskH}_3] \\
&+ \mathrm{pr}[d' = d|\neg(\mathsf{AskH}_2 \vee \mathsf{AskH}_3)]\mathrm{pr}[\neg(\mathsf{AskH}_2 \vee \mathsf{AskH}_3)] \\
&\leq \mathrm{pr}[\mathsf{AskH}_2 \vee \mathsf{AskH}_3] + \frac{1}{2}(1 - \mathrm{pr}[\mathsf{AskH}_2 \vee \mathsf{AskH}_3])
\end{aligned}$$

as $\mathrm{pr}[d' = d|\neg(\mathsf{AskH}_2 \vee \mathsf{AskH}_3)] = 1/2$ (because if none of AskH_2 or AskH_3 occurs, $\mathcal{A}$'s view is independent of m_0 and m_1 and she cannot do better than guessing with probability $1/2$) and hence

$$\mathrm{pr}[\mathsf{AskH}_2 \vee \mathsf{AskH}_3] \geq \epsilon.$$

When coming back to non-conditional probabilities, we find

$$\Pr[(\mathsf{AskH}_2 \vee \mathsf{AskH}_3) \wedge \neg\mathsf{Fail}] \geq \epsilon \Pr[\neg\mathsf{Fail}].$$

The probability $\Pr[\neg\mathsf{Fail}]$ remains to be assessed. We know that event $\neg\mathsf{Fail}$ requires the following conditions to be simultaneously satisfied.

E_1: $\mathcal{B}$ does not abort as a result of a private key extraction query.

E_2: $\mathcal{B}$ does not abort during the challenge phase because of $\mathcal{A}$'s choice for her target identity ID^*.

E_3: $\mathcal{B}$ does not fail because the constructed challenge C^* was previously queried to the decryption oracle for the identity ID^*.

E_4: $\mathcal{B}$ does not provide an incorrect simulation because of a wrongly rejected ciphertext.

The above events are clearly independent. We already observed that $\Pr[E_4] \geq 1 - q_D/2^k$ and $\Pr[E_3] \geq 1 - q_D/2^k$. We also have $\Pr[E_1] = (1 - 1/(q_{KE} + 1))^{q_{KE}} \geq 1/e$ (as shown in the proof technique of [68]) and $\Pr[E_2] = 1/(q_{KE} + 1)$. Putting those observations together, we find that $\Pr[\neg\mathsf{Fail}] \geq e^{-1}(q_{KE} + 1)^{-1}(1 - q_D/2^k)^2$ and we finally obtain

$$\begin{aligned} \Pr[(\mathsf{AskH}_2 \vee \mathsf{AskH}_3) \wedge \neg\mathsf{Fail}] &\geq \frac{1}{e(q_{KE}+1)}\big(1 - \frac{q_D}{2^k}\big)^2\epsilon \\ &> \frac{1}{e(q_{KE}+1)}\big(\epsilon - \frac{q_D}{2^{k-1}}\big). \end{aligned}$$

When the adversary halts (it is reasonable to expect that she does not enter an infinite loop if her environment looks consistent) and produces a result, the latter is ignored: if $\mathcal{B}$ did not obtain the searched Gap-BDH solution $\hat{e}(P,P)^{abc}$ when handling h_2 queries at some moment of the simulation, it can expect to find the relevant element $\hat{e}(P,P)^{abcu^*}$ among the inputs of h_3 queries made by $\mathcal{A}$. For all entries $\langle \omega_i, h_{3,i} \rangle$ $(i = 1, \ldots, q_{h_3})$ contained in L_3, it checks whether

$$\mathcal{O}_{DBDH}(P, aP, u^*(bP), cP, \omega_i) = 1$$

and outputs ω_i^{1/u^*} if the latter condition holds for some $i \in \{1, \ldots, q_{h_3}\}$. □

The above reduction is more efficient than the one obtained from the BDH assumption through the Fujisaki-Okamoto transform [89] in the original IBE. Although our proof relies on a stronger assumption, we believe that this fact has a certain theoretical interest because, as argued in [121], a tight reduction from a given assumption might be preferable to a loose reduction from a potentially weaker assumption. On the other hand, the Gap-BDH assumption does not appear as a much stronger assumption than the (already non-standard) BDH assumption.

Interestingly, if we compare our security reduction for Hybrid-IBE with the one of Galindo [94, 93] for another variant of the Boneh-Franklin

IBE obtained through the first Fujisaki-Okamoto transform [88], we find that ours is as efficient as Galindo's one (which relies on the DBDH assumption) but our construction happens to be more efficient as no re-encryption is needed for the receiver.

4.2. A hybrid identity-based encryption scheme

In provable security purposes, motivated by the design of public key encryption schemes that can be shown to reach the widely admitted required level of security against adaptive chosen-ciphertext attacks [185] in the random oracle model [29], Bellare and Rogaway introduced the notion of plaintext-awareness [30]. This notion captures the general idea of rendering a decryption oracle useless by making impossible the creation of valid ciphertexts by the adversary. In very recent works [176, 177], Phan and Pointcheval gave evidence that chosen-ciphertext security is achievable without plaintext-awareness and showed designs of public key encryption schemes that are secure in the strongest sense [185] in the random oracle model although all ciphertexts are valid and have a corresponding plaintext.

Meanwhile, Kurosawa and Matsuo [126] showed how to turn the DHIES [2] hybrid construction into a redundancy-free encryption system in the standard model (but under the strong and non-standard oracle Diffie-Hellman assumption that actually looks as strong as the random oracle model) by removing the message authentication code (MAC) from it and replacing the IND-CPA symmetric encryption scheme by an IND-CCA one. Their approach is actually a combination of a key encapsulation mechanism (KEM)[5] [74, 75] with a symmetric encryption scheme that was also proven secure in the random oracle model by Cramer and Shoup [74] under a more standard assumption.

In this section, we extend the technique of Kurosawa and Matsuo to the identity-based setting in the random oracle model and show a hybrid

[5]This primitive was introduced by Shoup [198, 199] and can be thought of as a public key encryption scheme that takes no plaintext as input but rather produces an encapsulation of a randomly chosen symmetric key which is intended to be used as a symmetric encryption key by a suitable data encapsulation mechanism (DEM). The combination of a KEM with a DEM yields a traditional public key encryption scheme.

variant of BasicIdent that reaches the IND-ID-CCA2 security level without introducing redundancies in ciphertexts that are thus shorter than in FullIdent and XBR-IBE. As in the latter, no re-encryption is required for receivers and the decryption operation is thus also more efficient than in FullIdent.

In the forthcoming paragraphs, we first formally describe the level of security that is required for the symmetric part of our hybrid construction and we recall the description of the hybrid Hashed El Gamal encryption scheme that is obtained by combining Cramer and Shoup's Hashed El Gamal KEM with a super pseudo-random permutation. We then set out our hybrid IBE and we provide a detailed security proof in the random oracle model.

4.2.1. Required security notion for ciphers

As the modification of DHIES presented in [126], our modification of the Boneh-Franklin IBE [40] makes use of a symmetric cipher (i.e. a deterministic length-preserving symmetric encryption scheme) that is secure against chosen-ciphertext attacks instead of one that is only secure against passive attacks (as required by the Fujisaki-Okamoto transform [89] which just uses a "one-time-pad"[6] in the original FullIdent scheme).

Recall that a symmetric encryption scheme is a triple of algorithms $SE = (K, E, D)$. The key generation algorithm K generates a key $k \stackrel{R}{\leftarrow} \{0,1\}^\lambda$ for a security parameter λ. The encryption algorithm E takes a key k and a plaintext m to produce a ciphertext $c = E(k, m)$. The decryption algorithm takes a key k and a ciphertext c to return $m/reject = D(k, c)$. In the definition of chosen-ciphertext security for symmetric encryption schemes, the adversary can query a decryption oracle $D(k, .)$ as well as an encryption oracle $E(k, .)$. We recall below a security notion for ciphers that is considered in [178] and [126].

Definition 3.3 *A symmetric cipher (E, D) is secure in the IND-CCA sense if no PPT adversary $\mathcal{A}$ has a non negligible advantage in the following game:*

1. *The challenger chooses a key $k \stackrel{R}{\leftarrow} \{0,1\}^\lambda$.*

[6] A "one-time-pad" is a bitwise exclusive OR of the message with a secret key of identical length.

2. *$\mathcal{A}$ queries the encryption oracle $E(k,.)$ and the decryption oracle $D(k,.)$. She then outputs messages (m_0, m_1) that were not submitted to $E(k,.)$ (which is deterministic) and receives a $c^* = E(k, m_b)$ for $b \stackrel{R}{\leftarrow} \{0,1\}$.*
3. *$\mathcal{A}$ issues new queries as in step 2 but is disallowed to ask for the decryption of c^* and the encryptions of m_0 and m_1.*
4. *$\mathcal{A}$ eventually outputs a guess b' for b. Her advantage is*

$$Adv^{\mathsf{sym}}(\mathcal{A}) := 2 \times Pr[b' = b] - 1.$$

This notion of indistinguishability can also be defined at lower levels of passive adversaries who have no oracle access at all or to chosen-plaintext attackers (also termed IND-CPA attackers) who have only access to encryption oracles.

The CMC [107] and EME [108] modes of operations are both length preserving and they were shown to be secure in the sense of IND-CCA assuming that the underlying block cipher is a strong pseudo-random permutation (AES could be used for instance).

4.2.2. The Hashed El Gamal cryptosystem

Keygen: given security parameters k and λ such that λ is polynomial in k, this algorithm chooses a cyclic group $\mathbb{G}$ of prime order $q > 2^k$ and a generator $g \in \mathbb{G}$. It also selects a private key $x \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and computes $y = g^x \in \mathbb{G}$. The public key contains y, a hash function $H : \mathbb{G} \times \mathbb{G} \rightarrow \{0,1\}^\lambda$ and the description of a symmetric encryption scheme (E, D) of keylength λ.

Encrypt: to encrypt a message m, pick a random $r \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and output the ciphertext

$$C = \langle g^r, E_{SK}(M) \rangle$$

where $SK = H(g^r, y^r) \in \{0,1\}^\lambda$

Decrypt: given $C = \langle A, B \rangle \in \mathbb{G}_1 \times \{0,1\}^n$, the recipient returns $M = D_{SK}(B)$ where $SK = H(A, A^x)$.

FIGURE 3.3. The Hashed El Gamal hybrid encryption scheme

In [74], Cramer and Shoup established the security of the above variant of the El Gamal [96] cryptosystem in the random oracle model under the Gap Diffie-Hellman assumption assuming that the symmetric

encryption scheme is itself secure against chosen-ciphertext attacks.

They explained that the scheme remains secure if the symmetric key is derived from a hash value of the sole group element y^r but a more efficient reduction is obtained by deriving the key from the pair (g^r, y^r). Indeed, in the security proof the number of calls to the decision Diffie-Hellman oracle is linear (instead of quadratic) in the number of adversarial random oracle queries.

If the Hashed El Gamal KEM is combined with a symmetric encryption scheme without redundancy such as Desai's scheme [76] or any super pseudorandom permutation [107, 108], it directly yields a public key encryption scheme where all elements of the ciphertext space are accepted as valid ciphertexts and have a corresponding plaintext. The next subsection presents a straightforward extension of Cramer and Shoup's result to the Boneh-Franklin IBE.

4.2.3. A hybrid variant of the Boneh-Franklin IBE

Setup: given security parameters k and λ so that λ is polynomial in k, this algorithm chooses a k-bit prime number q, symmetric bilinear map groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of order q, a generator $P \in \mathbb{G}_1$, hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1$, $H_2 : {\mathbb{G}_1}^2 \times \mathbb{G}_2 \to \{0,1\}^\lambda$, as well as a chosen-ciphertext secure cipher (E, D) of keylength λ. It finally picks a master key $\mathsf{mk} := s \xleftarrow{R} \mathbb{Z}_q^*$ and the corresponding public key $P_{pub} := sP \in \mathbb{G}_1$. The system-wide public key is

$$\mathsf{params} := \{q, \mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, e, H_1, H_2, G, n, E, D, \lambda, l\}$$

where n denotes a bound on the size of plaintexts.

Keygen: given a user's identity $\mathsf{ID} \in \{0,1\}^*$, the PKG computes $Q_{\mathsf{ID}} = h_1(\mathsf{ID}) \in \mathbb{G}_1$ and returns a private key $d_{\mathsf{ID}} = sQ_{\mathsf{ID}} \in \mathbb{G}_1$.

Encrypt: to encrypt a message M under the system-wide public key P_{pub} and an identity $\mathsf{ID} \in \{0,1\}^*$, compute $Q_{\mathsf{ID}} = h_1(\mathsf{ID}) \in \mathbb{G}_1$, pick a random $r \xleftarrow{R} \mathbb{Z}_q^*$ and output the ciphertext

$$C = \langle rP, E_{SK}(M) \rangle$$

where $SK = h_2(Q_{\mathsf{ID}}, rP, \hat{e}(P_{pub}, Q_{\mathsf{ID}})^r) \in \{0,1\}^\lambda$

Decrypt: given $C = \langle A, B \rangle \in \mathbb{G}_1 \times \{0,1\}^n$, the recipient returns $M = D_{SK}(B)$ where $SK = h_2(Q_{\mathsf{ID}}, A, \hat{e}(A, d_{\mathsf{ID}}))$.

FIGURE 3.4. Our Hybrid-IBE scheme

This subsection presents another secure modification of the Boneh-Franklin IBE that is (almost) as efficient as BasicIdent. On one hand the new scheme, that we call Hybrid-IBE, produces shorter ciphertexts than FullIdent and than XBR-IBE while, as in the latter, the receiver does not have to compute a scalar multiplication in $\mathbb{G}_1$ upon decryption.

We mention that redundancy-free IBE schemes may also be obtained with the OAEP 3-round generic construction [177] but the security could only be proved in a relaxation of the security model of definition 2 and ciphertexts would also be longer than those of Hybrid-IBE. The security of the latter is shown by the theorem below for which the proof uses a similar technique to [74].

Theorem 3.2 *Let us assume that an IND-ID-CCA2 adversary $\mathcal{A}$ has an advantage ϵ over Hybrid-IBE when running in time τ, asking q_{h_i} queries to oracles h_i ($i = 1, 2$), q_D decryption queries and q_{KE} key extraction queries. Then, for any $0 \leq \nu \leq \epsilon$, there either exists*

- *a PPT algorithm $\mathcal{B}$ to solve the Gap-BDH problem with an advantage*

$$Adv^{\mathsf{Gap\text{-}BDH}}(\mathcal{B}) \geq \frac{1}{e(q_{KE}+1)}\left(\epsilon - \frac{q_D}{2^k} - \nu\right)$$

  within time $\tau' \leq \tau + (q_{h_1} + q_{KE})\tau_{mult} + q_D\tau_{sym} + q_{h_2}\Phi$.
- *an attacker that breaks the IND-CCA security of the symmetric encryption scheme (E, D) with advantage ν within a time τ'*

where τ_{mult} is the cost of a multiplication in $\mathbb{G}_1$, τ_{sym} denotes the complexity of a symmetric decryption, Φ stands for the cost of a call to the decision oracle and e is the base of the natural logarithm.

Proof. Let $(aP, bP, cP, \mathcal{O}_{DBDH})$ be an instance of the Gap-BDH problem. As in the proof of theorem 3.1, $\mathcal{O}_{DBDH}(.)$ is a decision oracle that, on input (aP, bP, cP, ω), answers 1 if $\omega = \hat{e}(P, P)^{abc}$ and 0 otherwise. We describe an algorithm $\mathcal{B}$ using $\mathcal{A}$ and $\mathcal{O}_{DBDH}(.)$ to compute $\hat{e}(P, P)^{abc}$.

Algorithm $\mathcal{B}$ initializes $\mathcal{A}$ with the system-wide public key $P_{pub} = aP$ and simulates her view as explained below. W.l.o.g., we assume that H_1-queries are distinct (otherwise, a list may be used to store inputs and responses) and that any key extraction, decryption or H_2 query involving an identity comes after a H_1-query on the same identity.

- H_1 queries: for such a query on an identity ID, $\mathcal{B}$ flips a bit $coin \in \{0,1\}$ taking the value 0 with probability ξ and the value 1 with probability $1-\xi$. If $coin = 0$, $\mathcal{B}$ returns $uP \in \mathbb{G}_1$ for some $u \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and it answers $u(bP) \in \mathbb{G}_1$ if $coin = 1$. In both cases, a triple $(\mathsf{ID}, u, coin)$ is stored in a list L_1.
- Private key queries: when the private key associated to an identity $\mathsf{ID} \in \{0,1\}^*$ is requested, $\mathcal{B}$ recovers the triple $(\mathsf{ID}, u, coin)$ from L_1. If $coin = 1$, $\mathcal{B}$ aborts since it is unable to coherently answer the query. Otherwise, it returns uP_{pub} as a private key.
- Queries to $H_2(.)$: according to a proof technique already used in [74, 199, 75] for KEMs, these queries are processed using three lists $L_{2,a}$, $L_{2,b}$ and $L_{2,c}$ which are initially empty:
  - $L_{2,a}$ contains triples $(Q_{\mathsf{ID}_i}, A_i, \omega_i)$ to which a hash value was previously assigned and the corresponding digest $h_{2,i} \in \{0,1\}^\lambda$.
  - $L_{2,b}$ contains triples $(Q_{\mathsf{ID}_i}, A_i, \omega_i)$ such that a quadruple $(Q_{\mathsf{ID}_i}, A_i, \omega_i, h_{2,i})$ exists in $L_{2,a}$ for some $h_{2,i} \in_R \{0,1\}^\lambda$ and $\mathcal{O}_{DBDH}(P, Q_{\mathsf{ID}_i}, A_i, P_{pub}, \omega_i) = 1$.
  - $L_{2,c}$ will contain triples $(Q_{\mathsf{ID}_i}, A_i, h_{2,i})$ for which $\mathcal{B}$ has implicitly assigned a value $h_{2,i} \stackrel{R}{\leftarrow} \{0,1\}^\lambda$ to $H_2(Q_{\mathsf{ID}_i}, A_i, \omega_i)$ although it does not know the unique element $\omega_i \in \mathbb{G}_2$ for which $\mathcal{O}_{DBDH}(P, Q_{\mathsf{ID}_i}, A_i, P_{pub}, \omega_i) = 1$.

  More precisely, when $\mathcal{A}$ submits a triple $(Q_{\mathsf{ID}}, A, \omega)$ to $H_2(.)$,
  - $\mathcal{B}$ first checks if $L_{2,a}$ contains a tuple $(Q_{\mathsf{ID}}, A, \omega, h_2)$ for some $h_2 \in \{0,1\}^\lambda$ (meaning the a hash value was previously assigned to the same input). If it does, h_2 is returned.
  - Otherwise, $\mathcal{B}$ submits $(P, Q_{\mathsf{ID}}, A, P_{pub}, \omega)$ to the $\mathcal{O}_{DBDH}(.)$ oracle which decides whether it is a valid BDH tuple.
    - If it is, then:
      - If $A = cP$ and $coin = 1$ (i.e. $H_1(\mathsf{ID})$ was defined to be $u(bP)$), $\mathcal{B}$ halts and outputs $\omega^{1/u}$ which is the searched solution. We denote by AskH_2 the event that such a hash query is made .
      - Otherwise, $\mathcal{B}$ continues and adds $(Q_{\mathsf{ID}}, A, \omega)$ in $L_{2,b}$.

- If $L_{2,c}$ contains a triple $(Q_{\mathsf{ID}}, A, h_2)$ for some $h_2 \in \{0,1\}^\lambda$, the tuple $(Q_{\mathsf{ID}}, A, \omega, h_2)$ is stored in $L_{2,a}$ and h_2 is returned to $\mathcal{A}$. Otherwise, $\mathcal{B}$ continues.
- It selects a string $h_2 \xleftarrow{R} \{0,1\}^\lambda$, inserts the tuple $(Q_{\mathsf{ID}}, A, \omega, h_2)$ into $L_{2,a}$ and answers h_2 to $\mathcal{A}$.

- Decryption queries: upon receiving a ciphertext $C = \langle A, B\rangle \in \mathbb{G}_1 \times \{0,1\}^n$ and an identity ID, the simulator $\mathcal{B}$ does the following:
  - it checks if $(Q_{\mathsf{ID}}, A, \omega)$ exists in $L_{2,b}$ for some $\omega \in \mathbb{G}_2$. If it does, $\mathcal{B}$ retrieves the tuple $(Q_{\mathsf{ID}}, A, \omega, h_2)$ from $L_{2,a}$ and returns the symmetric decryption $D_{h_2}(B)$ of B using $h_2 \in \{0,1\}^\lambda$ as a symmetric key. Otherwise, it continues.
  - It tests whether $L_{2,c}$ contains a triple $(Q_{\mathsf{ID}}, A, h_2)$ for some string $h_2 \in \{0,1\}^\lambda$. In this case, the latter is used to compute a symmetric decryption $D_{h_2}(B)$ that is returned as a result. Otherwise, a random $h_2 \xleftarrow{R} \{0,1\}^\lambda$ is chosen and $(Q_{\mathsf{ID}}, A, h_2)$ is inserted into $L_{2,c}$ ($\mathcal{B}$ thereby implicitly assigns the hash value h_2 to the oracle H_2 on the unique input $(Q_{\mathsf{ID}}, A, \omega)$ for which $\mathcal{O}_{DBDH}(P, Q_{\mathsf{ID}}, A, P_{pub}, \omega) = 1$ although the relevant $\omega \in \mathbb{G}_2$ is still unknown) while $D_{h_2}(B)$ is returned to $\mathcal{A}$.

After the find stage, $\mathcal{A}$ comes with messages $M_0, M_1 \in \{0,1\}^n$ and a target identity ID^*. Let $(\mathsf{ID}^*, u^*, coin^*)$ be the corresponding triple in L_1. If $coin^* = 0$, $\mathcal{B}$ aborts and reports "failure" because, in such a situation, $\mathcal{A}$ is of no help in $\mathcal{B}$'s endeavour. Otherwise, it sets $A^* = cP \in \mathbb{G}_1$, checks whether $L_{2,c}$ contains a triple $(Q_{\mathsf{ID}^*}, A^*, h_2^*)$ for $Q_{\mathsf{ID}^*} = h_1(\mathsf{ID}^*)$ and some $h_2^* \in \{0,1\}^\lambda$ (if not, $\mathcal{B}$ inserts it for a string $h_2 \xleftarrow{R} \{0,1\}^\lambda$ of its choice) to compute a symmetric encryption $B^* = E_{h_2^*}(M_d)$, for $d \xleftarrow{R} \{0,1\}$, and return the challenge $C^* = \langle A^*, B^*\rangle$. In the unlikely event (its probability is less than $q_D/2^k$) that C^* was previously submitted to the decryption oracle for the identity ID^*, $\mathcal{B}$ aborts.

At the second stage, $\mathcal{B}$ processes all queries as above and $\mathcal{A}$ eventually produces a bit d'. In a real game, we have $\Pr[d' = d] = (\epsilon + 1)/2$ and, provided the simulation is perfect, the latter equality still holds as $\mathcal{A}$'s view is indistinguishable from a real environment. It can be

showed that the simulation is imperfect with a probability smaller than $e^{-1}(q_{KE}+1)^{-1}(1-q_D/2^k)$. Indeed, let us define the following events:

- E_1: $\mathcal{B}$ does not abort as a result of a private key extraction query.
- E_2: $\mathcal{B}$ does not abort during the challenge phase because $\mathcal{A}$ chooses a target identity ID^* for which $coin^* = 0$.
- E_3: $\mathcal{B}$ does not fail because the constructed challenge C^* was previously queried to the decryption oracle for the identity ID^*.

Those events are independent. We observed that $\Pr[E_3] \geq 1 - q_D/2^k$. We also have $\Pr[E_1] = (1 - 1/(q_{KE}+1))^{q_{KE}} \geq 1/e$ (as shown in the proof technique of [68]) and $\Pr[E_2] = 1/(q_{KE}+1)$. It comes that if $\mathsf{Fail} = \neg E_1 \vee \neg E_2 \vee \neg E_3$, we have $\Pr[\neg\mathsf{Fail}] = e^{-1}(q_{KE}+1)^{-1}(1-q_D/2^k)$.

On the other hand, if AskH_2 does not occur and thus if $\mathcal{A}$ never makes the relevant $h_2(Q_{\mathsf{ID}^*}, A^*, \omega^*)$ query during the game, the only way for her to produce a correct guess for d is to succeed in a chosen-ciphertext attack against the symmetric cipher (E, D): indeed, in the latter case, each decryption query on a ciphertext $C' = (A^*, B)$, with $B \neq B^*$, for the target identity ID^* corresponds to a symmetric decryption request for a completely random key SK^*. It follows that, if (E, D) is a chosen-ciphertext secure symmetric encryption scheme, the event AskH_2 is very likely to happen and $\mathcal{B}$ is able to extract the Gap-BDH solution.

More formally, for any event E, if we denote by $\mathrm{pr}[E]$ the conditional probability $\Pr[E|\neg\mathsf{Fail}]$, we have

$$\begin{aligned}\mathrm{pr}[d'=d] &= \mathrm{pr}[d'=d|\mathsf{AskH}_2]\mathrm{pr}[\mathsf{AskH}_2] + \mathrm{pr}[d'=d|\neg\mathsf{AskH}_2]\mathrm{pr}[\neg\mathsf{AskH}_2] \\ &\leq \mathrm{pr}[\mathsf{AskH}_2] + \mathrm{pr}[d'=d|\neg\mathsf{AskH}_2](1-\mathrm{pr}[\mathsf{AskH}_2])\end{aligned}$$

and, since $\mathrm{pr}[d'=d] = (\epsilon+1)/2$ and $\mathrm{pr}[d'=d|\neg\mathsf{AskH}_2] \leq (\nu+1)/2$, it comes that

$$\frac{\epsilon+1}{2} \leq \frac{\nu+1}{2} + \frac{1-\nu}{2}\mathrm{pr}[\mathsf{AskH}_2] \leq \frac{\nu+1}{2} + \frac{1}{2}\mathrm{pr}[\mathsf{AskH}_2]$$

and hence $\mathrm{pr}[\mathsf{AskH}_2] \geq \epsilon - \nu$. When going back to non-conditional probabilities, we find the announced lower bound

$$\begin{aligned}\Pr[\mathsf{AskH}_2 \wedge \neg\mathsf{Fail}] &\geq \frac{1}{e(q_{KE}+1)}\big(1 - q_D 2^{-k}\big)\big(\epsilon - \nu\big) \\ &> \frac{1}{e(q_{KE}+1)}\big(\epsilon - \frac{q_D}{2^k} - \nu\big)\end{aligned}$$

on $\mathcal{B}$'s probability of success. □

We observe that the above reduction is exactly as efficient as XBR-IBE's one and it additionally produces shorter ciphertexts thanks to the absence of redundancy.

As for Galindo's IBE [94, 93], an essentially optimal reduction can be obtained for Hybrid-IBE by applying a trick suggested in [121] at the expense of an additional pairing computation at encryption. A similar hybrid technique can be applied to a 1-level certificate-based [98] encryption scheme as well as to a variant of the new certificateless encryption scheme [6] recently proposed in [7].

5. Conclusion

This chapter presented two simple methods to increase the efficiency of Boneh and Franklin's famous identity-based encryption scheme. The first method is an extension of Bellare and Rogaway's construction [29] for trapdoor permutations. It only offers a computational gain at decryption. The second method which is a hybrid construction additionally provides shorter ciphertexts but requires using a symmetric scheme that is secure against chosen-ciphertext attacks. If the latter is a cipher (that is a deterministic and length-preserving encryption algorithm), the hybrid construction yields a first example of secure identity-based cryptosystem without redundancy in the sense of Phan and Pointcheval [176, 177]: all elements of the ciphertext space correspond to some plaintext.

CHAPTER 4

Generic constructions of certificateless encryption in the strongest model and an efficient scheme

Abstract. Certificateless cryptography (CL-PKC) is a concept that aims at enjoying the advantages of identity based cryptography without suffering from its inherent key escrow. Several methods were recently suggested to generically construct a certificateless encryption (CLE) scheme by combining identity based schemes with ordinary public key cryptosystems. Whilst the security of a sequential composition of this kind was proved in a weakened security model, we show that all these constructions are insecure against chosen-ciphertext attackers in the strongest model of security given by Al-Riyami and Paterson. We show how to easily fix these problems and give a method to achieve generic CLE constructions which are provably CCA-secure in the random oracle model. We finally propose a new efficient pairing-based scheme that performs better than previous proposals without pre-computation. We also prove its security in the random oracle model.

1. Certificateless cryptography

In 2003, Al-Riyami and Paterson [6] invented a paradigm called certificateless public key cryptography (CL-PKC) which is intermediate between identity-based [195, 40] and traditional PKI-supported cryptography. The concept was introduced to suppress the inherent key-escrow property of identity-based cryptosystems (ID-PKC) without losing their most attractive advantage which is the absence of digital certificates and their important management overhead.

Independently of [6] and a little bit earlier, Gentry [98] introduced a different but related concept named certificate based encryption (CBE) for which a signature analogue was studied in [120]. This approach is

closer to the context of a traditional PKI model as it involves a certification authority (CA) providing an efficient implicit certification service for clients' public keys.

Although very different at first glance, the CBE and CL-PKC concepts were first argued [6] to be closely related and both constructions of [6, 98] use the properties of pairings. A subsequent work of Yum and Lee considered the relations between identity-based (IBE), certificate based (CBE) and certificateless encryption schemes (CLE) and established a result of essential equivalence [224] between the three primitives but this result does not hold for the strongest security model developed in [6] for CLE schemes. The same authors also proposed generic constructions of certificateless signatures [223] and encryption schemes [222] but only established the security of their designs in security models that are seemingly weakened w.r.t. the original model considered in [6] for the public key encryption case.

A more recent work [7] thoroughly investigated the connections between the CLE and CBE paradigms by proposing a simplified definition and a revised security model for certificate based encryption before proving that any secure certificateless encryption (CLE) scheme can be turned into a secure CBE in the amended model. The authors of [7] also identified some potential weaknesses remaining in the amended model and argued that a purely generic conversion from secure CBE schemes in the latter model to some secure CLE counterpart in the appropriate CL-PKC model is very unlikely to exist.

Among other related recent results, we mention a paper [59] describing a quite similar scheme to [7], a flawed construction of authenticated encryption [130] and another work that investigates identity-based and certificateless extensions of key encapsulation mechanisms [32]. A very recent paper by Baek et al. [13] also showed how to devise a certificateless encryption scheme without pairings. The latter construction enjoys a better efficiency than pairing-based proposals [6, 7, 59] but is supported by a weaker security model and prevents users from generating their public key independently from the system's authority.

The contribution of the present chapter to the area of certificateless cryptography is two-fold. It first identifies some weaknesses in generic constructions independently considered in [5] and [222]. It shows that one of these flaws is also present in the second provably secure CLE

scheme of Al-Riyami and Paterson [7] where it can be very easily fixed. The chapter then explains how to obtain generic constructions which are provably secure in the random oracle model. It does so by first giving a generic random oracle-using conversion to turn any CLE scheme which is only secure against chosen-plaintext attacks into an IND-CCA scheme in the full model of Al-Riyami and Paterson.

The second contribution of the chapter is to describe a new efficient pairing-based scheme yielding some advantages over previous constructions [6, 7, 59, 32]: its encryption operation does not require to compute any pairing (only the decryption algorithm does) and is thus significantly faster than in previous proposals [6, 7, 59, 32]. The security proof of the new scheme is nevertheless obtained under a stronger computational assumption than for previous schemes in the literature.

In the forthcoming sections of this chapter, we first review the formal definition and adversarial model of CLE schemes in section 2 where the original construction of Al-Riyami and Paterson [6] is also recalled. Section 3 illustrates the power of their security model by showing how generic constructions studied so far are insecure in it. We explain in section 4 how to repair them and we prove the security of the fixed constructions in the random oracle model. Our new certificateless cryptosystem is then depicted in section 5 where security proofs in the random oracle model are detailed.

2. Formal models and examples

We now recall the components of a certificateless encryption scheme before detailing the relevant formal security model [6].

2.1. Definition of certificateless encryption (CLE) schemes

Definition 4.1 *A certificateless encryption scheme (CLE) is a 7-uple of algorithms which are the following:*

Setup: *is a probabilistic algorithm run by a Key Generation Center (KGC), that, given a security parameter k, returns a randomly chosen master key* mk *and a list of public parameters* params.

Partial-Private-Key-Extract: *is a possibly probabilistic algorithm, run by the KGC, that takes as input a user's identifier* ID_A *and the master key* mk *to return his/her partial private key* d_A.

Set-Secret-Value: *is a probabilistic algorithm that, given a list of public parameters* params*, returns a randomly chosen secret value* x_A *for that user. This algorithm and the next two ones are performed by the user himself.*

Set-Private-Key: *is a deterministic private key generation algorithm that, given public parameters* params*, an user's partial private key* d_A *and secret value* x_A*, outputs a private key* S_A.

Set-Public-Key: *is a deterministic public key generation algorithm that, given public parameters* params *and a user's secret value* x_A*, computes his/her public key* pk_A*. Given* params*, it must be publicly verifiable that* pk_A *is well-formed*[1].

Encrypt: *is a probabilistic algorithm taking as input a plaintext* m*, parameters* params*, a receiver's identity* ID_A *and his public key* pk_A *to produce a ciphertext* $C = \mathsf{Encrypt}(m, \mathsf{params}, \mathsf{ID}_A, \mathsf{pk}_A)$.

Decrypt: *is a deterministic algorithm that, given a ciphertext* C*, a list of public paramaters* params *and user* ID_A*'s private key, outputs a plaintext* m *or a distinguished symbol* $\perp$.

For consistency, it is obviously required that $\mathsf{Decrypt}(C, \mathsf{params}, S_A) = m$ *whenever* $C = \mathsf{Encrypt}(m, \mathsf{params}, \mathsf{ID}_A, \mathsf{pk}_A)$ *for all messages* $m \in \mathcal{M}$ *and public keys* $\mathsf{pk}_A = \mathsf{Set\text{-}Public\text{-}Key}(\mathsf{params}, x_A)$ *for which the private key is* $S_A = \mathsf{Set\text{-}Private\text{-}Key}(\mathsf{params}, \mathsf{Partial\text{-}Private\text{-}Key\text{-}Extract}(\mathsf{ID}_A), x_A)$ *and the secret value is* $x_A = \mathsf{Set\text{-}Secret\text{-}Value}(\mathsf{params})$.

Unlike Setup and Partial-Keygen that are run by a Key Generation Center (KGC), algorithms Set-Secret-Value, Set-Pr-Key and Set-Pub-Key are executed by the user. The latter's private key is thus not computable by the KGC and key escrow is not inherent.

The recent pairing-free scheme of Baek et al. [13] fits a slightly different model where users have to obtain their partial private keys before generating their public key (actually, a part of the latter is computed by the KGC and included in the partial private key). This approach is

[1]Hereby, we mean that public keys are usually required to have a special structure. As we shall see in concrete examples, they might be elements of a particular group or they might be made of a pair of group elements satisfying a special relation.

closer to the "self-certified" paradigm [101] which is another approach suggested by Girault in 1991 to use public key cryptography without traditional digital certificates and without involving an escrow authority. As explained in [6], the self-certified schemes presented in [101] are structurally somewhat similar to certificateless schemes that have been studied so far [6, 7, 59, 27]. In a self-certified scheme, an entity chooses its own private key sk and the corresponding public key pk and delivers pk to a trusted authority (TA). The TA combines pk with the identity ID of that entity to produce a witness w. This witness may just be the TA's signature on a combination of pk and ID as in [101]. Given w, ID and the TA's public key, anyone can retrieve pk, whereas only the TA can extract the witness w from pk and ID. The scheme thus provide implicit certification in the sense that the receiver's public key is incorporated in the witness w and can be retrieved from it using the TA's public key and the receiver's identity: in some sense, the witness can be viewed as a light-weight certificate that *contains* the receiver's public key.

Self-certified schemes have an advantage over certificateless ones in that the communication between an entity and the TA does not need to be confidential: there are no partial private keys to be transported to entities. Moreover, the self-certified paradigm requires investing fewer trust in authorities than within a certificateless system. But, as we will see, all certificateles schemes can be modified to allow the detection of dishonest authorities. They then reach the level 3 in the hierarchy of [101]. On the other hand, except a particular scheme named "implicit certification scheme" that was proven secure by Brown et al. [46], self-certified schemes that have been studied so far do not enjoy security proofs in formal models.

2.2. Security model

In [6], two kinds of adversaries are distinguished against CLE schemes. A Type I adversary is not given the KGC's master key but is enabled to replace public keys of arbitrary identities with other public keys of her choice (provided those public keys have the correct form). Considering such an adversarial behavior seems very natural as, in the absence of digital certificates, anyone can alter public directories by replacing end-users' public keys with other ones without being caught nor even detected. Similarly to attackers against identity-based cryptosystems,

Type I adversaries can also ask for partial and full private keys of arbitrary identities.

In contrast, a Type II adversary knows the KGC's master key (and thus does not need a partial key exposure oracle) and may still obtain full private keys for arbitrary identities but is disallowed to replace public keys during the game.

For both types of adversary, depending on the attack that is considered (chosen-plaintext or chosen-ciphertext attacks), we may provide them or not with an additional oracle decrypting adversarially chosen ciphertexts for the private key associated to arbitrary identities.

In the chosen-ciphertext scenario, the authors of [6] consider decryption oracles that should be able (thanks to suitable knowledge extractors) to output consistent answers even for identities whose public key has been replaced with other ones for which they do not know the matching private key. The latter requirement might a priori appear too strong to the reader but the authors of [6] argue that decryption queries involving identities whose public keys have been replaced with arbitrary other ones are far more useful to a Type I attacker (and especially when the latter does not know the private key associated to the new public key). Anyway, our new construction of section 5 perfectly supports this constraint that turns out to be one of the noticeable strengths of the model and that does not exist in the certificate based formalism.

In the security analysis of generic constructions in section 3.2, we will illustrate the importance of considering adversaries who replace public keys instead of merely corrupt their owner and learn his/her secret value.

Definition 4.2 ([6]) *A CLE scheme is IND-CCA secure if no probabilistic polynomial time (PPT) adversary $\mathcal{A}$ of Type I or II has a non-negligible advantage in the following game:*

1. *Given a security parameter k, the challenger runs* $\mathsf{Setup}(k)$ *and then delivers the resulting parameters* params *to $\mathcal{A}$ who also receives the master key* mk *if she is of Type II. Otherwise,* mk *is kept secret.*
2. *$\mathcal{A}$ is given access to*
   - *a public key broadcast oracle* $\mathsf{Public\text{-}Key\text{-}Broadcast}$ *taking as input identities and returning the associated public keys.*

- *a partial key exposure oracle* Partial-Private-Key-Extract *(if she is of Type I as such an oracle is useless otherwise) returning partial private keys associated to users' identities.*
- *a private key exposure oracle* Private-Key-Extract *revealing private keys of entities whose public key was not replaced.*
- *a decryption oracle* Decrypt *which, given a ciphertext and an identity* (C, ID)*, returns the decryption of* C *using the private key associated to the current value of entity* ID*'s public key.*

If $\mathcal{A}$ *is of Type I, she has also access to a public key replacement oracle* Public-Key-Replace *which, given an identifier* ID *and a valid public key* pk'*, replaces user* ID*'s public key with* pk'*)*

3. $\mathcal{A}$ *outputs messages* m_0, m_1 *and an identity* ID^* *of uncorrupted private key. If* $\mathcal{A}$ *is of Type I,* ID^* *may not have been queried to both oracles* Public-Key-Replace *and* Partial-Private-Key-Extract*. She gets a ciphertext* $C^* = \mathsf{Encrypt}(m_b, \mathsf{params}, \mathsf{ID}^*, \mathsf{pk}^*)$ *where* $b \overset{R}{\leftarrow} \{0,1\}$ *and* pk^* *is the public key currently associated to* ID^**.*
4. *She then issues a new sequence of queries but is disallowed to ask for the decryption of* C^* *for the combination* $(\mathsf{ID}^*, \mathsf{pk}^*)$ *under which* m_b *was encrypted at step 3. Moreover no private key exposure query can be made on* ID^* *at any time and, in a Type I attack,* ID^* *may not be submitted to both oracles* Public-Key-Replace *and* Partial-Private-Key-Extract*.*
5. $\mathcal{A}$ *eventually outputs a bit* b' *and wins if* $b' = b$*. As usual, her advantage is* $Adv^{\mathsf{ind-cca}}_{CLE}(\mathcal{A}) := 2 \times Pr[b' = b] - 1$*.*

The above definition captures a chosen-ciphertext scenario where maximal power is granted to adversaries. The weaker chosen-plaintext security (or IND-CPA security) notion is formalized by a similar game where attackers have no decryption oracles.

The security models considered in [222, 13] are weaker in the sense that they forbid Type I attackers to ever extract the partial private key of the target entity. The above model allows them to do so if they do not additionally replace the associated public key. Besides, the model of [222] only imposes challengers to correctly handle decryption queries for entities whose public key was not replaced.

From here on, we will stick to the strong model of definition 4.2.

2.3. Review of some previous CLE schemes

In order to provide the reader with a concrete example of a certificateless scheme, the next figure recalls the first solution put forth by Al-Riyami and Paterson in [6]. An improvement of the latter construction is then recalled.

2.3.1. The Al-Riyami-Paterson scheme

The description given in figure 4.1 is obtained by applying the Fujisaki-Okamoto [89] conversion to a simpler version that is only secure against chosen-plaintext attacks. The resulting scheme, called $\mathsf{FullCLE}$, reaches the chosen-ciphertext security in the sense of definition 4.2.

The correctness of the scheme directly stems from the bilinearity of the map:

$$\hat{e}(U, S_A) = \hat{e}(rP, x_A s h_1(\mathsf{ID}_A)) = \hat{e}(x_A sP, h_1(\mathsf{ID}_A))^r.$$

The security relies on the so-called *generalized Bilinear Diffie-Hellman assumption* that was introduced in [6] and which is the infeasibility of finding a pair $\langle Q, \hat{e}(P,Q)^{abc} \rangle \in \mathbb{G}_1 \times \mathbb{G}_2$ given $\langle P, aP, bP, cP \rangle$.

The purpose of the first step of the encryption algorithm is to verify that the public key is correctly formed (i.e. that its components X_A and Y_A have equal discrete logarithms for the bases P and P_{pub}). This public key validation procedure turns out to be expensive as it involves two pairing evaluations but it must fortunately only be performed once at the first use of a public key.

If the detection of dishonest authorities is required, the partial private key generation algorithm can be modified in such a way that entity A's partial private key is $d_A = sh_1(\mathsf{ID}_A || \mathsf{pk}_A)$. This alternative partial private key generation provides a kind of implicit certification of the entity's public key. Dishonest KGCs that issue several partial private keys for the same identity can then be detected and the scheme thus reaches the level 3 in Girault's hierarchy [101]. Interestingly, secure channels are no longer necessary between end-users and the KGC but entities then have to choose their secret value and set out their public key before obtaining their partial and full private keys.

Setup: given security parameters k and k_1 where k_1 is polynomial in k, this algorithm chooses a k-bit prime number q, symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of order q, a generator $P \in \mathbb{G}_1$ and hash functions $h_1 : \{0,1\}^* \to \mathbb{G}_1$, $h_2 : \mathbb{G}_2 \to \{0,1\}^{k_1}$, $h_3 : \{0,1\}^{n+k_1} \to \mathbb{Z}_q^*$ and $h_4 : \{0,1\}^{k_1} \to \{0,1\}^n$. A master key $\mathsf{mk} := s \xleftarrow{R} \mathbb{Z}_q^*$ and a public key $P_{pub} = sP \in \mathbb{G}_1$ are also chosen. The public parameters are

$$\mathsf{params} := \{q, k, k_1, \mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, \hat{e}, h_1, h_2, h_3, h_4, n, \mathcal{M}, \mathcal{C}\}$$

where $\mathcal{M} := \{0,1\}^n$, $\mathcal{C} := \mathbb{G}_1 \times \{0,1\}^{n+k_1}$ are cleartext and ciphertext spaces.

Partial-Private-Key-Extract: takes as input entity A's identifier $\mathsf{ID}_A \in \{0,1\}^*$ and extracts A's partial private key $d_A = sh_1(\mathsf{ID}_A) \in \mathbb{G}_1$.

Set-Secret-Value: given params and A as inputs, this algorithm picks $x_A \in \mathbb{Z}_q^*$ which is returned as user A's secret value.

Set-Private-Key: given params, user A's partial private key $d_A \in \mathbb{G}_1$ and his secret value $x_A \in \mathbb{Z}_q^*$, this algorithm computes the private key $S_A = x_A d_A = x_A sh_1(\mathsf{ID}_A) \in \mathbb{G}_1$.

Set-Public-Key: this algorithm takes as input params and entity A's secret value $x_A \in \mathbb{Z}_q^*$ and produces A's public key

$$\mathsf{pk}_A := \langle X_A = x_A P, Y_A = x_A P_{pub} \rangle \in \mathbb{G}_1 \times \mathbb{G}_1.$$

Encrypt: to encrypt $m \in \{0,1\}^n$ using the identifier $\mathsf{ID}_A \in \{0,1\}^*$ and the public key $\mathsf{pk}_A = \langle X_A = x_A P, Y_A = x_A P_{pub} \rangle$,

(1) Check that $\hat{e}(Y_A, P) = \hat{e}(P_{pub}, X_A)$.
(2) Pick $\sigma \xleftarrow{R} \{0,1\}^{k_1}$, set $r = h_3(\sigma, m) \in \mathbb{Z}_q^*$ and compute the ciphertext

$$C = \langle U, V, W \rangle = \langle rP, \sigma \oplus h_2(\hat{e}(Y_A, h_1(\mathsf{ID}_A))^r), m \oplus h_4(\sigma) \rangle$$

Decrypt: given a ciphertext $C = \langle U, V, W \rangle \in \mathcal{C}$,

(1) Use S_A to recover $\sigma = V \oplus h_2(\hat{e}(U, S_A)) \in \{0,1\}^{k_1}$ and then $m = W \oplus h_4(\sigma) \in \{0,1\}^n$.
(2) Compute $r = h_3(\sigma, m) \in \mathbb{Z}_q^*$, return m if $U = rP$ and $\perp$ otherwise.

FIGURE 4.1. The FullCLE scheme

2.3.2. An improvement of FullCLE

In [7], the inventors of the certificateless paradigm proposed a variant (named FullCLE*) of their scheme that is significantly more efficient than FullCLE in situations where few messages must be encrypted using a

Setup: is the same as in FullCLE except that 5 hash functions are involved: $h_1 : \{0,1\}^* \to \mathbb{G}_1$, $h_2 : \mathbb{G}_2 \to \{0,1\}^{k_1}$, $h_2' : \mathbb{G}_1 \to \{0,1\}^{k_1}$, $h_3 : \{0,1\}^{n+k_1} \to \mathbb{Z}_q^*$, $h_4 : \{0,1\}^{k_1} \to \{0,1\}^n$. A master key $\mathsf{mk} := s \xleftarrow{R} \mathbb{Z}_q^*$ and a public key $P_{pub} = sP \in \mathbb{G}_1$ are also chosen. The public parameters are

$$\mathsf{params} := \{q, k, k_1, \mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, \hat{e}, h_1, h_2, h_2', h_3, h_4, n, \mathcal{M}, \mathcal{C}\}$$

where $\mathcal{M} := \{0,1\}^n$, $\mathcal{C} := \mathbb{G}_1 \times \{0,1\}^{n+k_1}$ are cleartext and ciphertext spaces.

Partial-Private-Key-Extract: is unchanged as well as **Set-Secret-Value**.

Set-Private-Key: given params, user A's partial private key $d_A \in \mathbb{G}_1$ and his secret value $x_A \in \mathbb{Z}_q^*$, the full private key is the pair $S_A = (x_A, d_A) \in \mathbb{Z}_q^* \times \mathbb{G}_1$.

Set-Public-Key: this algorithm takes as input params and entity A's secret value $x_A \in \mathbb{Z}_q^*$ and produces A's public key $Y_A = x_A P \in \mathbb{G}_1$.

Encrypt: to encrypt $m \in \{0,1\}^n$ using the identifier $\mathsf{ID}_A \in \{0,1\}^*$ and the public key $\mathsf{pk}_A = Y_A = x_A P$,

(1) Check that $Y_A^q = 1_{\mathbb{G}_1}$.
(2) Pick $\sigma \xleftarrow{R} \{0,1\}^{k_1}$, set $r = h_3(\sigma, m) \in \mathbb{Z}_q^*$ and compute the ciphertext

$$C = \langle U, V, W \rangle = \langle rP, \sigma \oplus h_2(\hat{e}(P_{pub}, h_1(\mathsf{ID}_A))^r) \oplus h_2'(rY_A), m \oplus h_4(\sigma) \rangle$$

Decrypt: given a ciphertext $C = \langle U, V, W \rangle \in \mathcal{C}$,

(1) Compute $\sigma = V \oplus h_2(\hat{e}(U, d_A)) \oplus h_2'(x_A U) \in \{0,1\}^{k_1}$ and then $m = W \oplus h_4(\sigma) \in \{0,1\}^n$.
(2) Compute $r = h_3(\sigma, m) \in \mathbb{Z}_q^*$, return m if $U = rP$ and $\perp$ otherwise.

FIGURE 4.2. The FullCLE* scheme

given public key. In FullCLE*, public keys are made of a single group element and checking their validity only requires an elliptic curve scalar multiplication but the encryption phase itself entails an additional scalar multiplication in $\mathbb{G}_1$. The plaintext is actually scrambled twice using two distinct superposed one-time masks and, in some sense, the scheme may be regarded as an optimized composition of the Boneh-Franklin IBE with an El Gamal-like encryption.

In order to achieve the security in the sense of definition 4.2, the authors of [7] again applied the Fujisaki-Okamoto conversion [89] recalled in chapter 3. As shown in the next section, this does not suffice as some

special care is needed to integrate IBE systems with traditional public key encryption schemes in order to design secure CLE schemes.

3. On the power of public key replacement oracles

This section highlights the strength of the security model in the chosen-ciphertext scenario. We first exemplify that allowing decryption queries even for entities whose public key has been replaced does harm the security of FullCLE*. We also show how to very easily fix the problem. We then explain how attacks by public key replacements may compromise the security of generic constructions of certificateless encryption.

3.1. The FullCLE* case

We here find that a Type I adversary $\mathcal{A}_I$ can easily break the non-malleability of FullCLE* in the scenario of definition 4.2 by replacing twice the target identity's public key. In more details, if the challenge ciphertext is $C^* = \langle U^*, V^*, W^* \rangle$ and x^* denotes the secret value of the target identity ID^* (that can be known to a Type I adversary $\mathcal{A}$ replacing entity ID^*'s public key before the challenge phase), the attacker can replace the target identity's public key with $x'P$ after the challenge phase and then ask for the decryption of $C' = \langle U^*, V^* \oplus h_2(x^*U) \oplus h_2(x'U), W^* \rangle$ (which is an encryption of the same plaintext as C^* for the combination $(\mathsf{ID}^*, x'P)$). Since decryption queries remain allowed even for entities of replaced public key, $\mathcal{A}_I$ can issue a decryption query on $C' \neq C$ for the identity ID' and recover the plaintext.

Attackers replacing public keys are thus able to take advantage of multiplicative relations between those public keys and parts of ciphertexts. Fortunately, such an attack is easily thwarted by hashing the recipient's public key along with his identity and the pair (σ, m) at step 2 of the encryption algorithm: the encrypting multiplier r is thus obtained as $r = h_3(\sigma, m, \mathsf{ID}_A, \mathsf{pk}_A)$. A variant of FullCLE* independently proposed by Cheng and Comley [59] is immune to the latter "multiplicative" attack because it scrambles σ with a hash value of both rY_A and $\hat{e}(P_{pub}, Q_{\mathsf{ID}_A})^r$ instead of using separate masks. FullCLE and another recently proposed certificateless encryption scheme [32] are also immune to the latter "multiplicative" attack. Interestingly, this attack does not

work anymore if the alternative private key generation method (recalled in section 2.3.1) is used to bind public keys to identities. Neither does it against the certificate-based encryption scheme obtained from $\mathsf{FullCLE}^*$ using the generic transformation of Al-Riyami and Paterson [7].

These observations shed new light on the power of adversaries who are able to replace entities' public keys rather than simply obtaining their secret value from user-corrupting oracles. Actually, the $\mathsf{FullCLE}^*$ scheme remains secure in a model in which attackers cannot replace public keys but are rather provided with an oracle returning secret values of arbitrary identities. The latter model is thus strictly weaker than the one captured by definition 4.2.

3.2. The case of generic constructions

In [5] and [222], generic constructions of certificateless encryption were independently proposed. Their idea is basically to combine strongly secure identity-based and traditional public key encryption schemes in a sequential or parallel fashion. More precisely, if Π^{IBE} and Π^{PKE} respectively denote an IBE and an ordinary public key encryption system, a CLE scheme Π^{CLE} can be obtained with the sequential composition depicted in figure 4.3 and named $\mathsf{Generic\text{-}CLE}$-1. Its security was proved by Yum and Lee [222] in a weak model where adversaries are restricted not to issue a partial key exposure query on the target identity ID^* (recall that such a query is allowed in the strong model if the public key associated to ID^* is never replaced) nor to require the correct decryption of ciphertexts encrypted under identities of replaced public keys.

This construction is insecure against Type I attacks in the full model of definition 4.2 even if its building blocks Π^{IBE} and Π^{PKE} are each IND-CCA secure in their model. We show it using simple arguments such as those given in [80] against the security of naive multiple-encryptions. Let $C^* = \mathcal{E}^{IBE}_{\mathsf{ID}^*}(\mathcal{E}^{PKE}_{\mathsf{pk}^*}(m_b^*))$ be the challenge ciphertext in the game of definition 4.2 where m_b^* (for a random bit $b \in \{0,1\}$) denotes one of the messages produced by the adversary $\mathcal{A}_I$ in her challenge request. Assume that $\mathcal{A}_I$ never replaces the public key of ID^* but rather extracts the partial private key d_{ID^*} after the challenge phase. She then obtains $\mathcal{E}_1 = \mathcal{D}^{IBE}_{d_{\mathsf{ID}^*}}(C^*) = \mathcal{E}^{PKE}_{\mathsf{pk}^*}(m_b)$ and she may compute another encryption $C' = \mathcal{E}^{IBE}_{\mathsf{ID}^*}(\mathcal{E}_1) \neq C^*$ of the same plaintext and obtain m_b^*.

Setup: is an algorithm running the setup algorithms of both Π^{IBE} and Π^{PKE}. The message space of Π^{CLE} is the message space of Π^{PKE} while its ciphertext space is the one of Π^{IBE}. Both schemes have to be compatible in the sense that the plaintext space of Π^{IBE} must contain the ciphertext space of Π^{PKE}.

Partial-Private-Key-Extract: is the private key generation algorithm of Π^{IBE}.

Set-Secret-Value and Set-Public-Key: run the key generation algorithm of Π^{PKE} to obtain a private key sk and a public key pk. The former is the secret value and the latter becomes the public key.

Set-Private-Key: returns $S_A := (d_A, \mathsf{sk}_A)$ where d_A is obtained by running the key generation algorithm of Π^{IBE} for the identity ID_A and sk_A is entity A's secret value obtained from Π^{PKE}'s key generation algorithm.

Encrypt: to encrypt $m \in \mathcal{M}^{PKE}$ using the identifier $\mathsf{ID}_A \in \{0,1\}^*$ and the public key pk_A,

(1) Check that pk_A is a valid public key for Π^{PKE}.
(2) Compute and output the ciphertext $C = \mathcal{E}^{IBE}_{\mathsf{ID}}(\mathcal{E}^{PKE}_{\mathsf{pk}_A}(m))$ where $\mathcal{E}^{IBE}_{\mathsf{ID}}$ and $\mathcal{E}^{PKE}_{\mathsf{pk}_A}$ respectively denote the encryption algorithms of Π^{IBE} and Π^{PKE} for the identity ID and the public key pk_A.

Decrypt: to decrypt C using $S_A = (d_A, \mathsf{sk}_A)$,

(1) Compute $\mathcal{D}^{IBE}_{d_A}(C)$ using the decryption algorithm of Π^{IBE}. If the result is $\perp$, return $\perp$ and reject the ciphertext.
(2) Otherwise, compute $\mathcal{D}^{PKE}_{\mathsf{sk}_A}(\mathcal{D}^{IBE}_{d_A}(C))$ using the decryption algorithm of Π^{PKE} and return the result.

FIGURE 4.3. The Generic-CLE-1 construction

This does not contradict the result of Yum and Lee [222] since they considered a weaker security model in which attackers may not extract the partial private key for the target identity ID^*.

In [5], a reverse-ordered composition (that we call Generic-CLE-2) where ciphertexts have the form $C = \mathcal{E}^{PKE}_{\mathsf{pk}_A}(\mathcal{E}^{IBE}_{\mathsf{ID}}(m))$ is suggested. This composition is vulnerable against an attacker replacing the target entity's public key before the challenge phase. Knowing the secret value sk^* in the challenge phase, the adversary obtains $\mathcal{E}^{IBE}_{\mathsf{ID}^*}(m_b)$ that is re-encrypted into $C' = \mathcal{E}^{PKE}_{\mathsf{pk}^*}(\mathcal{E}^{IBE}_{\mathsf{ID}^*}(m_b)) \neq C^*$ which may be submitted to the decryption oracle even though entity ID^*'s public key was replaced

in the model of [6].

In [5], a 'parallel' construction (that we will call Generic-CLE-3) was also considered. It encrypts a plaintext m into

$$C = \langle \mathcal{E}^{PKE}_{\mathsf{pk}_A}(m_1), \mathcal{E}^{IBE}_{\mathsf{ID}}(m_2) \rangle$$

where m_1 and m_2 are subject to the constraint $m = m_1 \oplus m_2$. This parallel approach is vulnerable to a similar attack to those outlined by Dodis and Katz [80] against multiple-encryption schemes: if $C^* = \langle \mathcal{E}^*_1, \mathcal{E}^*_2 \rangle$ is the challenge ciphertext in the IND-CCA game, both kinds of adversaries $\mathcal{A}_I$ or $\mathcal{A}_{II}$ may first request the decryption of $C'_1 = \langle \mathcal{E}^*_1, \mathcal{E}^{IBE}_{\mathsf{ID}}(0^{IBE}) \rangle$ and then the decryption of $C'_2 = \langle \mathcal{E}^{PKE}_{\mathsf{pk}}(0^{PKE}), \mathcal{E}^*_2 \rangle$, where 0^{PKE} and 0^{IBE} are plaintexts made of zeros in Π^{IBE} and Π^{PKE}. By combining the results m'_1 and m'_2 of both decryption requests into $m'_1 \oplus m'_2$, the adversary $\mathcal{A}_I$ gets back the plaintext encrypted in C^*. This attack works even if Π^{IBE} and Π^{PKE} are both IND-CCA secure and it does not even require $\mathcal{A}_I$ to replace any public key. Unlike the previous two attacks, it also works in the weaker models of [59, 222].

In [80], Dodis and Katz gave generic techniques to thwart such attacks and build IND-CCA secure (possibly parallel) multiple-encryption schemes from public key encryption schemes which are individually IND-CCA. They showed that their methods apply to the design of certificate-based encryption schemes [98] without resorting to the random oracle model. Because of the strong constraint imposed on decryption oracles in definition 4.2, those techniques do not seem to directly apply in the present context (although they do so in the relaxed models considered in [59, 222]). In simulation-based security proofs, the difficulty is that the simulator does not know the secret value of entities whose public key was replaced. It is to note that, unlike the attack outlined against FullCLE*, the first and third attacks do not use the full power of the security model of [6].

4. Secure Generic constructions in the random oracle model

We now explain how to obtain generic constructions that withstand the attacks outlined in section 3.2 and that are provably secure in the random oracle model.

We first show a generic random oracle-using transformation that

turns any IND-CPA certificateless encryption scheme into a secure CLE system in the chosen-ciphertext scenario of definition 4.2. We then show that all the generic compositions recalled in section 3.2 are IND-CPA if they start from chosen-plaintext secure IBE and PKE schemes.

4.1. From chosen-plaintext to chosen-ciphertext security

This transformation is a modification of the first conversion of Fujisaki and Okamoto [88] which is known to provide IND-CCA secure public key encryption schemes from IND-CPA ones. Given an IND-CPA secure cryptosystem $\Pi = (\mathcal{K}, \mathcal{E}_{\mathsf{pk}}, \mathcal{D}_{\mathsf{sk}})$, the original Fujisaki-Okamoto conversion basically encrypts the plaintext along with a random string using a hash value of them as a randomness: the enhanced encryption algorithm of $\Pi' = (\mathcal{K}, \overline{\mathcal{E}}_{\mathsf{pk}}, \overline{D}_{\mathsf{sk}})$ is

$$\overline{\mathcal{E}}_{\mathsf{pk}}(m, \sigma) = \mathcal{E}_{\mathsf{pk}}(m||\sigma, H(m||\sigma))$$

for a random string σ. The matching decryption algorithm is

$$\begin{aligned} \overline{\mathcal{D}}_{\mathsf{sk}}(C) \quad &= \quad m \ \text{ if } \ C = \mathcal{E}_{\mathsf{pk}}(m||\sigma, H(m||\sigma)) \\ &\qquad \text{and } \perp \text{ otherwise} \end{aligned}$$

where $(m||\sigma) = \mathcal{D}_{\mathsf{sk}}(C)$. Our adaptation of this conversion simply consists in hashing the recipient's identity and his public key along with the message and the random string in the enhanced encryption algorithm.

The second Fujisaki-Okamoto [89] conversion, which is used in the FullCLE and FullCLE* schemes, can be customized in the same way to be generically applied in the certificateless setting. Although this second transformation is the most powerful one (as it only starts from a public key encryption scheme that is one-way against chosen-plaintext attacks), we prefer using the first one [88] which is simpler, uses fewer random oracles and is sufficient for our purposes.

To handle decryption queries of the chosen-ciphertext attacker, the strategy of the plaintext extractor is essentially the following: for every new random oracle query on a string $(m||\sigma||\mathsf{pk}||\mathsf{ID})$, it returns a random value r and runs the encryption algorithm of the weakly secure CLE scheme with the identity ID and the public key pk (that may have been replaced or not) to encrypt $(m||\sigma)$ using the randomness r. The resulting ciphertext C is stored in a list. By doing so, the simulator anticipates

subsequent decryption queries, knowing that any valid ciphertext was previously computed and stored in the list with high probability.

Theorem 4.1 *Let Π^{CLE} be an IND-CPA certificateless encryption scheme and suppose that*

$$\mathcal{E}^{\mathsf{params}}_{ID,pk}(M,R) \quad \textit{and} \quad \mathcal{D}^{\mathsf{params}}_{S_{\mathsf{ID}}}$$

are its encryption and decryption algorithms where ID *and* pk *respectively denote the recipient's identity and his public key, M is a message of $n+k_0$ bits, R is a random string of ℓ bits while S_{ID} is the recipient's private decryption key. Then, an IND-CCA certificateless scheme $\overline{\Pi}^{CLE}$ can be obtained using modified encryption and decryption algorithms*

$$\overline{\mathcal{E}}^{\mathsf{params}}_{ID,pk}(m,\sigma) = \mathcal{E}^{\mathsf{params}}_{ID,pk}(m||\sigma, H(m||\sigma||\mathsf{pk}||\mathsf{ID}))$$

where $H : \{0,1\}^* \rightarrow \{0,1\}^\ell$ is a random oracle, $m \in \{0,1\}^n$ is the plaintext and $\sigma \in \{0,1\}^{k_0}$ is a random string. The modified decryption algorithm is

$$\begin{aligned}\overline{\mathcal{D}}^{\mathsf{params}}_{S_{\mathsf{ID}}}(C) \quad &= \quad m \quad \textit{if} \quad C = \mathcal{E}^{\mathsf{params}}_{ID,pk}(m||\sigma, H(m||\sigma||\mathsf{pk}||\mathsf{ID})) \\ &\qquad \textit{and } \perp \textit{ otherwise}\end{aligned}$$

where $(m||\sigma) = \mathcal{D}^{\mathsf{params}}_{S_{\mathsf{ID}}}(C)$.

More precisely, assume that a Type I (resp. Type II) IND-CCA attacker $\mathcal{A}$ has an advantage ϵ against $\overline{\Pi}^{CLE}$ when running in a time τ, making q_D decryption queries and q_H random oracle queries. It implies a Type I (resp. Type II) IND-CPA attacker $\mathcal{B}$ with an advantage

$$\epsilon' > (\epsilon - q_H/2^{k_0-1})(1-2^{-\ell_0})^{q_D}$$

against Π^{CLE} when running in a time $\tau' < \tau + O(q_H\tau_{\mathcal{E}})$, where $\tau_{\mathcal{E}}$ is the cost of the original encryption algorithm and

$$\ell_0 = \log_2\Big(\min_{\substack{m\in\{0,1\}^{n+k_0}\\ \mathsf{ID},\mathsf{pk}}} [\#\{\mathcal{E}^{\mathsf{params}}_{ID,pk}(m,r)|r\in\{0,1\}^\ell\}]\Big)$$

is the logarithm of the cardinality of the smallest set of encrypted values that can be obtained for fixed plaintext, identity and public key.

Proof. The proof is quite similar to the one of theorem 3 in [88] but we have to show that the adapted conversion generically works in our context. We outline how $\mathcal{B}$ uses $\mathcal{A}$ to succeed in a chosen-plaintext attack against her challenger $\mathcal{CH}$. $\mathcal{B}$ starts by forwarding to $\mathcal{A}$ the public parameters (together with the KGC's master key in the scenario

of a Type II attack) she obtains from $\mathcal{CH}$. Recall that Π^{CLE} can be itself a random oracle-using scheme. All random oracles pertaining to Π^{CLE} are thus controlled by $\mathcal{CH}$. The chosen-ciphertext attacker $\mathcal{A}$ has also access to a decryption oracle and an additional random oracle H that are simulated by $\mathcal{B}$ as follows:

- random oracle queries related to Π^{CLE} as well as public key broadcast, public key replacement (in the case of Type I attacks) and partial/full private key exposure queries are passed to $\mathcal{CH}$ whose answers are relayed to $\mathcal{A}$.
- Whenever $\mathcal{A}$ submits a string $(m||\sigma||\mathsf{pk}||\mathsf{ID})$ to the H oracle, $\mathcal{B}$ first checks if H was previously queried on the same input and returns the previously answered value if it was. Otherwise, $\mathcal{B}$ returns a randomly chosen $r \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$. She then runs the encryption algorithm of Π^{CLE} to compute

$$C = \mathcal{E}_{\mathsf{ID},\mathsf{pk}}^{\mathsf{params}}(m||\sigma, r)$$

  which is a $\overline{\Pi}^{CLE}$ encryption of m under the public key pk and the identity ID using the randomness $\sigma \in \{0,1\}^{k_0}$ (as well as a Π^{CLE} encryption of $(m||\sigma)$ for the randomness r). In order to anticipate subsequent decryption queries, a record containing the input $(m||\sigma||\mathsf{pk}||\mathsf{ID})$ as well as the returned r and the corresponding ciphertext C is then stored in a list L_H. Note that $\mathcal{B}$ might need $\mathcal{CH}$ to answer queries for random oracles related to Π^{CLE} to be able to compute C.
- Decryption queries: when $\mathcal{A}$ submits a ciphertext C and an identity ID, $\mathcal{B}$ first recovers the public key pk currently associated to ID (by issuing a public key broadcast query to $\mathcal{CH}$). She then searches in list L_H for a tuple of the form $((m||x||\mathsf{pk}||\mathsf{ID}), r, C)$ in order to return the corresponding m if such a tuple exists and $\perp$ otherwise.

When $\mathcal{A}$ decides that phase 1 is over, she outputs messages (m_0, m_1) and an identity ID^* (whose private key was not exposed and that was not submitted to both Public-Key-Replace and Partial-Private-Key-Extract oracles). At that point, $\mathcal{B}$ obtains the current value pk^* of the public key associated to ID^* (by issuing a Public-Key-Broadcast query to $\mathcal{CH}$)

before randomly choosing two strings $\sigma_0, \sigma_1 \overset{R}{\leftarrow} \{0,1\}^{k_0}$ and in turn sending her challenge request $(M_0 = (m_0||\sigma_0), M_1 = (m_1||\sigma_1), \mathsf{ID}^*)$ to $\mathcal{CH}$. The latter then returns a Π^{CLE} encryption C^* of $M_b = (m_b||\sigma_b)$ for the identity ID^* and the currently associated public key pk^* using some coin $r^* \overset{R}{\leftarrow} \mathbb{Z}_q^*$.

As in the proof of theorem 2 in [88], if $\mathcal{A}$ ever queries H on the input $(m_d||\sigma_d||\mathsf{pk}^*||\mathsf{ID}^*)$ for $d \in \{0,1\}$, $\mathcal{B}$ halts and outputs the corresponding bit d as a result which is very likely to be correct in this case: since $\mathcal{A}$ has absolutely no information on $\sigma_{\overline{b}}$ ($\overline{b}$ being the complement bit of b), one can show as in [88] that $\mathcal{A}$ only asks for the hash value $H(m_{\overline{b}}||\sigma_{\overline{b}}||\mathsf{pk}^*||\mathsf{ID}^*)$ with probability $q_H/2^{k_0}$ throughout the game). On the other hand, if such an H-query never occurs, $\mathcal{B}$ outputs exactly the same result b' as $\mathcal{A}$ does and obviously succeeds against $\mathcal{CH}$ if $\mathcal{A}$ yields a correct guess $b' = b$.

The probability for $\mathcal{B}$ to wrongly reject a ciphertext during the game is smaller than $1-(1-2^{-\ell_0})^{q_D}$. Indeed, for a given decryption query on a ciphertext C and an identity ID, assume that $(m||\sigma) = \mathcal{D}_{S_{\mathsf{ID}}}^{\mathsf{params}}(C)$ and does not figure (together with ID and pk) in list L_H. The probability that $H(m||\sigma||\mathsf{pk}||\mathsf{ID})$ takes a value encrypting $(m||\sigma)$ into C is at most $2^{-\ell_0}$ (as at most $2^{\ell-\ell_0}$ distinct random values $r \in R$ may encrypt a given ciphertext into the same ciphertext by the definition of ℓ_0) .

It comes that $\mathcal{B}$'s advantage against $\mathcal{CH}$ is at least

$$\epsilon' > (\epsilon - q_H/2^{k_0-1})(1-2^{-\ell_0})^{q_D}$$

and that her running time is bounded by $\tau' < \tau + O(q_H\tau_{\mathcal{E}})$ where $\tau_{\mathcal{E}}$ is the time complexity of the encryption algorithm of the basic scheme Π^{CLE}. She also has to issue $q_D + 1$ public key broadcast oracle queries to $\mathcal{CH}$ and q_H queries to random oracles pertaining to Π^{CLE}. □

4.2. Generic IND-CPA secure compositions

From now, we only have to consider generic constructions that are only secure against chosen-plaintext attacks. By applying to them the random oracle-using conversion, we end up with provably secure generic constructions in the random oracle model.

Let $\Pi^{IBE} = (\mathsf{Setup}^{IBE}, \mathsf{Keygen}^{IBE}, \mathcal{E}^{IBE}, \mathcal{D}^{IBE})$ be an IBE scheme and $\Pi^{PKE} = (\mathcal{K}^{PKE}, \mathcal{E}_{pk}^{PKE}, \mathcal{D}_{\mathsf{sk}}^{PKE})$ be a traditional public key encryption scheme.

Theorem 4.2 *If Π^{IBE} is IND-ID-CPA and Π^{PKE} is IND-CPA, then the Generic-CLE-1 is IND-CPA.*

To prove the above theorem, we need to separately consider Type I and Type II adversaries.

Lemma 4.1 *A Type I IND-CPA adversary $\mathcal{A}_I$ having an advantage ϵ against Generic-CLE-1 implies either an IND-ID-CPA adversary with advantage $\epsilon/(2q_{\mathsf{ID}})$ against Π^{IBE} or an IND-CPA adversary with advantage $\epsilon/(2q_{\mathsf{ID}})$ against Π^{PKE}, where q_{ID} is the total number of distinct identities involved in $\mathcal{A}_I$'s requests.*

Proof. We show how to construct from $\mathcal{A}_I$ an adversary $\mathcal{B}$ that either performs an IND-ID-CPA attack against Π^{IBE} or an IND-CPA attack against Π^{PKE}. We assume that challengers $\mathcal{C}^{IBE}$, $\mathcal{C}^{PKE}$ for both games are available to $\mathcal{B}$.

We distinguish two kinds of Type I adversaries:

- **Type I-A adversaries:** choose to replace the public key associated to the target identity ID^* at some moment of the game (they thus cannot ask for the corresponding partial private key).
- **Type I-B adversaries:** do not replace the public key of the target identity ID^* but rather decide to ask for the associated partial private key at some moment.

Before the initialization phase, $\mathcal{B}$ has to guess which kind of Type I adversary $\mathcal{A}$ will be during the game. She thus begins by choosing a random bit $c \stackrel{R}{\leftarrow} \{0,1\}$. If $c = 0$, $\mathcal{B}$ bets on a Type I-A attack from $\mathcal{A}$. She chooses to play against $\mathcal{C}^{IBE}$ and aborts $\mathcal{C}^{PKE}$. If $c = 1$, she hopes that $\mathcal{A}$ will act as a Type I-B adversary and rather plays against $\mathcal{C}^{PKE}$.

In $\mathcal{B}$'s interaction with $\mathcal{A}_I$, we call ID_i the i^{th} distinct identity that is the subject of a query (hash query, full or partial key extraction, public key replacement or even challenge query) made by $\mathcal{A}_I$. Let q_{ID} be the total number of distinct identities involved in some query (including the unique challenge query). The adversary $\mathcal{B}$ randomly chooses an index $\ell \in \{1, \ldots, q_{\mathsf{ID}}\}$. Depending on the bit c, the setup phase is performed in two different ways:

- **Case $c = 0$:** $\mathcal{B}$ generates the public parameters of the KGC for the CLE scheme. Those include the public parameters params

of Π^{IBE} which are generated by $\mathcal{C}^{IBE}$ and the space of public keys of Π^{PKE}.

Case $c = 1$**:** $\mathcal{B}$ runs herself the Setup algorithm of Π^{IBE} to obtain a master key mk and public parameters params including a system-wide public key P_{pub}. The KGC's public parameters also include the description of the public key space of Π^{PKE}. Before sending them to $\mathcal{A}_I$, $\mathcal{B}$ also obtains a challenge public key pk^* from its challenger $\mathcal{C}^{PKE}$.

We call $\mathcal{H}$ the event that $\mathcal{A}_I$ chooses to be challenged on the target identity ID_ℓ. We denote by $\mathcal{F}_0$ the event that she extracts the partial private key for ID_ℓ and we let $\mathcal{F}_1$ be the event that she replaces the public key of entity ID_ℓ at some point of the attack.

As in the proof techniques of [5, 6, 7] $\mathcal{B}$ uses a strategy which is roughly the following. If $c = 0$ and events $\mathcal{F}_0$ and $\mathcal{H}$ occur, $\mathcal{B}$ will have to abort (as it will have failed in guessing which kind of Type I adversary is played by $\mathcal{A}_I$) exactly as if $\mathcal{F}_1$ and $\mathcal{H}$ both occur while $c = 1$. On the other hand, a private key extraction query on the identity ID_ℓ also causes $\mathcal{B}$'s abortion whereas if either $\neg\mathcal{F}_0 \wedge \mathcal{H}$ or $\neg\mathcal{F}_1 \wedge \mathcal{H}$ occurs, $\mathcal{B}$'s success probability will be related to that of $\mathcal{A}$.

Throughout the game, $\mathcal{A}_I$'s queries are dealt with as follows:

- queries to $\mathsf{Public\text{-}Key\text{-}Broadcast}$ on input ID_ν (i.e. the ν^{th} distinct identity to be ever involved in some query): $\mathcal{B}$ returns the previously assigned public key if it exists. Otherwise,
  - in the case $c = 0$: $\mathcal{B}$ runs the key generation algorithm $\mathcal{K}^{PKE}$ of Π^{PKE} to obtain a key pair $(\mathsf{pk}_\nu, \mathsf{sk}_\nu)$. The public key pk is returned and the triple $(\mathsf{ID}_\nu, \mathsf{sk}_\nu, \mathsf{pk}_\nu)$ is stored in a list L_{pub}.
  - in the case $c = 1$: if $\nu = \ell$, $\mathcal{B}$ returns the challenge public key pk^*. If $\nu \neq \ell$, $\mathcal{B}$ responds with a public key pk_ν obtained from the $\mathcal{K}^{PKE}$ algorithm and stores the matching private key sk_ν in an entry $(\mathsf{ID}_\nu, \mathsf{sk}_\nu, \mathsf{pk}_\nu)$ of L_{pub}.
- queries to $\mathsf{Partial\text{-}Private\text{-}Key\text{-}Extract}$ on an input ID_ν:

  - in the case $c = 0$: if $\nu = \ell$, then $\mathcal{B}$ stops and declares "failure". Otherwise, she asks $\mathcal{C}^{IBE}$ for the private key d_{ID_ν} associate to ID_ν and passes the answer back to $\mathcal{A}_I$.
  - in the case $c = 1$, $\mathcal{B}$ normally runs the partial key generation algorithm as she knows the master key.

- queries to Private-Key-Extract on an input ID_ν: if $\nu = \ell$, $\mathcal{B}$ fails. Otherwise, from the specification of the Public-Key-Broadcast simulator, it comes that $\mathcal{B}$ necessarily knows the associated secret value sk_ν which can be recovered from L_{pub} (recall that such a query is only authorized if entity ID_ν's public key was not replaced).
  - if $c = 0$, $\mathcal{B}$ asks $\mathcal{C}^{IBE}$ for the private key d_{ID_ν} of the identity ID_ν. She then returns the full private key $(d_{\mathsf{ID}_\nu}, \mathsf{sk}_\nu)$.
  - if $c = 1$, $\mathcal{B}$ extracts the partial private key herself (using the master secret mk) and then the full private key using the known secret value sk_ν retrieved from L_{pub}.

- queries to Public-Key-Replace for some identity ID_ν and public key pk': if $\nu = \ell$ and $c = 1$, $\mathcal{B}$ fails. Otherwise, she ensures that pk' has the right form and, if so, replaces the corresponding triple of L_{pub} with $(\mathsf{ID}_\nu, ?, \mathsf{pk}')$ (where ? stands for an unknown secret value).

At some moment, $\mathcal{A}$ ends the first phase by outputting a target identity ID^* together with messages m_0, m_1 taken from the plaintext space $\mathcal{M}^{PKE}$ of Π^{PKE}. If $\mathsf{ID}^* \neq \mathsf{ID}_\ell$, $\mathcal{B}$ fails. Otherwise, two distinct strategies are used to build a challenge ciphertext for the entity ID_ℓ and the current value pk_ℓ of the associated public key (which is pk^* in the case $c = 1$).

Case $c = 0$ **:** $\mathcal{B}$ encrypts both m_0 and m_1 into $c_0 = \mathcal{E}_{\mathsf{pk}_\ell}(m_0)$ and $c_1 = \mathcal{E}_{\mathsf{pk}_\ell}(m_1)$ which are sent to $\mathcal{C}^{IBE}$ together with the target identity ID^* as a challenge request. The resulting challenge $C^* = \mathcal{E}^{IBE}_{\mathsf{ID}^*}(c_b)$ (where $b \in \{0, 1\}$ is a random bit chosen by $\mathcal{C}^{IBE}$) is transmitted to $\mathcal{A}_I$.

Case $c = 1$ **:** $\mathcal{B}$ sends m_0 and m_1 as a challenge request to $\mathcal{C}^{PKE}$. The latter responds with a ciphertext $c^* = \mathcal{E}_{\mathsf{pk}^*}(m_b)$ (for a random bit $b \in \{0, 1\}$) which is in turn encrypted by $\mathcal{B}$ into

$C^* = \mathcal{E}^{IBE}_{\mathsf{ID}^*}(c^*)$ that is given to $\mathcal{A}_I$.

In the second phase, $\mathcal{A}_I$'s environment is simulated as previously. Eventually, $\mathcal{A}_I$ outputs a bit b' which is produced by $\mathcal{B}$ as a result. Clearly, if the simulation has not failed and if $\mathcal{A}_I$ is successful, so is $\mathcal{B}$.

The latter's advantage can be assessed by a similar reasoning to lemma 2 of [6]: $\mathcal{B}$ reaches a failure state in the following situations:

0. For $i = 0, 1$, if $\mathcal{F}_i$ occurs while $c = i$. We call these events $\mathcal{H}_i$.
1. Because of a private key exposure query for the identity ID_ℓ. We let $\mathcal{F}_2$ denote this event.
2. Or because $\mathcal{A}$ chooses a target identity $\mathsf{ID}^* \neq \mathsf{ID}_\ell$. This corresponds to the event $\neg\mathcal{H}$.

As in [6], event $\mathcal{H}$ implies $\neg\mathcal{F}_2$ so that

$$\begin{aligned}\Pr[\mathcal{B} \text{ does not abort}] &= \Pr[\neg\mathcal{H}_0 \wedge \neg\mathcal{H}_1 \wedge \mathcal{H}] \\ &= \Pr[\neg\mathcal{H}_0 \wedge \neg\mathcal{H}_1|\mathcal{H}]\Pr[\mathcal{H}] = \frac{1}{q_{\mathsf{ID}}}\Pr[\neg\mathcal{H}_0 \wedge \neg\mathcal{H}_1|\mathcal{H}] \\ &= \frac{1}{q_{\mathsf{ID}}}(1 - \Pr[\mathcal{H}_0|\mathcal{H}] - \Pr[\mathcal{H}_1|\mathcal{H}])\end{aligned}$$

where the last equality follows from the fact that $\Pr[\mathcal{H}_0 \wedge \mathcal{H}_1|\mathcal{H}] = 0$. On the other hand, as in [6], we have

$$\Pr[\mathcal{H}_i|\mathcal{H}] = \Pr[(c = i) \wedge \mathcal{F}_i|\mathcal{H}] = \frac{1}{2}\Pr[\mathcal{F}_i|\mathcal{H}]$$

since the event $\mathcal{F}_i|\mathcal{H}$ is independent of the event $(c = i)$. Finally, as $\Pr[\mathcal{F}_0 \wedge \mathcal{F}_1|\mathcal{H}] = 0$, we have $\Pr[\mathcal{F}_0|\mathcal{H}] + \Pr[\mathcal{F}_1|\mathcal{H}] \leq 1$ and it comes that

$$\Pr[\mathcal{B} \text{ does not abort}] \geq \frac{1}{2q_{\mathsf{ID}}}.$$

□

Lemma 4.2 *A Type II IND-CPA adversary $\mathcal{A}_{II}$ having an advantage ϵ against Generic-CLE-1 implies an IND-CPA adversary $\mathcal{B}$ with advantage ϵ/q_{ID} against Π^{PKE}, where q_{ID} is the total number of distinct identities involved in $\mathcal{A}_I$'s requests.*

Proof. We describe how $\mathcal{B}$ uses $\mathcal{A}$ to break the chosen-plaintext security of Π^{PKE}.

$\mathcal{B}$ first runs the Setup algorithm of Π^{IBE} to obtain a KGC's key

pair (P_{pub}, mk) and public parameters params. She also selects an index $\ell \xleftarrow{R} \{1, \ldots, q_{\mathsf{ID}}\}$ where q_{ID} is the total number of identities involved in some query (including the challenge query) and obtains a challenge public key pk^* from its challenger $\mathcal{C}^{PKE}$. The adversary $\mathcal{A}_{II}$ is launched with params, and mk as inputs.

$\mathcal{A}_{II}$ then performs a polynomially bounded number of queries handled as follows.

- queries to Public-Key-Broadcast on input ID_i (we call ID_i the i^{th} distinct identity involved in some query): if $i = \ell$, $\mathcal{B}$ returns pk^*. Otherwise, she runs the key generation algorithm $\mathcal{K}^{PKE}$ to obtain a key pair $(\mathsf{pk}_i, \mathsf{sk}_i)$ and returns pk_i. The triple $(\mathsf{ID}_i, \mathsf{pk}_i, \mathsf{sk}_i)$ is stored in a list L_{pub}.
- queries to Partial-Private-Key-Extract on an input ID_i: we assume that ID_i was previously submitted to Public-Key-Broadcast (otherwise, $\mathcal{B}$ can still make the latter query for itself). If $i = \ell$, $\mathcal{B}$ aborts. Otherwise, she knows the corresponding secret value sk_i (stored in L_{pub}) and the partial private key d_{ID_i} (which is computable using the master key mk) and returns the private key $(d_{\mathsf{ID}_i}, \mathsf{sk}_i)$.

At the challenge step, $\mathcal{A}$ outputs messages (m_0, m_1) and a target identity ID^*. At that point, $\mathcal{B}$ aborts if $\mathsf{ID}^* \neq \mathsf{ID}_\ell$. Otherwise, she forwards (m_0, m_1) as a challenge query to $\mathcal{C}^{PKE}$ which responds with $c^* = \mathcal{E}\mathsf{pk}^{*PKE}(m_b)$ for a random bit $b \in \{0, 1\}$. The latter ciphertext is further encrypted into $C^* = \mathcal{E}^{IBE}_{\mathsf{ID}^*}(c^*)$ and given as a challenge to $\mathcal{A}_{II}$.

Adversarial queries in the guess stage are treated as in the find stage and $\mathcal{A}_{II}$'s final result $b' \in \{0, 1\}$ is output by $\mathcal{B}$ as a guess for the hidden bit of $\mathcal{C}^{PKE}$. Clearly, if $\mathcal{A}_{II}$ is successful, so is $\mathcal{B}$. The latter has a probability of $1/q_{\mathsf{ID}}$ to successfully guess the identity on which $\mathcal{A}_{II}$ produces her attack. Moreover, if $\mathcal{B}$ is lucky and correctly guesses, she never aborts when answering a private key extraction query. It comes that, if $\mathcal{A}_{II}$ has an advantage ϵ, $\mathcal{B}$ has an advantage ϵ/q_{ID}. □

The proofs of chosen-plaintext security of Generic-CLE-2 and Generic-CLE-3 are very similar and omitted here. In lemmas 4.1 and 4.2, q_{ID} can be the number of random oracle queries for hash functions mapping

identifiers onto cyclic subgroups or finite fields if we assume that any query involving a given identity comes after a hash query on it.

This shows how to obtain secure generic constructions in the random oracle model. In the case of Generic-CLE-1, if the encryption schemes of Π^{PKE} and Π^{IBE} use distinct sets of randomness R_1 and R_2, the enhanced CLE scheme should use a random oracle $H : \{0,1\}^* \to R_1 \times R_2$ so that an encryption of a plaintext m using the random string σ is given by

$$\overline{\mathcal{E}}^{CLE}_{\mathsf{ID},\mathsf{pk}}(m||\sigma) = \mathcal{E}^{IBE}_{\mathsf{ID}}(\mathcal{E}^{PKE}_{\mathsf{pk}}(m||\sigma, r_1), r_2)$$

where $(r_1||r_2) = H(m||\sigma||\mathsf{pk}||\mathsf{ID})$. In the case of Generic-CLE-3, we have

$$\overline{\mathcal{E}}^{CLE}_{\mathsf{ID},\mathsf{pk}}(m||\sigma) = \langle \mathcal{E}^{PKE}_{\mathsf{pk}}(m_1, r_1), \mathcal{E}^{IBE}_{\mathsf{ID}}(m_2, r_2)\rangle$$

with $m_1 \oplus m_2 = m||\sigma$.

5. A new efficient construction

We here present our new efficient certificateless encryption scheme. The security of our construction is proved to rely on the intractability of the following problem that was introduced in [35] by Boneh and Boyen.

Definition 4.3 ([35]) *The* **p-*Bilinear Diffie-Hellman Inversion problem*** *(p-BDHI) consists in, given* $\langle P, \alpha P, \alpha^2 P, \ldots, \alpha^p P\rangle \in \mathbb{G}_1^{p+1}$, *computing* $\hat{e}(P,P)^{1/\alpha} \in \mathbb{G}_2$.

The p-Bilinear Diffie-Hellman assumption is the intractability of the above problem. It was used by Boneh and Boyen [35] to prove the security of a selective-ID [49] secure identity-based encryption scheme in the standard model. Its decisional variant (i.e. the infeasibility of distinguishing $\hat{e}(P,P)^{1/\alpha}$ from random elements of $\mathbb{G}_2$ even after having seen $\langle P, \alpha P, \alpha^2 P, \ldots, \alpha^p P\rangle$) was more recently studied in [82] where lower bounds were given on its hardness in generic groups.

5.1. The scheme

This new scheme is called NewFullCLE to distinguish it from its simplest form NewBasicCLE that only reaches the chosen-plaintext security level. In this construction, partial private keys are signatures computed using a signature scheme independently considered in [37] and [227] unlike previous CLE schemes [6, 7, 59] that use partial private keys computed according to Boneh et al.'s short signature algorithm

Setup: given security parameters k,k_0 so that k_0 is polynomial in k, this algorithm chooses a k-bit prime number q, symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of order q, a generator $P \in \mathbb{G}_1$ and hash functions $h_1 : \{0,1\}^* \to \mathbb{Z}_q^*$, $h_2 : {\mathbb{G}_2}^2 \to \{0,1\}^{n+k_0}$, $h_3 : \{0,1\}^* \to \mathbb{Z}_q^*$. A master key $\mathsf{mk} := s \xleftarrow{R} \mathbb{Z}_q^*$ and a public key $P_{pub} = sP \in \mathbb{G}_1$ are also chosen. The group element $g = \hat{e}(P,P) \in \mathbb{G}_2$ is also included among the public parameters which are

$$\mathsf{params} := \{q, k, k_0, \mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, g, \hat{e}, h_1, h_2, h_3, n, \mathcal{M}, \mathcal{C}\}$$

where $\mathcal{M} := \{0,1\}^n$, $\mathcal{C} := \mathbb{G}_1 \times \{0,1\}^{n+k_0}$ respectively denote cleartext and ciphertext spaces.

Partial-Private-Key-Extract: takes as input entity A's identifier ID_A and extracts A's partial private key $d_A = \frac{1}{s+h_1(\mathsf{ID}_A)} P \in \mathbb{G}_1$.

Set-Secret-Value: given params and A as inputs, this algorithm picks $x_A \xleftarrow{R} \mathbb{Z}_q^*$ which is returned as user A's secret value.

Set-Private-Key: given params, user A's partial private key $d_A \in \mathbb{G}_1$ and his secret value $x_A \in \mathbb{Z}_q^*$, this algorithm returns the pair $S_A = (x_A, d_A) \in \mathbb{Z}_q^* \times \mathbb{G}_1$ as a private key.

Set-Public-Key: takes as input params and entity A's secret value $x_A \in \mathbb{Z}_q^*$ and produces A's public key $\mathsf{pk}_A := y_A = g^{x_A} \in \mathbb{G}_2$.

Encrypt: to encrypt $m \in \{0,1\}^n$ using the identifier $\mathsf{ID}_A \in \{0,1\}^*$ and the public key $\mathsf{pk}_A = y_A = g^{x_A}$, the sender

(1) Checks that $y_A^q = 1_{\mathbb{G}_2}$.
(2) Picks $\sigma \xleftarrow{R} \{0,1\}^{k_0}$, computes $r = h_3(m||\sigma||\mathsf{pk}_A||\mathsf{ID}_A) \in \mathbb{Z}_q^*$ and the ciphertext is

$$C = \langle c_1, c_2 \rangle = \langle r h_1(\mathsf{ID}_A) P + r P_{pub}, (m||\sigma) \oplus h_2(g^r || y_A^r) \rangle$$

Decrypt: given a ciphertext $C = \langle c_1, c_2 \rangle \in \mathcal{C}$, the receiver uses his partial private key d_A to compute $\omega = \hat{e}(c_1, d_A)$ and then $(m||\sigma) = c_2 \oplus h_2(\omega||\omega^{x_A}) \in \{0,1\}^{n+k_0}$. The message is accepted iff $c_1 = r(h_1(\mathsf{ID}_A)P + P_{pub})$ with $r = h_3(m||\sigma||\mathsf{pk}_A||\mathsf{ID}_A) \in \mathbb{Z}_q^*$.

FIGURE 4.4. The NewFullCLE scheme

[43]. The NewFullCLE scheme is constructed on the Sakai-Kasahara IBE [189, 55, 56] which bears itself similarities with the second selective-ID secure identity-based encryption scheme that was proved secure without random oracles by Boneh and Boyen [35]. As for the Cheng-Chen

[55] variant of the Sakai-Kasahara IBE, its security proof holds in the random oracle model [29]. The correctness of the above construction is easy to check as we have

$$\hat{e}(rh_1(\mathsf{ID}_A)P + rP_{pub}, d_A) \\ = e\big(r(s + h_1(\mathsf{ID}_A))P, \frac{1}{s + h_1(\mathsf{ID}_A)}P\big) = \hat{e}(P, P)^r.$$

Including g^r among the inputs of h_2 in step 2 of the encryption algorithm is necessary to achieve a security reduction under the p-BDHI assumption. The string $(m||\sigma)$ could be hidden by a hash value of only y_A^r but the security would have to rely on a newly defined unnatural assumption.

Interestingly, hashing g^r along with y_A^r is no longer necessary if the scheme is transformed into a certificate-based encryption scheme in the sense of Gentry [98]. This is due to particularities of the certificate-based security model which are not detailed here.

5.2. Efficiency discussions

As for the second CLE scheme proposed by Al-Riyami and Paterson [7], the validity of the public key can be checked very efficiently. As in FullCLE*, assuming that the bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ are chosen by a higher level authority and commonly used by several distinct KGCs, end-users may generate their public key independently of any authority in the system.

The encryption algorithm only entails two exponentiations in $\mathbb{G}_2$ and a multi-exponentiation in $\mathbb{G}_1$. The receiver has to compute a pairing, a single exponentiation in $\mathbb{G}_2$ and multi-exponentiation in $\mathbb{G}_1$. The decryption operation may be optimized by the receiver who can precompute and store $h_1(\mathsf{ID}_A)P + P_{pub}$ in such a way that a simple scalar multiplication in $\mathbb{G}_1$ suffices to verify the validity of the ciphertext. Such a pre-computation also enables to speed up the encryption operation for senders who encrypt several messages under the same public key. Their workload then becomes comparable to the complexity of an El Gamal [96] encryption.

From a computational point of view, NewFullCLE has the same efficiency as FullCLE* [7] if pre-computations are used in both schemes (although NewFullCLE might be more efficient on curves of embedding

degree 2 as an exponentiation in $\mathbb{G}_2$ is cheaper than a scalar multiplication in $\mathbb{G}_1$ in this case) as the pairing can be computed in advance for each identity in $\mathsf{FullCLE}^*$. However, our construction performs better in the absence of pre-computations as its encryption procedure does not compute any pairing. The encryption algorithm is also faster than its counterpart in schemes of [59, 32] for similar parameters and without pre-computations. Moreover, $\mathsf{NewFullCLE}$ does not need a special (and much less efficient) hash function mapping strings onto a cyclic group (and it thus benefits from a faster partial private key generation algorithm) while all schemes have comparable complexities at decryption.

Regarding key sizes, users' public keys lie in $\mathbb{G}_2$ and thus have longer representations (typically 1024 bits without optimizations) than elements in $\mathbb{G}_1$. However, pairing compression techniques due to Barreto and Scott [24] allow to compress them to the third (say 342 bits) of their original length on supersingular curves in characteristic 3 or even to 1/6 of their length using ordinary curves such as those of Barreto and Naehrig [23]. Those compression techniques additionally increase the speed of exponentiations in $\mathbb{G}_2$.

The version of the scheme depicted in section 5.1 uses symmetric pairings that can only be instantiated with supersingular curves. However, it can be described in terms of asymmetric pairings and ordinary curves as well. In environments where bandwidth is of primary concern, it might be desirable to minimize the size of ciphertexts even at the expense of a long system-wide public key (which is less likely to transit across the network). In such a setting, it is then preferable to instantiate the scheme with asymmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ and ordinary curves such as MNT curves or BN curves [155, 23]. In this case, a publicly computable but non-necessarily invertible isomorphism such as the one specified by definition 1.5 must be available. Users' public keys still lie in $\mathbb{G}_T$ while the system-wide public key and entities' partial private keys should respectively be $P_{pub} = sP_2$ and $d_A = 1/(h_1(\mathsf{ID}_A)+s)P_2$ for generators $P_2 \in \mathbb{G}_2$ and $P_1 = \psi(P_2) \in \mathbb{G}_1$. In that bandwidth-optimized version of the scheme, users' public keys can be about 512-bit long on MNT curves [155] or even shorter on BN curves [23]. Ciphertexts are 331 bits longer than plaintexts if $k_0 = 160$.

5.3. Security results

As announced, we give a proof of security under the p-Bilinear Diffie-Hellman Inversion assumption.

Theorem 4.3 *If hash functions h_i $(i = 1, 2, 3)$ are modelled as random oracles, the NewFullCLE scheme is secure in the sense of definition 4.2 under the p-BDHI assumption.*

Setup: given security parameters k and k_0 so that k_0 is polynomial in k, this algorithm outputs a k-bit prime q, the description of symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of order q, a generator $P \in \mathbb{G}_1$ and hash functions $h_1 : \{0,1\}^* \to \mathbb{Z}_q$, $h_2 : {\mathbb{G}_2}^2 \to \{0,1\}^n$. The algorithm also selects a master key $\mathsf{mk} := s \xleftarrow{R} \mathbb{Z}_q^*$ and sets $P_{pub} = sP \in \mathbb{G}_1$ as the corresponding public key. The ciphertext space is $\mathcal{C} := \mathbb{G}_1 \times \{0,1\}^n$ while the space of plaintexts is $\mathcal{M} := \{0,1\}^n$. The public parameters also contain $g = \hat{e}(P, P) \in \mathbb{G}_2$:

$$\mathsf{params} := \{k, k_0, q, \mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, g, \hat{e}, h_1, h_2, n, \mathcal{M}, \mathcal{C}\}.$$

Partial-Private-Key-Extract: is the same as in NewFullCLE just like **Set-Secret-Value**, **Set-Private-Key** and **Set-Public-Key**.

Encrypt: to encrypt $m \in \{0,1\}^n$ using identifier $\mathsf{ID}_A \in \{0,1\}^*$ and the public key $\mathsf{pk}_A = y_A = g^{x_A} \in \mathbb{G}_2$, the sender

(1) Checks that $y_A^q = 1_{\mathbb{G}_2}$.
(2) Chooses a random $r \xleftarrow{R} \mathbb{Z}_p^*$ and computes the ciphertext

$$C = \langle c_1, c_2 \rangle = \langle r h_1(\mathsf{ID}_A) P + r P_{pub}, m \oplus h_2(g^r || y_A^r) \rangle$$

Decrypt: given a ciphertext $C = \langle c_1, c_2 \rangle \in \mathcal{C}$, the receiver computes $\omega = \hat{e}(c_1, d_A)$ and then $m = c_2 \oplus h_2(\omega || \omega^{x_A}) \in \{0,1\}^n$.

FIGURE 4.5. The NewBasicCLE scheme

The proof of the above theorem is rather long. In a first step, lemma 4.3, which is a simple corollary of theorem 4.1, shows that an IND-CCA attacker of Type I or II against NewFullCLE implies the same type of chosen plaintext attacker against a simplified version of the scheme called NewBasicCLE. In a second step, the proof separately considers both kinds of adversaries to establish the chosen-plaintext security of NewBasicCLE.

For Type II adversaries, we only need the weaker 1-BDHI assumption (that is, the hardness of the p-BDHI problem with $p = 1$).

Lemma 4.3 *A Type I (resp. Type II) IND-CCA attacker $\mathcal{A}$ having an advantage ϵ against* NewFullCLE *when running in a time τ, making q_D decryption queries and q_{h_i} queries to oracles h_i ($i = 1, 2, 3$) implies a Type I (resp. Type II) IND-CPA attacker $\mathcal{B}$ with an advantage*

$$\epsilon' > (\epsilon - q_{h_3}/2^{k_0-1})(1 - 2^{-k})^{q_D}$$

against NewBasicCLE *when runnning in a time $\tau' < \tau + O(q_{h_3}\tau_{mult})$, where τ_{mult} is the maximum of the costs of a scalar multiplication in $\mathbb{G}_1$ and an exponentiation in $\mathbb{G}_2$, and making $q_D + 1$ public key broadcast queries, $q_{h_1} + q_{h_3}$ queries to h_1 and $q_{h_2} + q_{h_3}$ queries to h_2.*

The following theorem claims that Type I adversaries are harmless against NewFullCLE as a Type I attacker would imply a PPT algorithm solving the p-BDHI problem with a non-negligible probability.

Lemma 4.4 *Assume that a Type I IND-CCA attacker $\mathcal{A}$ has an advantage ϵ over* NewFullCLE *when running a time τ, making q_{h_i} queries to random oracles h_i ($i = 1, 2, 3$), q_D decryption queries, q_{pk} public key queries, q_{ke} private key extraction queries. Then there is an algorithm $\mathcal{B}$ solving the p-BDHI problem, for $p = q_{h_1}$, with a probability*

$$\epsilon' > \frac{1}{2(q_{h_1} + q_{h_3})(q_{h_2} + q_{h_3})}(\epsilon - q_{h_3}/2^{k_0-1})(1 - 2^{-k})^{q_D}$$

within a time $\tau' < \tau + O((q_{h_1}^2 + q_{ke} + q_{pk} + q_D + q_{h_3})\tau_{exp})$ where τ_{exp} is the maximum of the costs of a scalar multiplication in $\mathbb{G}_1$ and an exponentiation in $\mathbb{G}_2$.

The proof of lemma 4.4 combines the assertions of lemma 4.3 and lemma 4.5 below which claims that a chosen-plaintext adversary of Type I implies an algorithm solving the p-BDHI problem.

Lemma 4.5 *Suppose that a Type I IND-CPA adversary $\mathcal{A}_I$ has an advantage ϵ over* NewBasicCLE *when running in a time τ, asking q_{h_i} queries to random oracles h_i ($i = 1, 2$), q_{ke} private key extraction queries and q_{pk} public key queries. Then there exists an algorithm $\mathcal{B}$ to solve the p-BDHI problem with $p = q_{h_1}$ with an advantage $\epsilon' > \epsilon/2(q_{h_1}q_{h_2})$ and within a time $\tau' < \tau + O((q_{h_1}^2 + q_{ke} + q_{pk})\tau_{mult})$ where τ_{mult} denotes the cost of a scalar multiplication in $\mathbb{G}_1$.*

PROOF. Algorithm $\mathcal{B}$ takes as input $\langle P, \alpha P, \alpha^2 P, \ldots, \alpha^p P\rangle$ and undertakes to extract $\hat{e}(P,P)^{1/\alpha}$ from its interaction with $\mathcal{A}_I$. W.l.o.g., we can assume that $\mathcal{A}_I$ issues up to $q_{pke} = q_{h_1} - 1 = p - 1$ partial private key extraction queries because, if q_{pke} is strictly less than $q_{h_1} - 1$, the simulator can still issue dummy queries to the partial key generation oracle on its own.

We distinguish the same two kinds of Type I adversaries as in the proof of lemma 4.1:

Type I-A adversaries: replace the public key for the target identity ID^* at some point (and cannot ask for the corresponding partial private key).

Type I-B adversaries: rather decide to ask for the partial private key of the target identity ID^* at some moment.

Before the initialization phase, $\mathcal{B}$ has to guess which kind of Type I adversary $\mathcal{A}_I$ will be. It thus chooses a random bit $c \stackrel{R}{\leftarrow} \{0,1\}$. If $c = 0$, $\mathcal{B}$ bets on a Type I-A attack from $\mathcal{A}_I$ whereas it hopes that $\mathcal{A}_I$ will behave as a Type I-B adversary if $c = 1$. It also selects an index $\ell \stackrel{R}{\leftarrow} \{1, \ldots, q_{h_1}\}$, elements $I_\ell \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and $w_1, \ldots, w_{\ell-1}, w_{\ell+1} \ldots, w_{q_{h_1}} \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$. For $i = 1, \ldots, \ell-1, \ell+1, \ldots, q_{h_1}$, it computes $I_i = I_\ell - w_i$. Depending on the value of c, the setup phase is performed differently:

Case $c = 0$: $\mathcal{B}$ uses its input to compute a generator $H \in \mathbb{G}_1$ and a KGC's public key $P_{pub} := xH$, for some $x \in \mathbb{Z}_q^*$, such that it knows all of the q_{pke} pairs $(I_i, (1/(I_i + x))H)$ for $i \neq \ell$ as in the proof technique of [37]. To do so, $\mathcal{B}$ expands the polynomial $f(z) = \prod_{i=0, i\neq\ell}^{p}(z + w_i) = \sum_{j=0}^{p-1} c_j z^j \in \mathbb{Z}_q[z]$ to obtain the coefficients. A generator $H \in \mathbb{G}_1$ and another group element $U = \alpha H \in \mathbb{G}_1$ are then obtained as

$$\begin{aligned} H &= \sum_{j=0}^{p-1} c_j(\alpha^j P) = f(\alpha)P \\ U &= \sum_{j=1}^{p} c_{j-1}(\alpha^j P) = \alpha f(\alpha)P = \alpha H. \end{aligned}$$

As in [37], $q_{pke} = p - 1$ pairs $\big(w_i, H_i = (1/(w_i + \alpha))H\big)$ are obtained by expanding $f_i(z) = f(z)/(z + w_i) = \sum_{j=0}^{p-2} d_j z^j$ for

$i \in \{1, \ldots, p\} \backslash \{\ell\}$ and computing

$$\begin{aligned} H_i &= \sum_{j=0}^{p-2} d_j(\alpha^j P) = f_i(\alpha)P \\ &= \big(f(\alpha)/(\alpha + w_i)\big)P = \big(1/(\alpha + w_i)\big)H. \end{aligned}$$

The KGC's public key P_{pub} is chosen as

$$P_{pub} = -U - I_\ell H = (-\alpha - I_\ell)H$$

so that its (unknown) private key is implicitly set to $x = -\alpha - I_\ell \in \mathbb{Z}_q^*$. For all $i \in \{1, \ldots, p\} \backslash \{\ell\}$, we have $(I_i, -H_i) = \big(I_i, (1/(I_i + x))H\big)$. The attacker $\mathcal{A}_I$ then receives public parameters including a generator H and the master public key P_{pub}.

Case $c = 1$: $\mathcal{B}$ simply picks a random $x \xleftarrow{R} \mathbb{Z}_q^*$ and starts $\mathcal{A}$ with public parameters params containing the generator $H = \alpha^p P \in \mathbb{G}_1$ and $P_{pub} = xH = x(\alpha^p P) \in \mathbb{G}_1$ as a KGC's public key. By doing so, $\mathcal{B}$ knows the KGC's master key and is able to answer any subsequent partial key extraction query.

As in lemma 4.1, we define $\mathcal{H}$ as the event that $\mathcal{A}_I$ chooses to be challenged on the target identity ID_ℓ. We call $\mathcal{F}_0$ the event that she extracts the partial private key for ID_ℓ and we let $\mathcal{F}_1$ denote the event that she replaces the public key of entity ID_ℓ at some point of the attack.

As in the proof techniques of [5, 6, 7] and lemma 4.1, $\mathcal{B}$ uses a mixed-strategy which is roughly the following. If $c = 0$ and events $\mathcal{F}_0$ and $\mathcal{H}$ occur, $\mathcal{B}$ will have to abort (as it will have failed in guessing which kind of Type I adversary is played by $\mathcal{A}_I$) exactly as if $\mathcal{F}_1$ and $\mathcal{H}$ both occur while $c = 1$. On the other hand, a private key extraction query on the identity ID_ℓ also leads $\mathcal{B}$ to a failure state whereas if either $\neg\mathcal{F}_0 \wedge \mathcal{H}$ or $\neg\mathcal{F}_1 \wedge \mathcal{H}$ occurs, $\mathcal{B}$'s success probability will be related to that of $\mathcal{A}_I$.

The simulator $\mathcal{B}$ then initializes a counter ν to 1 and starts $\mathcal{A}_I$ on the input params. Throughout the game, adversarial queries are answered as follows (we assume that all h_1 queries are distinct and that $\mathcal{A}_I$ produces her attack on an identity ID^* for which she asks for the hash value $h_1(\mathsf{ID}^*)$):

- random oracle queries on h_1 (we call ID_ν the input of the ν^{th} query): $\mathcal{B}$ answers I_ν and increments ν.

- random oracle queries on h_2: for all such queries on inputs $\gamma_{0,l}||\gamma_{1,l}$, for $l = 1, \ldots, q_{h_2}$, $\mathcal{B}$ returns a randomly sampled element $h_{2,l}$ from $\{0,1\}^n$ and stores $(\gamma_{0,l}, \gamma_{1,l}, h_{2,l})$ into a list L_2. If the same query is asked a second time, $\mathcal{B}$ of course responds with the previously defined value.

- queries to Public-Key-Broadcast on input ID_ν (i.e. the ν^{th} identity submitted in a h_1-query): $\mathcal{B}$ first looks into the list L_{pub} to check whether a public key was previously assigned to entity ID_ν. If so, the latter key is returned. Otherwise,
  - in the case $c = 0$: $\mathcal{B}$ picks $l_\nu \xleftarrow{R} \mathbb{Z}_q^*$ and responds with the public key $y_\nu = \hat{e}(H, H)^{l_\nu}$.
  - in the case $c = 1$: if $\nu = \ell$, $\mathcal{B}$ returns the public key $y_\ell = \hat{e}(P, P)$ which equals $\hat{e}(H, H)^{\frac{1}{\alpha(2p)}}$. If $\nu \neq \ell$, $\mathcal{B}$ responds with $y_\nu = \hat{e}(H, H)^{l_\nu}$, for some $l_\nu \xleftarrow{R} \mathbb{Z}_q^*$, as in the situation $c = 0$.

  In both cases a triple $(\mathsf{ID}_\nu, l_\nu, y_\nu)$ is stored in L_{pub} (if $\nu = \ell$, l_ν is unknown in the case $c = 1$).

- queries to Partial-Private-Key-Extract on an input ID_ν:
  - in the case $c = 0$: if $\nu = \ell$, then $\mathcal{B}$ halts and declares "failure". Otherwise, it knows that $h_1(\mathsf{ID}_\nu) = I_\nu$ and returns $-H_\nu = \big(1/(I_\nu + x)\big)H \in \mathbb{G}_1$.
  - in the case $c = 1$, $\mathcal{B}$ behaves as specified by the partial key generation algorithm as it knows the master key.

- queries to Private-Key-Extract on an input ID_ν: if $\nu = \ell$, $\mathcal{B}$ aborts. Otherwise, from the specification of the Public-Key-Broadcast simulator, it comes that $\mathcal{B}$ necessarily knows the associated secret value l_ν (recall that such a query is only authorized if entity ID_ν's public key was not replaced).
  - if $c = 0$, $\mathcal{B}$ additionally knows that the partial private key $-H_\nu$ and can thus compute the full private key $(l_\nu, -H_\nu)$.
  - if $c = 1$, $\mathcal{B}$ extracts the partial private key itself (using the master secret x) and then the full private key using l_ν.

- queries to Public-Key-Replace for some identity ID_ν and public key $\mathsf{pk}' = y'_\nu$: if $\nu = \ell$ and $c = 1$, $\mathcal{B}$ halts and declares "failure". Otherwise, it ensures that pk' is a $\mathbb{G}_2$ element and, if so, replaces the corresponding triple of L_{pub} with $(\mathsf{ID}_\nu, ?, y_\nu)$ (where ? stands for an unknown element of $\mathbb{Z}_q^*$).

When $\mathcal{A}_I$ decides that the first phase is over, she outputs a target identity ID^* together with messages $m_0, m_1 \in \{0,1\}^n$. If $\mathsf{ID}^* \neq \mathsf{ID}_\ell$, $\mathcal{B}$ fails. Otherwise, two distinct strategies are used to build a challenge ciphertext for the entity ID_ℓ and the current public key $\mathsf{pk}^* = y^*$.

Case $c = 0$ **:** $\mathcal{B}$ draws $\sigma \xleftarrow{R} \mathbb{Z}_q^*$ and a random string $c_2^* \xleftarrow{R} \{0,1\}^n$ to return the ciphertext $C^* = \langle c_1^*, c_2^* \rangle$ where $c_1^* = -\sigma H \in \mathbb{G}_1$. If we define $\rho = \sigma/\alpha$ (α being the unknown element defining $\mathcal{B}$'s input) and since $x = -\alpha - I_\ell$, we can check that

$$c_1^* = -\sigma H = -\alpha\rho H = (I_\ell + x)\rho H = \rho I_\ell H + \rho P_{pub},$$

so that c_1^* appears as the first part of a ciphertext for the randomness $\rho \in \mathbb{Z}_q^*$. Realizing that c_2^* is not a scrambling of m_0 nor m_1 would require $\mathcal{A}_I$ to ask for the hash value $h_2(\hat{e}(H,H)^\rho || y^{*\rho})$ and such an event would provide $\mathcal{B}$ with the searched p-BDHI solution as we will see.

Case $c = 1$ **:** $\mathcal{B}$ picks $\lambda \xleftarrow{R} \mathbb{Z}_q^*$ and $c_2^* \xleftarrow{R} \{0,1\}^n$ to return $C^* = \langle c_1^*, c_2^* \rangle$ where

$$c_1^* = \lambda\big(I_\ell(\alpha^{p-1}P) + x(\alpha^{p-1}P)\big) = \lambda\alpha^{-1}(I_\ell H + P_{pub})$$

where $I_\ell = h_1(\mathsf{ID}_\ell)$. Without issuing a h_2-query on the input $(\hat{e}(H,H)^{\lambda\alpha^{-1}} || \hat{e}(P,P)^{\lambda\alpha^{-1}})$, $\mathcal{A}_I$ is unable to recognize that C^* is not an encryption of m_0 nor m_1 and such an event would obviously provide $\mathcal{B}$ with the solution to the p-BDHI problem.

In the second phase, $\mathcal{B}$ simulates $\mathcal{A}_I$'s environment exactly as in phase 1. $\mathcal{A}_I$ finally ends the game by outputting a bit b' that is ignored. With standard arguments, it can be shown that a successful attacker $\mathcal{A}_I$ is very likely to query the h_2 oracle on a relevant input γ (which is $\hat{e}(H,H)^\rho$ if $c = 0$ and $\hat{e}(P,P)^{\lambda\alpha^{-1}}$ if $c = 1$) at some moment of the simulation if the latter is indistinguishable from a real attack environment.

To produce a result, $\mathcal{B}$ selects a random entry $\langle \gamma_0, \gamma_1, . \rangle$ from the list

L_2 so that, with probability $1/q_{h_2}$, γ_0 or γ_1 is the relevant $\mathbb{G}_2$ element. Again, two cases can be distinguished:

- if $c = 0$, we have $\gamma_0 = \hat{e}(H, H)^{\rho} = \hat{e}(P, P)^{f(\alpha)^2\sigma/\alpha}$ and $\mathcal{B}$ can extract the p-BDHI solution by noting that, if $\gamma^* = \hat{e}(P, P)^{1/\alpha}$, then

$$\hat{e}(H, H)^{1/\alpha} = \gamma^{*(c_0^2)}\hat{e}\big(\sum_{i=0}^{p-2} c_i(\alpha^i P), c_0 P\big)\hat{e}\big(H, \sum_{j=0}^{p-2} c_{j+1}(\alpha^j)P\big).$$

- if $c = 1$, $\gamma_1 = \hat{e}(P, P)^{\lambda/\alpha}$ and $\gamma^* = \gamma_2^{1/\lambda}$ is the p-BDHI solution.

We can now assess $\mathcal{B}$'s advantage by a similar reasoning to lemma 2 of [6]: $\mathcal{B}$ reaches a failure state in the following situations:

0. For $i = 0, 1$, if $\mathcal{F}_i$ occurs while $c = i$. We call these events $\mathcal{H}_i$.
1. Because of a private key exposure query for the identity ID_ℓ. We let $\mathcal{F}_2$ denote this event.
2. Or because $\mathcal{A}$ chooses a target identity $\mathsf{ID}^* \neq \mathsf{ID}_\ell$. This corresponds to the event $\neg\mathcal{H}$.

As in [6], event $\mathcal{H}$ implies $\neg\mathcal{F}_2$ so that

$$\begin{aligned}
\Pr[\mathcal{B} \text{ does not abort}] &= \Pr[\neg\mathcal{H}_0 \wedge \neg\mathcal{H}_1 \wedge \mathcal{H}] = \Pr[\neg\mathcal{H}_0 \wedge \neg\mathcal{H}_1|\mathcal{H}]\Pr[\mathcal{H}] \\
&= \frac{1}{q_{h_1}}\Pr[\neg\mathcal{H}_0 \wedge \neg\mathcal{H}_1|\mathcal{H}] \\
&= \frac{1}{q_{h_1}}(1 - \Pr[\mathcal{H}_0|\mathcal{H}] - \Pr[\mathcal{H}_1|\mathcal{H}])
\end{aligned}$$

where the last equality follows from the fact that $\Pr[\mathcal{H}_0 \wedge \mathcal{H}_1|\mathcal{H}] = 0$. On the other hand, as in [6], we have

$$\Pr[\mathcal{H}_i|\mathcal{H}] = \Pr[(c = i) \wedge \mathcal{F}_i|\mathcal{H}] = \frac{1}{2}\Pr[\mathcal{F}_i|\mathcal{H}]$$

since the event $\mathcal{F}_i|\mathcal{H}$ is independent from the event $(c = i)$. Finally, as $\Pr[\mathcal{F}_0 \wedge \mathcal{F}_1|\mathcal{H}] = 0$, we have $\Pr[\mathcal{F}_0|\mathcal{H}] + \Pr[\mathcal{F}_1|\mathcal{H}] \leq 1$ and it comes that

$$\Pr[\mathcal{B} \text{ does not abort}] \geq \frac{1}{2q_{h_1}}.$$

On the other hand we saw that if $\mathcal{B}$ does not fail, it solves the p-BDHI problem with probability $1/q_{h_2}$ (whatever is the value of c). A lower bound on its advantage is then given by $\epsilon' > \epsilon/(2q_{h_1}q_{h_2})$.

Its running time is dominated by $O(q_{h_1}^2)$ operations in the preparation phase and in the solution extraction phase (where 2 pairings must

also be computed), q_{pk} scalar multiplications in $\mathbb{G}_1$ to answer public key broadcast queries and again q_{ke} scalar multiplications to respond to full key extraction queries.

□

Theorem 4.4 *Assume that a Type II IND-CCA attacker $\mathcal{A}_{II}$ has an advantage ϵ against* NewFullCLE *when running a time τ, making q_{h_i} queries to random oracles h_i ($i = 1, 2, 3$), q_D decryption queries, q_{pk} public key broadcast queries, q_{ke} full private key extraction queries. Then there exists an algorithm $\mathcal{B}$ to solve the 1-BDHI problem with a probability*

$$\epsilon' > \frac{1}{e(q_{ke}+1)(q_{h_2}+q_{h_3})}(\epsilon - q_{h_3}/2^{k_0-1})(1-2^{-k})^{q_D}$$

within a time $\tau' < \tau + O((q_{h_3} + q_{ke} + q_{pk} + q_D)\tau_{exp})$ where e is the base for the natural logarithm and τ_{exp} denotes the maximum time to perform a scalar multiplication in $\mathbb{G}_1$ and an exponentiation in $\mathbb{G}_2$.

The proof again applies lemma 4.3. Lemma 4.6 then just needs to show that a chosen plaintext (or IND-CPA) attacker of Type I against the simplified NewBasicCLE scheme allows solving the the 1-BDHI problem (i.e. the p-BDHI problem with $p = 1$).

Lemma 4.6 *Let us assume that a Type II IND-CPA adversary $\mathcal{A}_{II}$ has an advantage ϵ against* NewBasicCLE *when running in a time τ, asking q_{h_i} queries to random oracles h_i ($i = 1, 2$), q_{ke} private key extraction queries and q_{pk} public key broadcast queries. Then there is an algorithm $\mathcal{B}$ to solve the 1-BDHI problem with advantage $\epsilon' > \epsilon/(e(q_{ke}+1)q_{h_2})$ within a time $\tau' < \tau + O((q_{ke} + q_{pk})\tau_{mult})$ where e is the base for the natural logarithm and τ_{mult} is the cost of a scalar multiplication in $\mathbb{G}_1$.*

Proof. We start by describing the algorithm $\mathcal{B}$ taking as input $\langle P, \alpha P\rangle$ and using $\mathcal{A}_{II}$ as a subroutine to compute $\hat{e}(P,P)^{1/\alpha}$.

$\mathcal{B}$ first chooses a generator $H = \alpha P$ and generates the KGC's key pair $(P_{pub}, \mathsf{mk}) = (sH, s) \in \mathbb{G}_1 \times \mathbb{Z}_q^*$ itself. The adversary $\mathcal{A}$ is launched with public parameters including the KGC's master key $\mathsf{mk} = s$.

In the find stage, $\mathcal{A}_{II}$ performs a polynomially bounded number of queries to random oracles h_1 and h_2, to the public key broadcast oracle and to the private key extraction oracle.

- random oracle queries are answered in a standard fashion by uniformly choosing a random element from the appropriate

range and storing the inputs/outputs of these queries into lists L_1 and L_2.

- queries to Public-Key-Broadcast on input ID_i (we call ID_i the input of the i^{th} public key broadcast query): to answer such a query, $\mathcal{B}$ draws $\ell_i \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$. According to Coron's proof technique [68], it then returns the public key $y_i = \hat{e}(H,H)^{\ell_i}$ with probability $q_{ke}/(q_{ke}+1)$ and the public key $y_i = \hat{e}(P,P)^{\ell_i}$ with probability $1/(q_{ke}+1)$.
- queries to Partial-Private-Key-Extract on an input ID: we assume that ID was previously submitted to Public-Key-Broadcast (otherwise, $\mathcal{B}$ can still make the latter query for itself). If the corresponding public key y was set to $\hat{e}(H,H)^{\ell}$ for some known ℓ, $\mathcal{B}$ responds with the pair $(\ell, (1/(s+h_1(\mathsf{ID})))H)$. Otherwise (i.e. if y was set as an unknown power of $\hat{e}(H,H)$), $\mathcal{B}$ aborts.

At the challenge step, $\mathcal{A}_{II}$ outputs messages (m_0, m_1) and a target identity ID^*. At that point, if the corresponding public key was not fixed as $y^* = \hat{e}(P,P)^{\ell^*}$ for some known $\ell^* \in \mathbb{Z}_q^*$, $\mathcal{B}$ aborts. Otherwise, it picks $\mu \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and $c_2^* \stackrel{R}{\leftarrow} \{0,1\}^n$ and returns $C^* = \langle c_1^*, c_2^* \rangle$, where

$$c_1^* = \mu(H_1(\mathsf{ID}^*)P + sP) = \mu\alpha^{-1}(H_1(\mathsf{ID}^*)H + P_{pub}).$$

Unless asking oracle h_2 for the hash value of $(\gamma_0||\gamma_1) = (.||, \hat{e}(P,P)^{\ell^*\mu\alpha^{-1}})$, $\mathcal{A}_{II}$ is unable to recognize that C^* is not an encryption of m_0 nor m_1 and such an event would obviously provide $\mathcal{B}$ with the solution to the 1-BDHI problem.

On the other hand, with standard arguments, we can show that, if the simulator perfectly emulates a genuine attack environment and since $\mathcal{A}_{II}$ is assumed to be a distinguisher of advantage ϵ, she queries the h_2 oracle on the relevant $\mathbb{G}_2$ element γ at some moment of the game with probability ϵ. At the end of the latter, she halts and produces a result that is ignored by $\mathcal{B}$ that randomly selects an element from list L_2 and extracts the $\mathbb{G}_2$ element that was the input of the corresponding h_2-query. With a probability $1/q_{h_2}$, this element is $\gamma_1 = \hat{e}(P,P)^{\ell^*\mu\alpha^{-1}}$ and $\mathcal{B}$ then outputs $\gamma_1^{\frac{1}{\mu\ell^*}}$ which is the searched 1-BDHI solution.

In an analysis of $\mathcal{B}$'s advantage, its probability not to fail in answering a private key extraction query is at least $(1-1/(q_{ke}+1))^{q_{ke}} > 1/e$ while its probability not to abort in the challenge phase is $1/(q_{ke}+1)$.

On the other hand, its running time is dominated by $2q_{pk}$ scalar multiplications to answer public key broadcast queries and q_{ke} other multiplications to respond to private key extraction queries. □

The combination of lemmas 4.6 and 4.3 establishes the result claimed by theorem 4.4.

6. Conclusion

This chapter showed that it was not trivial to generically construct a certificateless cryptosystem which is secure in the strongest model by combining a secure identity based encryption scheme with a traditional public key cryptosystem. It pinpointed security problems in three simple generic constructions and fixed them using a generic conversion (inspired from the Fujisaki-Okamoto transformation) ensuring the security in the strong model given any scheme only withstanding chosen-plaintext attacks. We finally described a new scheme offering computational advantages over previous pairing-based constructions.

Part 3

Identity-Based Signatures and Signcryption Schemes

CHAPTER 5

Identity-Based Signatures

Abstract. This chapter contains two results about identity based signature (IBS) schemes. The first one is a new construction based on bilinear maps that is more efficient than all previous ones as its verification algorithm only requires to compute one pairing. We give a security proof for this new scheme under the p-Diffie-Hellman Inversion (p-DHI) assumption that has been recently used by several papers in the literature. The second result shows a new security proof for a scheme that was previously proven secure by Bellare, Namprempre and Neven. The new proof features a tighter security reduction in the random oracle model than any other known IBS.

1. Related work on IBS schemes

As previously mentioned in this thesis, the decade that followed Shamir's seminal paper [195] saw the appearance of several digital signature schemes supporting identity-based public keys. These include the Fiat-Shamir [84] and the Guillou-Quisquater [106] identity-based signatures (IBS) and others [33, 100, 170]. After the famous paper published by Boneh and Franklin [40] that showed how pairings may be practical in the design of identity-based cryptographic schemes, many pairing-based IBS schemes were proposed [190, 174, 111, 51, 219].

Although the concept of identity-based signature is already 21 years old, a formal definition of security for IBS schemes was only considered in 2003 by Cha and Cheon [51] who extended the usual notion of existential unforgeability against chosen-message attacks (EUF-CMA) [105] and proved the security of their scheme in this model. Independently of [51], Dodis, Katz, Xu, and Yung [81] defined a class of standard signature schemes that they call 'trapdoor', and then presented a random-oracle-using transformation turning any secure 'trapdoor standard signature' scheme into an IBS scheme that is provably secure in the model of Cha

and Cheon [51]. Security proofs for several existing IBS schemes, including those of [84, 106, 170, 111] may be obtained by observing that these are the result of applying the transformation of [81] to some underlying trapdoor standard signature already proven secure in the literature.

All such known trapdoor standard signatures have security proofs based on the powerful forking lemma [182, 183] or alternative proof techniques [165, 1] that pertain to signature schemes derived from some honest-verifier zero-knowledge identification scheme using the Fiat-Shamir heuristic [84]. It thus happens that all known IBS schemes can be thought of as being derived from some identity-based identification (IBI) scheme. Somewhat strangely, a provable security treatment of IBI schemes remained lacking until 2004, when two independent works [124, 28] gave formal security models for IBI schemes for the first time. In [124], Kurosawa and Heng showed how to turn a special kind of digital signature schemes into an IBI scheme which is secure against passive attacks (that is, where an adversary has access to an oracle returning transcripts of the interactive protocol but cannot directly play the cheating verifier in a run of the protocol before attempting to impersonate a prover as she could in an active attack).

On the other hand, at Eurocrypt 2004, Bellare, Namprempre and Neven [28] defined a general framework to provide security proofs for a large family of IBS schemes. Their framework proceeds by considering the security against passive, active and concurrent attacks[1] of underlying 'convertible' identifications schemes (i.e. standard identification protocols that can be converted into identity-based ones) or by using the provable security of underlying 'convertible standard signature'[2] schemes. In the same paper, they show that the existential unforgeability of any convertible standard signature in the usual model [105] implies the security of the resulting identity-based signature in the sense of Cha and Cheon [51]. Their framework allowed them to establish the security of several IBS schemes including [195, 33, 85, 166] and a variant of [190], the security status of which was unknown before. Except the scheme of [100]

[1]That is, attacks where an adversary can play the cheating verifier against several concurrent executions of the same prover before attempting to impersonate the latter.

[2]The latter notion characterizes signature schemes that can be turned into an IBS scheme and is a generalization of Dodis et al.'s notion of 'trapdoor signature scheme' [81].

for which they found an attack, their framework implies the security of all known IBS schemes (even a recent one proposed in [60]) except a scheme proposed by Okamoto in [166] and a new one which they proposed themselves as a simplification of [166]. The reason for this is that the schemes of [166, 28] do not seem to derive from any convertible identification scheme and direct security proofs were needed for them.

A contribution of the present chapter is to propose a new provably secure identity-based signature, discovered in the course of a joint work [21] with Paulo Barreto and Noel McCullagh, that does not either fall into the category of schemes to which the framework of [28] applies. Indeed, it can be shown that our scheme does not derive from a convertible scheme unless a recently studied computational assumption [37] is false. The new scheme happens to be more efficient than any previous pairing-based IBS schemes as its verification algorithm requires to compute a single bilinear map whereas all previous proposals [190, 111, 51, 219] require at least two pairing computations for the verifier. This improvement is obtained at the expense of a security resting on the p-Diffie-Hellman Inversion assumption [37, 227] which is stronger than the now well-studied Computational Diffie-Hellman assumption on which the security of the schemes [190, 111, 51, 60] provably relies. Our scheme, called DHI-IBS as a shorthand for "Diffie-Hellman Inversion-based Identity-based Signature", is proved secure using Pointcheval and Stern's forking lemma [182, 183].

The latter lemma is known to only provide loose reductions in the sense that it only allows turning a forger with advantage ϵ into a Turing machine solving a hard problem within a comparable running time with a probability $O(\epsilon^2/q_H)$, where q_H denotes a bound on the number of random oracle queries made by the forger. In the case of DHI-IBS, it leads us to a loose reduction under a Diffie-Hellman related assumption that is potentially stronger than the usual Diffie-Hellman problem. If one is willing to accept the p-Diffie-Hellman Inversion assumption as being reasonable and if one accepts that even a loose but polynomial reduction from it yields sufficient guarantees, one can settle for using DHI-IBS. On the other hand, some people from the research community who are concerned with concrete security might accept to pay a loss of efficiency to obtain stronger security bounds. Indeed, the last couple of years saw the rise of a new trend consisting of providing tight security

reductions for asymmetric cryptosystems (see [31] or [180] for instance). This led several authors to provide new security proofs for systems that were already well known to be secure in the random oracle model or for some of their variants [68, 69, 150]. Some authors even devised new schemes that, although seemingly less efficient than existing ones at first sight, provide much better security guarantees for the same security parameters and are then eventually more efficient for a similar desired level of security [103, 121, 61].

A second contribution of the present chapter is to show that a modification of the Sakai-Ogishi-Kasahara IBS (called SOK-IBS here) that was already proven secure in [28], has a much tighter security proof under the Diffie-Hellman assumption than the bounds given in [28]: we give a new proof in which an attacker with advantage ϵ implies a polynomial time algorithm for the Diffie-Hellman problem with an advantage $O(\epsilon/q_{KE})$, q_{KE} being a bound on the number of identities corrupted by the forger, and we stress that a fully optimal reduction from a potentially stronger but reasonable assumption exists. According to [121], we think that a tight reduction from a given assumption is preferable to a loose reduction w.r.t. a weaker assumption. Prior to this result, Kurosawa and Heng [124] claimed to achieve an improved security result for the Cha-Cheon IBS [51]. They exhibited a reduction from the Diffie-Hellman problem to a chosen-message attacker that is still quite loose: an attacker with a given advantage ϵ is used to build an algorithm to solve the Diffie-Hellman problem with probability $O(\epsilon^2/q_{KE}q_H^2)$ where q_{KE} is the number of identities corrupted by the adversary and q_H the number of hash queries. We believe that this can be improved to $O(\epsilon^2/q_{KE}q_H)$ which remains a looser bound than ours for SOK-IBS.

The chapter is organized as follows. Section 2 recalls the usual formal model of identity-based signatures. Our new scheme is described in section 3.1 and its security proof is given in section 4.1. Our improved security reduction for SOK-IBS is detailed in section 5.

2. Formal definition and security model

We firstly recall the syntax that is commonly used for IBS schemes. Definition 5.2 formalizes a security notion for IBS schemes that was considered in [51, 81, 28] as an extension of the usual notion of existential unforgeability under chosen-message attacks [105].

Definition 5.1 *An identity-based signature (IBS) scheme is a* 4*-uple of algorithms which are the following ones:*

Setup: *is a probabilistic algorithm run by a PKG that takes as input a security parameter to output a public/private key pair* (P_{pub}, mk) *for the PKG (*P_{pub} *is its public key and* mk *is its master key that is kept secret).*

Keygen: *is a key generation algorithm run by a PKG. It takes as input the PKG's master key* mk *and a user's identity* ID *to return the user's private key* d_{ID}.

Sign: *given a message* M*, the PKG's public key and a private key* d_{ID}*, this algorithm generates a signature* σ *on* M.

Verify: *is a deterministic verification algorithm that, given an alleged signature* σ *on a message* M *for an identity* ID*, outputs* 1 *or* 0 *depending on whether the signature is accepted or not.*

Definition 5.2 ([51]) *An IBS scheme is said to be* ***existentially unforgeable*** *under adaptive chosen message and identity attacks if no PPT adversary has a non-negligible advantage in this game:*

1. *The challenger runs the setup algorithm to generate the system's parameters and sends them to the adversary.*
2. *The adversary* $\mathcal{F}$ *performs a series of queries:*
   - *Key extraction queries:* $\mathcal{F}$ *produces an identity* ID *and receives the private key* S_{ID} *corresponding to* ID.
   - *Signature queries:* $\mathcal{F}$ *produces a message* M *and an identity* ID *and receives a signature on* M *that was generated by the signature oracle using the private key corresponding to the identity* ID.
3. $\mathcal{F}$ *eventually produces a triple* $(\mathsf{ID}^*, M^*, \sigma^*)$ *made of an identity* ID^**, whose corresponding private key was never asked during stage 2, and a message-signature pair* (M^*, σ^*) *such that* (M^*, ID^*) *was never submitted to the signature oracle.*

$\mathcal{F}$ *wins if the verification algorithm accepts the triple* $(\mathsf{ID}^*, M^*, \sigma^*)$*. Her advantage is defined to be her probability of victory taken over her cointosses and the challenger's ones.*

3. A faster identity-based signature from bilinear maps

Our new scheme is depicted in figure 5.1. Its security relies on the hardness of the following problem that was introduced in [227].

Definition 5.3 ([37, 227]) *The* **p-*Diffie-Hellman Inversion Problem*** *(p-DHIP) in ($\mathbb{G}_1$,$\mathbb{G}_2$) is, given $(P, Q, \alpha Q, \alpha^2 Q, \ldots, \alpha^p Q)$ as input, to compute $\frac{1}{\alpha} P$.*

The intractability of the latter problem for any PPT algorithm will be referred to as the p-Diffie-Hellman Inversion assumption. At Eurocrypt'04, Boneh and Boyen [37] introduced a related stronger assumption called p-Strong Diffie-Hellman (p-SDH) assumption which consists in finding a pair $(c, \frac{1}{c+\alpha} P) \in \mathbb{Z}_q \times \mathbb{G}_1$ given the same inputs.

3.1. The scheme

Setup: given k, the PKG chooses a large prime $p > 2^k$, asymmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ of order p and an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$. It then selects generators $(P, Q) \in \mathbb{G}_1 \times \mathbb{G}_2$ with $P = \psi(Q)$ and a master key $s \xleftarrow{R} \mathbb{Z}_p^*$ to compute the system-wide public key $Q_{pub} = sQ \in \mathbb{G}_2$. Finally, it chooses hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$, $H_2 : \{0,1\}^* \times \mathbb{G}_T \to \mathbb{Z}_q^*$. The public parameters are

$$\mathsf{params} := \{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P, Q, Q_{pub}, e, \psi, H_1, H_2\}.$$

Keygen: given a user's identity ID, the PKG computes the associated private key $d_{\mathsf{ID}} = \frac{1}{H_1(\mathsf{ID})+s} P$.

Sign: in order to sign a message $M \in \{0,1\}^*$, the signer does the following:

(1) Pick a random $x \xleftarrow{R} \mathbb{Z}_q^*$, compute $r = e(P, Q)^x$.
(2) Set $h = H_2(M, r) \in \mathbb{Z}_q^*$.
(3) Compute $S = (x + h) d_{\mathsf{ID}}$.

The signature on M is $\sigma = (h, S) \in \mathbb{Z}_q^* \times \mathbb{G}_1$.

Verify: a signature $\sigma = (h, S)$ on a message M is accepted if

$$h = H_2(M, e(S, Q_{\mathsf{ID}}) e(P, Q)^{-h})$$

where $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) Q + Q_{pub}$.

FIGURE 5.1. The DHI-IBS scheme

The method for obtaining private keys from identities is a simplification of a method suggested by Sakai and Kasahara [189]. The scheme

can be thought of as a non-interactive *proof of knowledge* of a digital signature computed using an algorithm discussed in two independent papers [37, 227].

At Eurocrypt'04, Bellare, Namprempre and Neven established a framework [28] to prove the security of a large family of identity-based signatures and they only found two schemes to which their framework does not apply. The present one does not either fall into the category of schemes to which it applies. Indeed, it can be shown that our IBS does not result from the transformation of any convertible standard identification or signature scheme (in the sense of [28]) unless the p-SDH problem [37] is easy. A direct security proof is thus needed.

4. Comparison with a related scheme

Setup and **Keygen** are the same as in our scheme. The system-wide parameters are

$$\mathsf{params} := \{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P, Q, Q_{pub}, e, \psi, H_1, H_2\}.$$

Sign: to sign a message $M \in \{0,1\}^*$, the signer does the following:

(1) Pick $x \overset{R}{\leftarrow} \mathbb{Z}_p^*$ and compute $r = e(P, Q_{\mathsf{ID}})^x \in \mathbb{G}_T$.
(2) Set $h = H_2(M, r) \in \mathbb{Z}_p^*$.
(3) Compute $S = xP + hd_{\mathsf{ID}}$.

The signature on M is $\sigma = (h, S) \in \mathbb{Z}_p^* \times \mathbb{G}_1$.

Verify: a signature $\sigma = (h, S)$ on a message M is accepted iff

$$h = H_2(M, e(S, Q_{\mathsf{ID}})e(P, Q)^{-h}).$$

where $Q_{\mathsf{ID}} = H_1(\mathsf{ID})Q + Q_{pub}$.

FIGURE 5.2. The Kurosawa-Heng IBS scheme

Independently of our work, Kurosawa and Heng [125] described an identity-based identification (IBI) protocol that has a security proof in the standard model. This protocol implicitly suggests another IBS scheme bearing some similarities with DHI-IBS and which can be proved secure under the same assumption. It turns out that our scheme is slightly faster in a signature generation than the Kurosawa-Heng IBS which is here described. We indeed observe that, even if the signing algorithm is optimized by pre-computing $e(P, Q_{\mathsf{ID}})$, it is still slower than

ours in the last step.

Our more efficient DHI-IBS scheme may be extended in the same way into a secure IBI in the standard model.

4.1. Security proof for DHI-IBS

The security proof relies on the forking lemma [182, 183]. As the security model of IBS schemes enables a forger to adaptively choose her target identity, we cannot directly apply the forking technique which is related to signature schemes where an attacker is challenged on a fixed public key chosen by her challenger. We must rather follow the approach of [51] that first considers a weaker attack model where adversaries are challenged on a given identity selected by the challenger. In [51], an IBS scheme is said to be secure against existential forgeries on adaptively chosen message and *given* identity attacks if no adversary has a non-negligible advantage in the weaker model of attack.

Lemma 5.1 ([51]) *If there is a forger $\mathcal{A}_0$ for an adaptively chosen message and identity attack having advantage ϵ_0 against our scheme when running in a time t_0 and making q_{h_1} queries to random oracle h_1, then there exists an algorithm $\mathcal{A}_1$ for an adaptively chosen message and given identity attack which has advantage $\epsilon_1 \geq \epsilon_0(1 - \frac{1}{2^k})/q_{h_1}$ within a running time $t_1 \leq t_0$. Moreover, $\mathcal{A}_1$ asks the same number key extraction queries, signature queries and H_2 queries as $\mathcal{A}_0$ does.*

Lemma 5.2 *Let us assume that there is an adaptively chosen message and given identity attacker $\mathcal{F}$ that makes q_{h_i} queries to random oracles H_i $(i = 1, 2)$ and q_s queries to the signing oracle. Assume that, within time t, $\mathcal{F}$ produces a forgery with probability $\epsilon \geq 10(q_s+1)(q_s+q_{h_2})/2^k$. There exists an algorithm $\mathcal{B}$ that is able to solve the p-DHIP for $p = q_{h_1}$ in expected time*

$$t' \leq 120686 q_{h_2}(t + O(q_s\tau_p))/\epsilon + O(q_{h_1}^2\tau_{mult})$$

where τ_{mult} denotes the cost of a scalar multiplication in $\mathbb{G}_2$ and τ_p is the cost of a pairing evaluation.

Proof. The proof relies on the forking lemma. We first show how to provide the adversary with a consistent view by coherently answering

all of her queries and we then explain how to apply the forking lemma.

Algorithm $\mathcal{B}$ takes as input an instance $(P, Q, \alpha Q, \alpha^2 Q, \ldots, \alpha^p Q)$ of the p-DHI problem in bilinear groups $(\mathbb{G}_1, \mathbb{G}_2)$ and aims at finding $\frac{1}{\alpha}P$. In a preparation phase, $\mathcal{B}$ builds generators $H \in \mathbb{G}_2$, $G = \psi(H) \in \mathbb{G}_1$ and a domain-wide public key $H_{pub} = xH \in \mathbb{G}_2$ (for some unknown element $x \in \mathbb{Z}_q^*$) such that it knows $p-1$ pairs $(I_i, \frac{1}{I_i+x}G)$ for $I_1, I_2, \ldots, I_{p-1} \in_R \mathbb{Z}_q^*$. To do so,

1. It picks random elements $I^* \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and $w_1, w_2, \ldots, w_{p-1} \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and expands the polynomial $f(z) = \prod_{i=1}^{p-1}(z + w_i)$ to obtain coefficients $c_0, \ldots, c_{p-1} \in \mathbb{Z}_q^*$ so that $f(z) = \sum_{i=0}^{p-1} c_i z^i$. For $i = 1, \ldots, p-1$, it also computes $I_i = I^* - w_i \in \mathbb{Z}_q^*$.
2. It sets $H = \sum_{i=0}^{p-1} c_i(\alpha^i Q) = f(\alpha)Q$ as a public generator of $\mathbb{G}_2$ and $G = \psi(H) = f(\alpha)P$ as a generator of $\mathbb{G}_1$. Another group element $H' \in \mathbb{G}_2$ is then set to $H' = \sum_{i=1}^{p} c_{i-1}(\alpha^i Q)$. We note that $H' = \alpha H$ although $\mathcal{B}$ does not know α.
3. For $i = 1, \ldots, p-1$, $\mathcal{B}$ expands $f_i(z) = f(z)/(z + w_i) = \sum_{i=0}^{p-2} d_i z^i$ that satisfy

$$\frac{1}{\alpha + w_i}G = \frac{f(\alpha)}{\alpha + w_i}P = f_i(\alpha)P = \sum_{i=0}^{p-2} d_i \psi(\alpha^i Q)$$

The $p - 1 = q_{h_1} - 1$ pairs $(w_i, G_i = \frac{1}{\alpha+w_i}G)$ are then computed by $\mathcal{B}$ according to the last member of the above equation.

The system-wide public key H_{pub} is chosen as

$$H_{pub} = -H' - I^* H = (-\alpha - I^*)H$$

so that its (unknown) private key is implicitly set to $x = -\alpha - I^* \in \mathbb{Z}_q^*$. For all $i \in \{1, \ldots, p-1\}$, we have $(I_i, -G_i) = (I_i, \frac{1}{I_i+x}G)$.

The simulator $\mathcal{B}$ is then ready to answer $\mathcal{F}$'s queries along the simulation. It first initializes a counter ℓ to 0 and launches $\mathcal{F}$ on the input (H_{pub}, ID^*) for a randomly chosen challenge identity $\mathsf{ID}^* \stackrel{R}{\leftarrow} \{0,1\}^*$.

- H_1 queries: when $\mathcal{F}$ probes oracle H_1 on an identity ID, $\mathcal{B}$ returns I^* if $\mathsf{ID} = \mathsf{ID}^*$. Otherwise, $\mathcal{B}$ increments ℓ by 1 and answers $I_\ell \in \mathbb{Z}_q^*$. In the latter case, the pair $(\mathsf{ID}, -G_\ell)$ is stored in a list L_1.
- Key extraction queries for an identifier $\mathsf{ID} \neq \mathsf{ID}^*$: $\mathcal{B}$ recovers the corresponding pair $(\mathsf{ID}, -G_\ell)$ in L_1 for which $-G_\ell$ was

computed during the preparation phase. The latter element is returned as a private key associated to ID and appears as a valid private key from $\mathcal{F}$'s view.

- Signing query: upon receiving such a query on a message-identity pair (M, ID), $\mathcal{B}$ picks random elements $S \xleftarrow{R} \mathbb{G}_1$, $h \xleftarrow{R} \mathbb{Z}_q^*$, computes

$$r = e(S, Q_{\mathsf{ID}})e(G, H)^{-h},$$

where $Q_{\mathsf{ID}} = H_1(\mathsf{ID})H + H_{pub}$ is computed thanks to a value $H_1(\mathsf{ID})$ recovered from list L_1, and then backpatches to define the value $H_2(M, r)$ as $h \in \mathbb{Z}_q^*$. This simulation is similar to those of all non-interactive honest verifier zero-knowledge proofs ($\mathcal{B}$ of course fails if the hash value $H_2(M, r)$ is already defined but such an event is very unlikely and its probability is taken into account in the bounds given by the forking lemma).

We have explained how to simulate $\mathcal{F}$'s environment in a chosen-message and given identity attack. We are ready to apply the forking lemma that essentially says the following: consider a scheme producing signatures of the form (M, r, h, S), where each of r, h, S corresponds to one of the three moves of an honest-verifier zero-knowledge protocol. Let us assume that a chosen-message attacker $\mathcal{F}$ forges a signature (M, r, h, S) in a time t with probability $\epsilon \geq 10(q_s+1)(q_s+q_h)/2^k$ (k being a security parameter so that h is uniformly taken from a set of 2^k elements) when making q_s signature queries and q_h random oracle calls. If the triples (r, h, S) can be simulated without knowing the private key, then there exists a Turing machine $\mathcal{F}'$ that uses $\mathcal{F}$ to produce two valid signatures (m, r, h_1, S_1), (m, r, h_2, S_2), with $h_1 \neq h_2$, in expected time $t' \leq 120686 q_h t/\epsilon$.

In our setting, from a forger $\mathcal{F}$, we build an algorithm $\mathcal{F}'$ that replays $\mathcal{F}$ a sufficient number of times on the input (H_{pub}, ID^*) to obtain two suitable forgeries $\langle M^*, r, h_1, S_1\rangle$, $\langle M^*, r, h_2, S_2\rangle$ with $h_1 \neq h_2$.

The reduction then works as follows. The simulator $\mathcal{B}$ runs $\mathcal{F}'$ to obtain two forgeries $\langle M^*, r, h_1, S_1\rangle$, $\langle M^*, r, h_2, S_2\rangle$ for the same message M^* and commitment r. If both forgeries satisfy the verification equation, we obtain the relations

$$e(S_1, Q_{\mathsf{ID}^*})e(G, H)^{-h_1} = e(S_2, Q_{\mathsf{ID}^*})e(G, H)^{-h_2},$$

with $Q_{\mathsf{ID}^*} = H_1(\mathsf{ID}^*)H + H_{pub} = (I^* + x)H = -\alpha H$. Then, it comes that

$$e((h_1 - h_2)^{-1}(S_1 - S_2), Q_{\mathsf{ID}^*}) = e(G, H),$$

and hence $T^* = (h_1 - h_2)^{-1}(S_2 - S_1) = \frac{1}{\alpha}G$. From T^*, $\mathcal{B}$ can proceed as in [37] to extract $\sigma^* = \frac{1}{\alpha}P$: it knows that $f(z)/z = c_0/z + \sum_{i=0}^{p-2} c_i z^i$ and eventually computes

$$\sigma^* = \frac{1}{c_0}\left[T^* - \sum_{i=0}^{p-2} c_i \psi(\alpha^i Q)\right] = \frac{1}{\alpha}P$$

which is returned as a result.

It finally comes that, if $\mathcal{F}$ forges a signature in a time t with probability $\epsilon \geq 10(q_s + 1)(q_s + q_{h_2})/2^k$, $\mathcal{B}$ solves the p-DHIP in expected time

$$t' \leq 120686 q_{h_2}(t + O(q_s \tau_p))/\epsilon + O(q_{h_1}^2 \tau_{mult})$$

where the last term accounts for the cost of the preparation phase. □

The combination of the above lemmas yields the following theorem.

Theorem 5.1 *In the random oracle model, let us assume that there exists an adaptively chosen message and identity attacker $\mathcal{A}$ that makes q_{h_i} queries to random oracles H_i ($i = 1, 2$) and q_s queries to the signing oracle. Assume that, within a time t, $\mathcal{A}$ produces a forgery with probability $\epsilon \geq 10(q_s+1)(q_s+q_{h_2})/2^k$. Then, there exists an algorithm $\mathcal{B}$ that is able to solve the p-Diffie-Hellman Inversion Problem for $p = q_{h_1}$ in an expected time*

$$t' \leq 120686 q_{h_1} q_{h_2} t/(\epsilon(1 - 1/2^k)) + O(q_{h_1}^2 \tau)$$

where τ is the cost of a scalar multiplication in $\mathbb{G}_2$.

4.2. Efficiency comparisons

In order to assess the comparative efficiency of several schemes, Paulo Barreto implemented them in accordance with their original descriptions. Table 5.1 summarises the number of relevant basic operations: namely, exponentiations in $\mathbb{G}_T$, scalar point multiplications in $\mathbb{G}_1$, and pairing evaluations. The same table compares the observed processing times (in milliseconds) for a supersingular curve of embedding degree $k = 6$ over $\mathbb{F}_{3^{97}}$, using implementations written in C++ and run on an Athlon XP 2 GHz. Subtleties in the algorithms determine

somewhat different running times even when the operation counts for those algorithms are equal. We see from these results that our system beats the efficiency of the most efficient pairing-based schemes [111, 51], especially in the verification algorithm which is about twice faster than in [51].

	Sign				Verify			
signature scheme	exp	mul	pairings	time (ms)	exp	mul	pairings	time (ms)
Heß[111]	1	2		2.50	1		$2^{\dagger}$	9.37
Cha-Cheon [51]		2		1.88		1	2	9.22
DHI-IBS		2		1.72		1	1	5.00

(†) One pairing is precomputable, incurring for the verifier a storage cost of one $\mathbb{G}_T$ element for each other user in the system, plus one $\mathbb{G}_T$ exponentiation.

TABLE 5.1. Efficiency comparisons with other IBS schemes

4.3. Signatures with partial message-recovery

We note that our IBS scheme can be turned into a signature scheme with partial message recovery using the technique of Zhang et al. [225] who recently proposed a variant of Heß's identity-based signature [111] enabling partial message-recovery using a technique similar to the one put forth by Abe and Okamoto [3] for Schnorr's signature [191].

The latter technique can actually be applied to any Fiat-Shamir like [84] signature scheme for which the signer's commitment (i.e. the quantity that is hashed together with the message and which is r in our scheme) can be recovered from the signature upon verification of the latter. For example, for the Guillou-Quisquater [106] and Fiat-Shamir [84] schemes, it allows sparing an overhead of 80 bits of bandwidth when compared to a message-signature concatenation produced using the usual signing procedures. This technique does not apply to all IBS schemes: for example, the Cha-Cheon [51] and the Sakai-Ogishi-Kasahara [190] schemes do not support it as their commitment cannot be recovered from the signature by verifiers.

In contrast, the recovering of r is actually performed in the verification algorithm of our scheme and it turns out that the partial message recovery technique of [3] can be applied to it.

5. Tighter security reductions for an existing IBS

Although concerned with the provable security of identity-based signatures, the research community did not focus on providing really strong security arguments for the various IBS proposed in the literature up to now. Indeed, Paterson's IBS [174] still has no formal security proof while Cha-Cheon [51] and Heß [111] gave proofs under the Diffie-Hellman assumption for their respective scheme. However, similarly to the security proof of our DHI-IBS scheme, these proofs were both obtained through Pointcheval and Stern's forking lemma [182, 183] which does not yield tight security reductions as mentioned by several previous papers in the literature [103, 121, 62, 150].

Setup: given a security parameter k, the PKG chooses symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of prime order $q > 2^k$, a generator P of $\mathbb{G}_1$, a master key $s \xleftarrow{R} \mathbb{Z}_q^*$ and the associated public key $P_{pub} = sP$. It also picks cryptographic hash functions of identical domain and range $H_1, H_2 : \{0,1\}^* \to \mathbb{G}_1^*$. The public parameters are

$$\texttt{params} := \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, P_{pub}, H_1, H_2\}.$$

Keygen: given a user's identity ID, the PKG computes $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in \mathbb{G}_1$ and the associated private key $d_{\mathsf{ID}} = sQ_{\mathsf{ID}} \in \mathbb{G}_1$.

Sign: in order to sign a message M,

(1) Pick $r \xleftarrow{R} \mathbb{Z}_q$, compute $U = rP$ and $H = H_2(\mathsf{ID}, M, U) \in \mathbb{G}_1$.
(2) Compute $V = d_{\mathsf{ID}} + rH \in \mathbb{G}_1$.

The signature on M is the pair $\sigma = \langle U, V \rangle \in \mathbb{G}_1 \times \mathbb{G}_1$.

Verify: given a signature $\sigma = \langle U, V \rangle$ on a message M for an identity ID, the verifier computes $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in \mathbb{G}_1$ and $H = H_2(\mathsf{ID}, M, U) \in \mathbb{G}_1$. The signature is accepted if $\hat{e}(P, V) = \hat{e}(P_{pub}, Q_{\mathsf{ID}})\hat{e}(U, H)$ and rejected otherwise.

FIGURE 5.3. The SOK-IBS scheme

In this section we show that Bellare et al.'s [28] modification of the Sakai-Ogishi-Kasahara IBS [190], which is described in figure 5.3, has a much tighter security reduction under the Diffie-Hellman assumption than any other known IBS. We point out that the latter actually corresponds to a one level instantiation of a (randomized) version of Gentry and Silverberg's alternative hierarchical IBS [99]. This scheme is the

same as the one obtained by applying Bellare et al.'s extended Fiat-Shamir heuristic [28] to the SOK identity-based identification scheme.

Although derived from an IBI that is only secure against passive adversaries (as shown in [28]), the modified SOK-IBS has better reductions than other IBS schemes for which the underlying IBI is secure against stronger attacks. Somewhat surprisingly, tighter security reductions can be obtained for the same reason as the one for which the underlying IBI scheme does not resist active attacks. Indeed, in the security proof, the simulator actually mimics the behavior of an active attacker against the IBI scheme through random oracle manipulations in order to perform an online extraction of a Diffie-Hellman solution: the adversary does not have to be rewinded in accordance with the forking technique unlike what happens in security proofs of other known IBS.

From an efficiency point of view, the signature issuing algorithm has almost the same complexity as Cha and Cheon's one [51] while the verification algorithm requires an additional pairing computation.

5.1. A new proof for SOK-IBS

This security analysis first presents a security reduction from the Diffie-Hellman problem to a chosen-message attacker against SOK-IBS that is more efficient than any other known security reduction (including those given in [124],[28]) for existing identity-based signatures ([51],[111],etc.). In a second step, we explain how to achieve an essentially optimal reduction from another Diffie-Hellman related assumption.

Theorem 5.2 *If a forger $\mathcal{F}$ has an advantage ϵ over SOK-IBS when running in a time t and asking q_{H_i} queries to random oracles H_i (i=1,2), q_{KE} private key extraction queries and q_S signing queries, then the CDH problem can be solved with an advantage*

$$\epsilon' > \frac{1}{e(q_{KE}+1)}\Big(\epsilon - \frac{1}{2^k}\big(q_S(q_{H_2}+q_S)+1\big)\Big)$$

within a time $t' < t + O((q_{H_1}+q_{H_2}+q_{KE}+q_S)t_m)$ where e is the base of the natural logarithm and t_m the cost of a scalar multiplication in $\mathbb{G}_1$.

Proof. We describe how a forger $\mathcal{F}$ can be used by a PPT algorithm $\mathcal{B}$ to solve the CDH problem. Let $(X = xP, Y = yP) \in \mathbb{G}_1 \times \mathbb{G}_1$ be a random instance of the CDH problem taken as input by $\mathcal{B}$. The latter

initializes $\mathcal{F}$ with $P_{pub} = X$ as a system's overall public key. The forger $\mathcal{F}$ then starts performing queries such as those described in definition 5.2. These queries are answered by $\mathcal{B}$ as follows (we still assume that any private key generation query or signature query involving an identity comes after an H_1-query for the same identity):

- H_1-queries: when an identity ID is submitted to the H_1 oracle, as in Coron's proof technique [68], $\mathcal{B}$ first picks $u \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$. It then returns $uP \in \mathbb{G}_1$ with probability $q_{KE}/(q_{KE}+1)$ and $uY \in \mathbb{G}_1$ with probability $1/(q_{KE}+1)$.
- Key extraction queries: when $\mathcal{F}$ asks for the private key associated to an identity ID, $\mathcal{B}$ returns $uP_{pub} = uX \in \mathbb{G}_1$ if $H_1(\mathsf{ID}^*)$ was fixed to $uP \in \mathbb{G}_1$. Otherwise, $\mathcal{B}$ outputs "failure" and halts because it is unable to coherently answer the query.
- H_2-queries: when a tuple (ID, M, U) is submitted to oracle H_2, $\mathcal{B}$ first scans a list L_2 to check whether H_2 was previously defined for that input. If yes, the defined value is returned. Otherwise, $\mathcal{B}$ picks a random $v \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$, stores the tuple (ID, M, U, v) in list L_2 and returns $vP \in \mathbb{G}_1$.
- Signature queries on a message M for an identity ID: $\mathcal{F}$ first recovers the previously defined value $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in \mathbb{G}_1$ from L_1. It then chooses $t, \nu \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ before setting $V = tP_{pub} = tX$, $U = \nu P_{pub} = \nu X$ and defining $H_2(\mathsf{ID}, M, U)$ as $\nu^{-1}(tP - Q_{\mathsf{ID}}) \in \mathbb{G}_1$ ($\mathcal{B}$ halts and declares "failure" if H_2 is already defined for the input (ID, M, U)). The pair $\langle U, V \rangle$ is returned to $\mathcal{F}$ and appears as a valid signature from the latter's point of view.

Eventually, $\mathcal{F}$ produces a message M^*, an identity ID^* and a forgery $\langle U^*, V^* \rangle$ for the pair (M^*, ID^*). If $H_1(\mathsf{ID}^*)$ was not fixed to a multiple u^*Y of Y, for some known value $u^* \in \mathbb{Z}_q^*$, then $\mathcal{B}$ outputs "failure". Otherwise, the list L_2 must contain a record $(\mathsf{ID}^*, M^*, U^*, v^*)$ with overwhelming probability (otherwise, $\mathcal{B}$ stops and reports "failure"). Hence, since $H^* = H_2(\mathsf{ID}^*, M^*, U^*)$ was defined to be $v^*P \in \mathbb{G}_1$, $\mathcal{B}$ knows that

$$\hat{e}(P, V^*) = \hat{e}(X, Q_{\mathsf{ID}^*})\hat{e}(U^*, H^*)$$

with $H^* = v^*P \in \mathbb{G}_1$ and $Q_{\mathsf{ID}^*} = u^*Y \in \mathbb{G}_1$ for some known elements $u^*, v^* \in \mathbb{Z}_q^*$. Then, it also knows that

$$\hat{e}(P, V^* - v^*U^*) = \hat{e}(X, u^*Y)$$

and that $u^{*-1}(V^* - v^*U^*) \in \mathbb{G}_1$ is the solution to the CDH instance.

When assessing $\mathcal{B}$'s advantage, its probability to fail in a signing query because of a collision on H_2 is at most $q_S(q_{H_2} + q_S)/2^k$ (as L_2 never contains more than $q_{H_2} + q_S$ entries) while $\mathcal{F}$'s probability to output a valid forgery $\langle U^*, V^* \rangle$ on M^* without asking the corresponding $H_2(\mathsf{ID}^*, M^*, U^*)$ query is at most $1/2^k$. Finally, by a similar analysis to Coron's one [68], the probability for $\mathcal{B}$ not to fail in answering a private key extraction query is at least $(1 - 1/(q_{KE} + 1))^{q_{KE}} > 1/e$. Its probability not to fail because $\mathcal{F}$ chooses a "bad" target identity ID^* is $1/(q_{KE} + 1)$. Eventually, $\mathcal{B}$'s advantage is at least

$$\frac{\epsilon}{e(q_{KE}+1)}\Big(1 - \frac{1}{2^k}\big(q_S(q_{H_2} + q_S) + 1\big)\Big) > \frac{1}{e(q_{KE}+1)}\Big(\epsilon - \frac{1}{2^k}\big(q_S(q_{H_2} + q_S) + 1\big)\Big).$$

□

We note that the obtained reduction is tighter than for any previously known ID-based signature scheme: at this stage, our bound on ϵ' is already much better than Kurosawa and Heng's one [124] which is $O(\epsilon^2/eq_{KE}q_H)$ when improved by replacing the quadratic degradation factor in q_H with a linear one[3]. As an example, for $k = 160$, if we allow $q_{H_1}, q_{H_2} < 2^{60}$ and $q_{KE}, q_S < 2^{30}$, we have $q_S(q_{H_2} + q_S)/2^k < 2 \times 2^{90}/2^{160} = 2^{-69}$. Assuming that the advantage of any algorithm in solving CDH within a time t is at most $\epsilon' < 2^{-60}$, we obtain that $(\epsilon - 2^{-69})/2^{32} \leq \epsilon' < 2^{-60}$ and the probability for an attacker to break SOK-IBS within a time bound close to t is bounded by $\epsilon \leq 2^{-28} + 2^{-69} < 2 \times 2^{-28} = 2^{-27}$.

We also compare key sizes that guarantee the infeasibility of breaking SOK-IBS and schemes to which the Kurosawa-Heng technique [124] applies. The latter uses a forger running in time $t'_{\mathcal{F}}$ to solve the CDH problem in expected time $t'_{CDH} = O((q_{KE}q_H/\epsilon^2)t'_{\mathcal{F}})$. We observed that a forger running in a time $t_{\mathcal{F}}$ against SOK-IBS yields an algorithm solving CDH in expected time $t_{CDH} = O((q_{KE}/\epsilon)t_{\mathcal{F}})$. To ensure the same level of security as SOK-IBS, the Cha-Cheon scheme [51] thus needs a security parameter which is about $2 \times \log_2(q_H/\epsilon) \approx 240$ bits longer if

[3]We believe that it suffices to append the signer's identity to the message hashed along with the commitment in the signature generation to obtain this bound.

both schemes are designed to ensure that no forger has a better advantage than 2^{-60}. When considering adversaries performing fewer than 2^{80} operations, we find that $k = 340$ is sufficient for SOK-IBS whereas a security parameter of $k' = 580 > 1.7 \times k$ is required for the Cha-Cheon IBS. When taking into account that, for similar security parameters, the verification algorithm of SOK-IBS is about 1.5 times as expensive as in Cha-Cheon (3 pairing operations being involved instead of 2) and that the complexity of Miller's algorithm grows linearly with k, we find that SOK-IBS is faster for both signing and verifying for a desired security level.

We have to precise that such a comparison only makes sense if both schemes are instantiated with symmetric pairings. Indeed, the Cha-Cheon scheme lends itself much better to an implementation with asymmetric pairings and ordinary curves aiming at minimizing the size of signatures: in such a setting, a part of an SOK-IBS signature would have to lie in a group of large representation.

5.2. An optimal reduction under a stronger assumption.

The theorem below shows that an optimal reduction exists from the stronger *one more Diffie-Hellman* assumption which is defined as follows.

Definition 5.4 ([34]) *Given $\langle P, aP\rangle \in \mathbb{G}_1$ for an unknown $a \in \mathbb{Z}_q$, a target oracle $\mathcal{T}_{\mathbb{G}_1}$ returning randomly chosen elements $Y_i \in \mathbb{G}_1$ (for $i = 1, \ldots, q_t$, q_t being the exact number of queries to this oracle) as well as a multiplication oracle $\mathcal{H}_{\mathbb{G}_1,a}(.)$ answering $aW \in \mathbb{G}_1$ when queried on an input $W \in \mathbb{G}_1$, the **one more Diffie-Hellman problem** (1m-CDHP) is to produce a list $((Z_1, 1), \ldots, (Z_{q_t}, q_t))$ of q_t pairs such that $Z_i = aY_i \in \mathbb{G}_1$, for $i = 1, \ldots, q_t$, without making more than $q_t - 1$ queries to the multiplication oracle.*

*The **one more Diffie-Hellman assumption** is the intractability of the one more Diffie-Hellman problem for any PPT algorithm.*

The above relaxation of the Diffie-Hellman problem was introduced by Boldyreva in [34] and subsequently used in [124, 28] to prove the security of interactive protocols.

In [94], inspired by a work of Koblitz and Menezes [122] who formalized a new RSA-related problem to study the concrete security of RSA

based signature schemes, Galindo considered another relaxation of the Diffie-Hellman problem called CDH1(q_{KE}, q_H) problem. Roughly said, given $\langle P, aP\rangle$ and a list of elements $Q_i \in \mathbb{G}_1$ for $i \in \{1, \ldots, q_{KE} + q_H\}$, the latter consists in choosing up to q_{KE} of those Q_i for which one obtains aQ_i and then producing a solution aQ_j for one of the q_H remaining Q_j. This problem is *not* harder than the *1m-CDH* problem as an oracle solving the latter can easily be shown to help in solving the former in polynomial time.

Although the best known reduction from the usual Diffie-Hellman problem to its CDH1(q_{KE}, q_H) relaxation involves $O(q_{KE})$ calls to a CDH1(q_{KE}, q_H) oracle, Galindo [94] gave arguments according to which the CDH and CDH1(q_{KE}, q_H) problems have similar complexities in practice. When putting those considerations altogether, it comes that a reasonable confidence can be invested in the one-more Diffie-Hellman assumption.

Theorem 5.3 shows the existence of an algorithm solving the $1m$-CDH problem whith a probability which is, up to a negligible term, as large as a forger's advantage against SOK-IBS. The proof is quite simple.

Theorem 5.3 *If a PPT chosen-message attacker $\mathcal{F}$ has an advantage ϵ over SOK-IBS when running in a time t, asking q_{h_i} queries to random oracles H_i (i=1,2), q_{KE} private key generation queries and q_S signing queries, then there is an algorithm $\mathcal{B}$ to solve the $1m$-CDHP with an advantage $\epsilon' > \epsilon - \big(q_S(q_{H_2}+q_S)+1\big)/2^k$ in a time $t' < t + O((q_{H_2}+q_S)t_m)$ where t_m is the cost of a scalar multiplication in $\mathbb{G}_1$.*

Proof. Let $\langle P, X = aP, \mathcal{T}_{\mathbb{G}_1}, \mathcal{H}_{\mathbb{G}_1,a}(.)\rangle$ be an instance of the one more CDH problem. To solve it, the simulator $\mathcal{B}$ runs $\mathcal{F}$ with the domain-wide key $P_{pub} = aP \in \mathbb{G}_1$. The forger $\mathcal{F}$ then starts querying the various oracles that are simulated as follows:

- queries on oracle H_1: when a new identity ID_i is submitted to this oracle, $\mathcal{B}$ queries the target oracle $\mathcal{T}_{\mathbb{G}_1}$ (recall that this oracle takes no input) and forwards the obtained random element $Y_i \in \mathbb{G}_1$ as an answer to $\mathcal{F}$. The pair (ID_i, Y_i) is stored in a list L_1. If the same identity is submitted to H_1 again, the stored answer is returned.

- Private key queries on identities ID_i: we assume ID_i was previously submitted to the H_1 oracle. The corresponding $Y_i \in \mathbb{G}_1$ that was obtained from $\mathcal{T}_{\mathbb{G}_1}$ is recovered from L_1 and sent by $\mathcal{B}$ to the multiplication oracle $\mathcal{H}_{\mathbb{G}_1,a}(.)$ whose output $aY_i \in \mathbb{G}_1$ is returned to $\mathcal{F}$ as a private key for ID_i. The elements $(\mathsf{ID}_i, Y_i, aY_i)$ are stored in a list L_E.
- H_2 queries and signing queries are dealt with exactly as in the proof of theorem 1.

Since $\mathcal{F}$ is assumed to produce a forgery for an uncorrupted identity ID^*, and since we can assume that $H_1(\mathsf{ID}^*)$ was asked during the game, it follows that the number q_{h_1} of target oracle queries made by $\mathcal{B}$ is strictly smaller than the number q_{kg} of queries to $\mathcal{H}_{\mathbb{G}_1,a}(.)$. Furthermore, the private key $d_{\mathsf{ID}^*} = V^* - v^*U^*$ associated to the uncorrupted identity ID^* can be extracted from the produced forgery $(M^*, \langle U^*, V^* \rangle)$ and from the content of the list L_2 (where $H_2(\mathsf{ID}^*, M^*, U^*)$ was defined to be $v^*P \in \mathbb{G}_1$) since we have the equality $\hat{e}(P, V^* - v^*U^*) = \hat{e}(X, Y^*)$ and Y^* is the value of $H_1(\mathsf{ID}^*)$ fixed by $\mathcal{T}_{\mathbb{G}_1}$. □

6. Conclusion

This chapter gave two new results regarding identity-based signatures. The first one is a new pairing-based scheme that is about twice as fast as previous ones at verification. The second result is to show the existence of improved security reductions for an existing scheme. The latter can even be shown to be tightly related to a reasonable computational assumption, which seems to be a very rare feature for an IBS scheme.

CHAPTER 6

An Identity-Based Undeniable Signature

Abstract. This chapter provides a first example of identity based undeniable signature using pairings. We extend to the identity based setting the security model for the notions of invisibility and anonymity given by Galbraith and Mao in 2003 and we prove that our scheme is existentially unforgeable under the Gap Bilinear Diffie-Hellman assumption in the random oracle model. We also prove that it has the invisibility property under the Decisional Bilinear Diffie-Hellman assumption and discuss the efficiency of the scheme.

1. Undeniable signatures

Undeniable signatures are a concept introduced by Chaum and van Antwerpen in 1989 [53]. It is a kind of signatures that cannot be verified without interacting with the signer. They are useful in situations where the validity of a signature should not be universally verifiable. For example, a software vendor might want to embed signatures into his products and allow only paying customers to check the authenticity of these products. If the vendor actually signed a message, he must be able to convince the customer of this fact using a confirmation protocol and, if he did not, he must also be able to convince the customer that he is not the signer thanks to a denial protocol. These proofs have to be non-transferable: once a verifier is convinced that the vendor did or did not sign a message, he should be unable to transmit this conviction to a third party.

In some applications, a signer needs to decide not only when but also by whom his signatures can be verified. For example a voting center can give a voter a proof that his vote was actually counted without letting him the opportunity to convince someone else of his vote. That is the motivation of designated verifier proofs [116] for undeniable signatures. This kind of proof involves the verifier's public key in such a way that

he is not able to convince a third party that a signer actually signed a message or not because he is able to produce such a valid proof himself using his private key. Several proof systems were proposed for undeniable signatures [87, 116, 181]. The use of designated verifier proofs [116] can provide non-interactive and non-transferable confirmation and denial protocols.

Several examples of undeniable signature schemes based on discrete logarithms were proposed [53, 52, 54]. The original construction of Chaum and van Antwerpen [53] was proved secure in 2001 by Okamoto and Pointcheval [168] thanks to the use of a new kind of computational problem. Several convertible[1] undeniable signatures were proposed [44, 186, 151]. In 1997, Michels and Stadler proposed a convertible undeniable signature scheme supporting designated-verifier verification. RSA-based undeniable signatures were designed by Gennaro, Krawczyk and Rabin [97] and Galbraith, Mao and Paterson [91]. In 2004, Monnerat and Vaudenay [157, 156] proposed schemes based on the hardness of other computational problems and, more recently, Laguillaumie and Vergnaud [129] published a pairing-based scheme where all signatures pertaining to a particular time-period can be converted into universally verifiable signatures.

In an identity-based setting, an example of identity-based undeniable signature was proposed in [109] but it was found to be insecure in [226]. A subsequent work [63] proposed a fix for the security flaw of the latter scheme but even the repaired scheme has no security proof. In a paper published at CT-RSA'04 [134], we showed the first example of such a provably secure scheme.

Chaum, van Heijst and Pfitzmann introduced the notion of 'invisibility' for undeniable signatures. Intuitively, it corresponds to the inability for a distinguisher to decide whether a message-signature pair is valid for a given user or not. The RSA-based schemes described in [91] and [97] do not provide invisibility. In [90], Galbraith and Mao described a new RSA-based undeniable signature that provides invisibility under the so-called composite decision Diffie-Hellman assumption and they show that

[1]See [44]. Convertible undeniable signatures are undeniable signatures that can be converted by the signer into universally verifiable signatures.

invisibility and anonymity[2] are essentially equivalent security notions for undeniable signature schemes satisfying some particular conditions. In this paper, we extend these two security notions to the identity-based setting and we prove in the random oracle model that our scheme is both existentially unforgeable and invisible under some reasonable computational assumptions. Invisibility and anonymity can also be shown to be equivalent in the context of identity-based cryptography; we will not elaborate on this here.

In section 2, we first describe a formal model and security notions related to identity-based undeniable signatures (IBUS). In section 3, we describe the different components of our scheme. We then show their correctness and discuss their efficiency. The rest of the chapter consists of a security analysis of the scheme in the random oracle model.

2. Formal model of identity-based undeniable signature

An identity-based undeniable signature (IBUS) is made of five algorithms.

Setup: the PKG takes as input a security parameter k and produces a public/private key pair (s, P_{pub}) and the system-wide public parameters params. s is the system's master key and P_{pub} is the PKG's public key that must be certified.

Keygen: given a user's identity ID, the PKG uses its master secret key s to compute the corresponding private key d_{ID} and transmit it to the user through a secure channel.

Sign: given a message $M \in \{0,1\}^*$ and his private key d_{ID}, the signer generates a signature σ on M for his identity ID.

Confirm: is an algorithm that takes as input a message $M \in \{0,1\}^*$, a designated verifier's identity $\mathsf{ID}_B \in \{0,1\}^*$, the signer's private key d_{ID_A} and a valid signature σ for the pair (M, ID_A). The output is a non-interactive and non-transferable proof that σ is actually a valid signature on M for the identity ID_A.

Deny: is a similar algorithm to **Confirm** but its input is an invalid signature σ for a given pair (M, ID_A), the private key d_{ID_A} and the designated verifier's identity ID_B. Its output is a

[2]This security notion is related to the inability for an adversary to decide which user generated a particular message-signature pair in a multi-user setting.

non-interactive designated verifier proof that σ is not a valid signature for the message M and the identity ID_A.

The first security notion that we consider is close to the one for other existing identity-based signatures: it is the notion of existential unforgeability under chosen-message attacks.

Definition 6.1 *An identity-based undeniable signature (IBUS) scheme is* ***existentially unforgeable*** *under chosen-message attacks if no PPT adversary has a non-negligible advantage in the following game:*

1. *The challenger runs the setup algorithm to generate the system-wide parameters and sends them to the adversary.*
2. *The adversary $\mathcal{F}$ performs a series of queries:*
   - *Key extraction queries: $\mathcal{F}$ produces an identity ID and obtains the matching private key d_{ID}.*
   - *Signature queries: $\mathcal{F}$ produces a message M and an identity ID and receives a signature on M that is generated by the signing oracle using the private key corresponding to the public key ID.*
   - *Confirmation/denial queries: $\mathcal{F}$ produces some message-signature pair (M, σ), a purported signer's identity ID_A together with an intended verifier's identifier ID_B which are given to an oracle that runs the confirmation/denial protocol using the private key d_{ID_A} to either convince ID_B that σ is actually related to M and ID_A or that it is not (in a non-transferable way).*
3. *After a number of queries, $\mathcal{F}$ produces a tuple $(\mathsf{ID}_A, M, \sigma)$ made of an identity ID_A, whose corresponding private key was not asked at stage 2, and a message-signature pair (M, σ) that was not trivially obtained from the signature oracle at stage 2 for the identity ID_A.*

The forger $\mathcal{F}$ wins the game if she is able to provide a non-transferable proof of validity of the signature σ for message M and identity ID_A for the identity of some uncorrupted verifier ID_B. Her advantage is defined as her probability of success taken over the coin-flippings of the challenger and $\mathcal{F}$.

A second security notion for undeniable signatures was introduced by Chaum, van Heijst and Pfitzmann [54] and is called 'invisibility'. Informally, this notion corresponds to the inability for a curious verifier to decide whether a given signature on a given message was issued by some signer even after having observed several executions of confirmation/denial protocols by the same signer for other signatures. Galbraith and Mao [90] proposed a general definition for this security notion. In the identity-based setting, we need to strengthen it a little to consider the fact that a dishonest user might be in possession of private keys associated to other identities before trying to validate or invalidate an alleged signature on a message for an identity without the help of the alleged signer.

Definition 6.2 *An IBUS scheme is said to satisfy the* ***invisibility*** *property if no PPT distinguisher $\mathcal{D}$ has a non-negligible advantage against a challenger in the following game:*

1. *The challenger performs the setup of the scheme and sends the public parameters to $\mathcal{D}$.*
2. *The distinguisher $\mathcal{D}$ issues a number of queries: key extraction queries, signature queries and confirmation/denial queries of the same kind as those of the previous definition. After a first series of queries, $\mathcal{D}$ asks for a challenge: she produces a pair (M, ID_A) made of a message and an identity for which the associated private key was not asked. The challenger then flips a coin $b \stackrel{R}{\leftarrow} \{0,1\}$. If $b = 0$, the challenger sends $\mathcal{D}$ a valid signature σ on M for the identity ID_A. Otherwise, $\mathcal{D}$ receives from the challenger a random element $\sigma \stackrel{R}{\leftarrow} \mathcal{S}$ taken at random from the signature space $\mathcal{S}$.*
3. *The distinguisher $\mathcal{D}$ then performs a second series of queries. This time, she is disallowed to perform any confirmation/denial query for the challenge $(\sigma, M, \mathsf{ID}_A)$. Besides, she may not ask for the private key associated to ID_A. If the signing algorithm is deterministic, she may not submit the pair (M, ID_A) to the signing oracle at any time. Eventually, $\mathcal{D}$ outputs a bit b' (that is 0 if $\mathcal{D}$ finds that (σ, M, ID) is a valid message-signature-identity tuple and 1 otherwise) and wins the game if $b = b'$.*

$\mathcal{D}$'s advantage is defined as $Adv^{inv}(\mathcal{D}) := 2 \times Pr[b = b'] - 1$.

The above probability is taken over the coin flippings of the distinguisher $\mathcal{D}$ and the challenger. Similarly to what is done in [90], we also consider the notion of anonymity which is slightly strengthened in the identity-based setting

Definition 6.3 *We say that an IBUS scheme satisfies the* ***anonymity*** *property if no PPT distinguiser $\mathcal{D}$ has a non-negligible advantage in the following game:*

1. *The challenger performs the setup of the scheme and sends the public parameters to $\mathcal{D}$.*
2. *The distinguisher $\mathcal{D}$ issues a number of key extraction, signature and confirmation/denial queries. After a first series of queries, $\mathcal{D}$ produces a message M and a pair of identities $\mathsf{ID}_0, \mathsf{ID}_1$ for which she did not obtain the matching private keys. The challenger then flips a coin $b \stackrel{R}{\leftarrow} \{0,1\}$ and provides $\mathcal{D}$ with a signature σ on M for the identity ID_b.*
3. *The distinguisher $\mathcal{D}$ then issues new queries with the restriction that she is disallowed to perform any confirmation/denial query for the challenge σ on identities ID_0, ID_1 and to request the private key associated to these identities. If the signature issuing algorithm is deterministic, she may not request a signature on M for ID_0 nor ID_1 at any time. Eventually, $\mathcal{D}$ outputs a bit b' for which it finds that σ is a valid signature on M for the identity $\mathsf{ID}_{b'}$.*

$\mathcal{D}$ wins the game if $b' = b$. Its advantage is defined as in definition 6.2.

It is shown in [90] that the notions of invisibility and anonymity are essentially equivalent for undeniable and confirmer signature schemes satisfying some particular properties. It is almost straightforward (by using the techniques of [90]) to show that this equivalence also holds in the identity-based setting. We will not do it here. In the next section, we describe a first example of identity-based undeniable signature and we just focus on proving its existential unforgeability and its invisibility in the random oracle model.

3. An identity-based undeniable signature

In a first step, we describe the signature issuing algorithm and we explain why it can be regarded as an adaptation of the Chaum-van Antwerpen scheme [53]. We will then describe algorithms to confirm valid signatures and disavow invalid purported ones thanks to non-interactive designated verifier zero-knowledge proofs.

3.1. The setup, key generation and signing algorithms

The first three algorithms of our scheme are depicted on the next figure. The scheme is reminiscent of a variant of the Chaum-van Antwerpen scheme that was proved secure by Okamoto and Pointcheval [168] under the Gap Diffie-Hellman assumption. Recall that, in the latter scheme, public keys have the form $y = g^x$ where g is the generator of a prime order cyclic group and a signature on a message m is given by $\sigma = h(m)^x$, the latter hash value $h(m)$ being taken over the cyclic group generated by g. To confirm (resp. deny) an alleged message-signature pair, the signer produces a (possibly interactive) non-transferable zero-knowledge proof of equality (resp. inequality) of the discrete logarithms of y and σ w.r.t bases g and $h(m)$.

Actually Chaum's scheme can be generalized using any hard-to-invert isomorphism $f : \mathbb{G} \to \mathbb{G}'$ between groups $\mathbb{G}, \mathbb{G}'$ where computing discrete logarithms is infeasible and which support the aforementioned kind of non-transferable zero-knowledge proofs. In Chaum's scheme, such an isomorphism is provided by the discrete exponentiation in a cyclic group. Using bilinear maps $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$, a suitable isomorphism $f_P : \mathbb{G}_1 \to \mathbb{G}_2$ for a base $P \in \mathbb{G}_1$ can be $f_P(Q) = \hat{e}(P, Q)$ as well. Indeed, we can check that inverting f_P is difficult if the Diffie-Hellman problem is intractable in the group generated by P.

Basically, in our scheme, the isomorphism f_P maps the signer's private key d_{ID} onto a publicly computable quantity $\hat{e}(P_{pub}, Q_{\mathsf{ID}})$ (i.e. the pairing of the system-wide public key and a hash value of the signer's identity) whereas a signature on a message M is the image of the private key d_{ID} for the isomorphism $f_{H_2(M)}$. In order to simplify the security proofs, the hash function H_2 takes as input a message M concatenated to the signer's identity and a random string which is only used to obtain a better security reduction.

Setup: given security parameters k and l so that l is polynomial in k, the PKG chooses symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of prime order $q > 2^k$, a generator P of $\mathbb{G}_1$ and hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1, H_2 : \{0,1\}^* \to \mathbb{G}_1$ that will be used in the signing algorithm itself and $H_3, H_4 : \{0,1\}^* \to \mathbb{Z}_q^*$ which will be respectively used in the confirmation and denial algorithms. It chooses a master secret $s \xleftarrow{R} \mathbb{Z}_q$ and computes $P_{pub} = sP \in \mathbb{G}_1$ that is made public. The system's public parameters are

$$\mathsf{params} := \{q, k, l, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, P_{pub}, H_1, H_2, H_3, H_4\}.$$

Keygen: given a user's identity ID, the PKG computes $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in \mathbb{G}_1$ and the associated private key $d_{\mathsf{ID}} = sQ_{\mathsf{ID}} \in \mathbb{G}_1$.

Sign: to sign a message $M \in \{0,1\}^*$, the signer uses the private key d_{ID_A} associated to her identity ID_A.

(1) She chooses $r \xleftarrow{R} \{0,1\}^l$ to compute $H_2(M, r, \mathsf{ID}_A) \in \mathbb{G}_1$.
(2) She then computes $\gamma = \hat{e}(H_2(M, r, \mathsf{ID}_A), d_{\mathsf{ID}_A}) \in \mathbb{G}_2$. The signature on M is given by

$$\sigma = \langle r, \gamma \rangle = \langle r, \hat{e}(H_2(M, r, \mathsf{ID}_A), d_{\mathsf{ID}_A}) \rangle \in \{0,1\}^l \times \mathbb{G}_2.$$

FIGURE 6.1. Our IBUS scheme

The confirmation and denial algorithms, which are detailed in the next two paragraphs, consist of non-interactive designated-verifier proofs of equality or inequality of two pre-images of isomorphism f_P and $f_{H_2(M)}$.

3.2. The confirmation algorithm

The confirmation protocol is a pairing-based adaptation of a (re-paired) designated verifier proof [116] proposed by Jakobsson, Sako and Impagliazzo that allows a prover to convince a designated verifier of the equality of two discrete logarithms. Actually, the original proof system proposed by Jakobsson et al. suffers from a security flaw as well as the quite similar one used by Galbraith and Mao. This was noticed by F. Zhang, who also found how to easily fix these problems.

The algorithm produces a non-interactive and non-transferable proof of equality of two inverses of the group isomorphisms $f_Q : \mathbb{G}_1 \to \mathbb{G}_2, Q \to f_Q(U) = \hat{e}(Q, U)$ with $Q = P$ and $Q = H_2(M, r, \mathsf{ID}_A)$. In an execution of the confirmation protocol, the verifier B takes the signature as valid

Confirm: to verify a signature σ on a message M, a verifier of identity ID_B needs the help of the signer ID_A. He sends her the pair (M, σ), where $\sigma = \langle r, \gamma \rangle \in \{0,1\}^l \times \mathbb{G}_2$ is the alleged signature. The alleged signer then runs the following confirmation protocol to produce a non-interactive designated-verifier proof that σ is a valid signature on M for her identity ID_A:

(a) She first computes $Q_{\mathsf{ID}_B} = H_1(\mathsf{ID}_B)$.
(b) She picks $U, R \overset{R}{\leftarrow} \mathbb{G}_1$ and $v \overset{R}{\leftarrow} \mathbb{Z}_q$ and computes

$$c = \hat{e}(P, U)\hat{e}(P_{pub}, Q_{\mathsf{ID}_B})^v \in \mathbb{G}_2$$

$$g_1 = \hat{e}(P, R) \in \mathbb{G}_2 \text{ and } g_2 = \hat{e}(H_2(M, r, \mathsf{ID}_A), R) \in \mathbb{G}_2.$$

(c) She takes the hash value $h = H_3(c, g_1, g_2, M, r, \gamma) \in \mathbb{Z}_q$.
(d) She computes $S = R - (h + v)d_{\mathsf{ID}_A}$.

The proof is made of (U, v, h, S) and is checked by the verifier like this: he first computes $c' = \hat{e}(P, U)\hat{e}(P_{pub}, Q_{\mathsf{ID}_B})^v$ and then $g_1' = \hat{e}(P, S)\hat{e}(P_{pub}, Q_{\mathsf{ID}_A})^{h+v}$ and $g_2' = \hat{e}(H_2(M, r, \mathsf{ID}_A), S)\gamma^{h+v}$. He accepts the proof if and only if $h = H_3(c', g_1', g_2', M, r, \gamma)$.

FIGURE 6.2. A confirmation algorithm for IBUS

if he is convinced that $f_P(d_{\mathsf{ID}_A}) = \hat{e}(P_{pub}, Q_{\mathsf{ID}_A})$ and γ have identical pre-images for isomorphisms $f_P(.) = \hat{e}(P, .)$ and $f_{H_2(M,r,\mathsf{ID}_A)}(.) = \hat{e}(H_2(M, r, \mathsf{ID}_A), .)$.

3.2.1. Completeness and soundness of the confirmation proof

It is easy to see that a correct proof is always accepted by the verifier B: if (U, v, h, S) is correctly computed by the prover, we have $\hat{e}(P, S) = \hat{e}(P, R)\hat{e}(P, d_{\mathsf{ID}_A})^{-(h+v)}$ and $\hat{e}(P, d_{\mathsf{ID}_A}) = \hat{e}(P_{pub}, Q_{\mathsf{ID}_A})$. We also have $\hat{e}(H_2(M, r, \mathsf{ID}_A), R) = \hat{e}(H_2(M, r, \mathsf{ID}_A), S)\hat{e}(H_2(M, r, \mathsf{ID}_A), d_{\mathsf{ID}_A})^{-(h+v)}$. In order to show the soundness, we notice that if a prover is able to provide two correct answers S_1, S_2 for the same commitment (c, g_1, g_2) and two different challenges h_1 and h_2, we then have the relations

$$\hat{e}(P, (h_2 - h_1)^{-1}(S_1 - S_2)) = \hat{e}(P_{pub}, Q_{\mathsf{ID}_A})$$

$$\hat{e}(H_2(M, r, \mathsf{ID}_A), (h_2 - h_1)^{-1}(S_1 - S_2)) = \gamma$$

which indicate that inverses $f_P^{-1}(\hat{e}(P_{pub}, Q_{\mathsf{ID}_A}))$, $f_{H_2(M,r,\mathsf{ID}_A)}^{-1}(\gamma)$ are equal.

Deny: in order to convince a designated verifier of identity ID_B that a given signature $\sigma = \langle r, \gamma \rangle$ on a message M is not valid for her identity,

(a) Entity ID_A computes $Q_{\mathsf{ID}_B} = H_1(\mathsf{ID}_B) \in \mathbb{G}_1$ and picks $U \xleftarrow{R} \mathbb{G}_1$, $v \xleftarrow{R} \mathbb{Z}_q$ to compute $c = \hat{e}(P, U)\hat{e}(P_{pub}, Q_{\mathsf{ID}_B})^v$.

(b) She computes a commitment $C = \left(\frac{\hat{e}(H_2(M,r,\mathsf{ID}_A), d_{\mathsf{ID}_A})}{\gamma}\right)^{\omega}$ for a randomly chosen $\omega \xleftarrow{R} \mathbb{Z}_q^*$.

(c) She produces a NIZK proof that she knows a pair $(R, \alpha) \in \mathbb{G}_1 \times \mathbb{Z}_q$ such that

$$C = \frac{\hat{e}(H_2(M, r, \mathsf{ID}_A), R)}{\gamma^{\alpha}} \quad \text{and} \quad 1 = \frac{\hat{e}(P, R)}{\hat{e}(P_{pub}, Q_{\mathsf{ID}_A})^{\alpha}} \tag{1}$$

To do this,

(1) She picks $V \xleftarrow{R} \mathbb{G}_1$, $v \xleftarrow{R} \mathbb{Z}_q$ to compute

$$\rho_1 = \hat{e}(H_2(M, r, \mathsf{ID}_A), V)\gamma^{-v} \in \mathbb{G}_2$$
$$\rho_2 = \hat{e}(P, V)y^{-v} \in \mathbb{G}_2$$

where $y = \hat{e}(P_{pub}, Q_{\mathsf{ID}_A})$.

(2) She computes $h = H_4(C, c, \rho_1, \rho_2, M, r, \gamma) \in \mathbb{Z}_q$.

(3) She computes

$$S = V + (h + v)R \in \mathbb{G}_1$$
$$s = v + (h + v)\alpha \in \mathbb{Z}_q.$$

The proof is made of (C, U, v, h, S, s). It can be verified by the verifier of identity ID_B who rejects the proof if $C = 1$ and otherwise computes $c' = \hat{e}(P, U)\hat{e}(P_{pub}, Q_{\mathsf{ID}_B})^v$, $\rho_1' = \hat{e}(H_2(M, r, \mathsf{ID}_A), S)\gamma^{-s}C^{-(h+v)}$ and $\rho_2' = \hat{e}(P, S)y^{-s}$ where $y = \hat{e}(P_{pub}, Q_{\mathsf{ID}_A})$. The verifier accepts the proof if and only if $h = H_4(C, c', \rho_1', \rho_2', M, r, \gamma)$.

FIGURE 6.3. A denial algorithm for IBUS

3.3. The denial algorithm

The denial protocol is an adaptation of a protocol proposed by Camenisch and Shoup [47] to prove the inequality of two discrete logarithms. This adaptation is a non-transferable proof of inequality of two inverses of the group isomorphisms $f_Q : \mathbb{G}_1 \to \mathbb{G}_2, Q \to f_Q(U) = \hat{e}(Q, U)$ with $Q = P$ and $Q = H_2(M, r, \mathsf{ID}_A)$. The verifier B deems the signature

invalid if he is convinced that $f_P(d_{\mathsf{ID}_A}) = \hat{e}(P_{pub}, Q_{\mathsf{ID}_A})$ and γ have different pre-images for isomorphisms $f_P(.) = \hat{e}(P, .)$ and $f_{H_2(M,r,\mathsf{ID}_A)}(.) = \hat{e}(H_2(M, r, \mathsf{ID}_A), .)$.

3.3.1. Completeness and soundness of the denial proof

One easily checks that an honest prover is always accepted by the designated verifier. To prove the soundness, one notices that if the prover is able to provide a proof of knowledge of a pair (R, α) satisfying equations (1), then the second of these equations implies $R = \alpha f_P^{-1}(y)$ with $y = \hat{e}(P_{pub}, Q_{\mathsf{ID}_A})$ by the bilinearity of the map. If we substitute this relation in the first equation of (1), it comes that

$$C = \Big(\frac{\hat{e}(H_2(M, r, \mathsf{ID}_A), f_P^{-1}(y))}{\gamma}\Big)^{\alpha}.$$

As the verifier checks that $C \neq 1$, this implies $\hat{e}(H_2(M, r, \mathsf{ID}_A), f_P^{-1}(y)) \neq \gamma$ and the signature γ is actually invalid. The soundness of the proof of knowledge in step (c) is easy to verify.

3.4. Non-transferability

In order for the non-interactive proofs to be non-transferable, they need a trapdoor commitment $\mathsf{Commit}(U, v) = \hat{e}(P, U)\hat{e}(P_{pub}, Q_{\mathsf{ID}_B})^v$ that allows the owner of the private key d_{ID_B} to compute commitment collisions: indeed, given a tuple $(U, v, \mathsf{Commit}(U, v))$, B can easily use d_{ID_B} to find a pair (U', v') such that $\mathsf{Commit}(U, v) = \mathsf{Commit}(U', v')$. This is essential for the proof to be non-transferable: the verifier B cannot convince a third party of the validity or of the invalidity of a signature since his knowledge of the private key d_{ID_B} allows him to produce such a proof himself. Indeed, given a message-signature pair (M, σ), with $\sigma = \langle r, \gamma \rangle \in \{0,1\}^l \times \mathbb{G}_2$, B can choose $S \overset{R}{\leftarrow} \mathbb{G}_1$, $x \overset{R}{\leftarrow} \mathbb{Z}_q$ and $U' \overset{R}{\leftarrow} \mathbb{G}_1$ to compute $c = \hat{e}(P, U')$, $g_1 = \hat{e}(P, S)\hat{e}(P_{pub}, Q_{\mathsf{ID}_A})^x$, $g_2 = \hat{e}(H_2(M, r, \mathsf{ID}_A), S)\gamma^x$ and $c = H_3(c, g_1, g_2, M, r, \gamma)$. He can then compute $v = x - h \bmod q$ and $U = U' - v d_{\mathsf{ID}_B} \in \mathbb{G}_1$ where d_{ID_B} is the verifier's private key. (U, v, h, S) is thus a valid proof built by the verifier with the trapdoor d_{ID_B}. This trapdoor also allows him to produce a false proof of a given signature's invalidity using the same technique with the denial protocol.

3.5. Efficiency considerations

From an efficiency point of view, the signature generation algorithm requires one pairing evaluation as a most expensive operation. The confirmation and denial protocols are more expensive: the first one requires four pairing evaluations (three if $\hat{e}(P_{pub}, Q_{\mathsf{ID}_B})$ is cached in memory: this can be done if the verifier often performs confirmation/denial queries), one exponentiation in $\mathbb{G}_2$ and one computation of the type $\lambda_1 P + \lambda_2 Q$ in $\mathbb{G}_1$. The verifier needs to compute three pairings (only two if $\hat{e}(P_{pub}, Q_{\mathsf{ID}_A})$ is cached), three exponentiations and three multiplications in $\mathbb{G}_2$. In the denial protocol, the prover must compute five pairings (four if $\hat{e}(P_{pub}, Q_{\mathsf{ID}_B})$ is cached), four exponentiations and four multiplications in $\mathbb{G}_2$, one computation of the type $\lambda_1 P + \lambda_2 Q$ and some extra arithmetic operations in $\mathbb{Z}_q$. The verifier must compute four pairings (three if $\hat{e}(P_{pub}, Q_{\mathsf{ID}_B})$ is cached), two exponentiations, one multi-exponentiation and three multiplications in $\mathbb{G}_2$. To improve the efficiency of the confirmation and denial algorithms, one can speed up the computation of commitments. Indeed, the prover can pre-compute $\hat{e}(P, P)$ once and for all. To generate a commitment in an execution of the confirmation protocol, he then picks $u, v, x \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and computes $c = \hat{e}(P, P)^u \hat{e}(P_{pub}, Q_{\mathsf{ID}_B})^v$, $R = xP$ $g_1 = \hat{e}(P, P)^x$, $g_2 = \hat{e}(H_2(M, r, \mathsf{ID}_A), R)$. The answer to the challenge h must then be computed as $S = R - (h + v) d_{\mathsf{ID}_A}$ and the proof is made of (u, v, h, S). This technique can also be applied in the denial protocol. It allows replacing 2 pairing evaluations by 2 scalar multiplications, one exponentiation and a multi-exponentiation in $\mathbb{G}_2$ (to compute c). A single pairing evaluation is then required for the prover at each execution of the confirmation and denial protocols if verifier-related pairings $\hat{e}(P_{pub}, Q_{\mathsf{ID}_B})$ are pre-computed.

Globally, it turns out that a signature validation/invalidation is expensive for verifiers as three pairings have to be computed even if a precomputation is performed. Fortunately, recent server-aided verification techniques [102] might be applied here. However, our IBUS proposal remains the only provably secure solution so far.

If we consider the length of signatures, the binary representation of a pairing is about 1024 bits long for recommended parameters [43] while the length l of the binary string can be of the order of 160 bits. This

provides us with signatures of about 1184 bits. This is roughly one half of the size of the RSA-based undeniable signature proposed in [90] (this scheme produces signatures of more than 2048 bits if 1024-bit moduli are used). If we compare our scheme with the original undeniable signature proposed by Chaum and van Antwerpen and proved secure by Okamoto and Pointcheval [168], both lengths are similar if the Chaum-van Heijst scheme is used over a group like $\mathbb{Z}_p^*$ with $|p| = 1024$ (this is no longer true if this scheme is used over a suitable[3] elliptic curve).

3.6. Convertible signatures

It is easy to notice that issued signatures can be selectively turned into universally verifiable signatures by the signer. In order to convert a genuine signature $\sigma = \langle r, \hat{e}(H_2(M, r, \mathsf{ID}_A), d_{\mathsf{ID}_A})\rangle$, the signer Alice just has to take a random $x \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and compute $R = xP$, $g_1 = \hat{e}(P, P)^x$, $g_2 = \hat{e}(H_2(M, r, \mathsf{ID}_A), R)$, the hash value $h = H(g_1, g_2, M, r, \gamma)$ and the answer $S = R - hd_{\mathsf{ID}_A}$. The proof, given by $(h, S) \in \mathbb{Z}_q \times \mathbb{G}_1$, is easily universally verifiable by a method similar to the verification in the confirmation protocol. Alice can also give a universally verifiable proof that a given signature is invalid for her identity by using the non-designated verifier counterpart of the denial algorithm.

3.7. Removing key escrow

If one does not wish to invest too much trust in a PKG, the key escrow property can be removed from the scheme by turning it into a certificateless undeniable signature in accordance with the paradigm introduced by Al-Riyami and Paterson [6]. In the obtained scheme, the signer Alice first sets up her public key $P_A = \langle X_A, Y_A\rangle = \langle x_A P, x_A P_{pub}\rangle \in \mathbb{G}_1^2$ for a secret randomly chosen $x_A \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and request a partial private key from a Key Generation Center (KGC). This KGC takes as input P_A, computes $Q_A = H_1(\mathsf{ID}_A, P_A) \in \mathbb{G}_1$ and the partial private key $D_A = sQ_A \in \mathbb{G}_1$.

Alice then sets her full private key as $S_A = x_A D_A \in \mathbb{G}_1$. A signature on a message M is then computed as $\sigma = \langle r, \hat{e}(H_2(M, r, \mathsf{ID}_A), S_A)\rangle$.

[3] "Suitable" here means ordinary as the Chaum-van Heijst scheme would be a regular digital signature scheme similar to Boneh et al.'s one [43] if it was instantiated with supersingular curves.

She can then validate or invalidate σ by proving the equality or inequality of the inverses of $f_{H_2(M,r,\mathsf{ID}_A)}(.) = \hat{e}(H_2(M,r,\mathsf{ID}_A),.)$ and $f_P(S_A) = \hat{e}(Y_A, Q_A)$ (that is publicly computable).

We do not give security proofs nor formal security models for the obtained scheme here. The advantage of easy key management is lost since the resulting scheme no longer supports human-memorizable public keys. On the other hand, the key escrow, which is often an undesirable feature in signature schemes, is removed as well as the need for public key certificates.

4. Security proofs for IBUS

We first give a proof in the random oracle model that our identity based undeniable signature is existentially unforgeable under adaptive chosen-message attacks. We then provide a proof of its invisibility.

Theorem 6.1 *If there exists an adversary $\mathcal{F}$ that can produce an existential forgery for IBUS with an advantage ϵ within a time t and when performing q_E key extraction queries, q_S signature queries, q_{CD} confirmation/denial queries and q_{H_i} queries on hash oracles H_i, for $i = 1, \ldots, 4$, then there exists an algorithm $\mathcal{B}$ to solve the Gap Bilinear Diffie-Hellman problem with an advantage*

$$\epsilon' \geq \frac{1}{e(q_E+1)}\left(\epsilon - \frac{q_S(q_S+q_{H_2})}{2^l} - \frac{q_{CD}(q_{H_3}+q_{CD})}{2^{k-1}}\right)$$

in a time $t' \leq t + 6\tau_p + O((q_E + q_{H_1} + q_{H_2} + q_{CD})\tau_m + (q_S + q_{CD})\tau_e + q_{CD}\Phi)$ where e is the base for the natural logarithm, τ_p denotes the time required for a pairing evaluation, τ_m is the cost of a scalar multiplication in $\mathbb{G}_1$, τ_e is the time to perform an exponentiation in $\mathbb{G}_2$ and Φ is the complexity of a call to the DBDH oracle.

Proof. We show an algorithm $\mathcal{B}$ using the adversary $\mathcal{F}$ to solve a random instance $(P, aP, bP, cP, \mathcal{O}_{DBDH})$ of the Gap Bilinear Diffie-Hellman problem where $\mathcal{O}_{DBDH}(.)$ denotes the corresponding decision oracle. Algorithm $\mathcal{B}$ will simulate the behaviour of $\mathcal{F}$'s challenger in the game of definition 6.1. It first provides $\mathcal{F}$ with system parameters $\mathsf{params} = \{q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, P_{pub}, H_1, H_2, H_3, H_4\}$ such that $P_{pub} = cP$ and where H_1, H_2, H_3 and H_4 are random oracles.

$\mathcal{F}$ now performs a series of queries as described in definition 6.1. To

deal with them, $\mathcal{B}$ uses lists L_{H_i} to keep track of answers to hash queries on H_i for $i = 1, \ldots, 4$. W.l.o.g., we can assume that hash queries on H_1 are distinct and that every key extraction on an identity ID is preceded by a random oracle query $H_1(\mathsf{ID})$. The queries made by $\mathcal{F}$ are handled as follows:

- H_1-queries on an identity $\mathsf{ID}_i \in \{0,1\}^*$ (we call ID_i the input of the i^{th} H_1-query): $\mathcal{B}$ first picks $\mu_i \stackrel{R}{\leftarrow} \mathbb{Z}_q$. It then flips a coin X that is 0 with probability δ and 1 with probability $1-\delta$ (the optimal value of δ will be determined further). $\mathcal{B}$ then inserts the tuple $(\mathsf{ID}_i, \mu_i, X)$ into the list L_{H_1}. If $X = 1$, $\mathcal{B}$ returns $H_1(\mathsf{ID}_i) = \mu_i(bP) \in \mathbb{G}_1$ to $\mathcal{F}$ (recall that b is unknown to $\mathcal{B}$). Otherwise, $\mathcal{B}$ returns $H_1(\mathsf{ID}_i) = \mu_i P \in \mathbb{G}_1$ as an answer.
- H_2-queries on an intput $(M_i, r_i, \mathsf{ID}_i)$: $\mathcal{B}$ returns the previously defined value if it exists. Otherwise, $\mathcal{B}$ picks $d_i \stackrel{R}{\leftarrow} \mathbb{Z}_q$, returns $d_i(aP) \in \mathbb{G}_1$ and stores the information $(M_i, r_i, \mathsf{ID}_i, d_i, d_i(aP))$ into L_{H_2}.
- key extraction queries for an identity ID_i: $\mathcal{B}$ searches L_{H_1} for the triple (ID_i, μ, X) that must exist. If $X = 1$, then $\mathcal{B}$ aborts as it is unable to answer the query. Otherwise, $\mathcal{B}$ returns μP_{pub} to $\mathcal{F}$ as a private key for ID_i.
- signing queries for a message M_i and an identity ID_i: $\mathcal{B}$ first chooses a random string $r \stackrel{R}{\leftarrow} \{0,1\}^l$ and checks if L_{H_2} already contains a tuple $(M_i, r, \mathsf{ID}_i, ., .)$. If it does, $\mathcal{B}$ fails (this happens with probability at most $(q_S + q_{H_2})/2^l$ as L_{H_2} never contains more than $q_S + q_{H_2}$ elements). Otherwise, $\mathcal{B}$ takes a random $d \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and inserts $(M_i, r, \mathsf{ID}_i, d, dP)$ into L_{H_2} (in such a way that a subsequent $H_2(M_i, r, \mathsf{ID}_i)$ query will receive dP as an answer). Because of the assumptions made above, L_{H_1} must contain a triple $(\mathsf{ID}_i, ., .)$ indicating which value $Q_{\mathsf{ID}_i} = H_1(\mathsf{ID}_i)$ was returned to $\mathcal{F}$ on her previously issued query. $\mathcal{B}$ then returns $\sigma = \langle r, \hat{e}(dP_{pub}, Q_{\mathsf{ID}_i})\rangle$.
- confirmation/denial queries: at any time, $\mathcal{F}$ can produce a message-signature pair (M, σ) together with identities ID_A (the one of the alleged signer) and ID_B (the one of the designated verifier) in a request for an execution of the confirmation/denial

algorithm. To handle such a query, $\mathcal{B}$ first parses σ into $\langle r, \gamma \rangle \in \{0,1\}^l \times \mathbb{G}_2$. It then recovers $Q_{\mathsf{ID}_A} = H_1(\mathsf{ID}_A)$ and $H = H_2(M, r, \mathsf{ID}_A)$ respectively from L_{H_1} and L_{H_2} before resorting to $\mathcal{O}_{DBDH}(.)$ to check whether $(P, P_{pub}, Q_{\mathsf{ID}_A}, H, \gamma)$ is a valid BDH tuple. If yes, a simulation of the confirmation algorithm is provided to $\mathcal{F}$. Otherwise, the latter receives a simulated output of the denial algorithm. The non-interactive proofs provided by the latter two algorithms are very easy to simulate exactly as any NIZK proof in the random oracle model. We do not give the details here but we notice that $\mathcal{B}$ can fail to simulate the confirmation and denial protocols with a probability smaller than $(q_{H_3} + q_{CD})2^{-k}$ (we assume $q_{H_3} \approx q_{H_4}$). This occurs if $\mathcal{B}$ has to set the value of H_3 or H_4 on a point where the oracle was previously defined.

Eventually, $\mathcal{F}$ produces a triple (M, ID, σ) where $\sigma = \langle r, \gamma \rangle$ is a purported signature of a signer of identity ID on the message M. To win, $\mathcal{F}$ must not have queried the private key for ID and must be able to produce a non-transferable proof of validity of σ. For the produced triple, $\mathcal{B}$ searches L_{H_1} and L_{H_2} for records (ID, μ, X) (that must exist because of the assumptions we made) and $(M, r, \mathsf{ID}, d, H)$. If $X = 0$ or if no tuple $(M, r, \mathsf{ID}, d, H)$ exists in L_{H_2}, then $\mathcal{B}$ declares "failure". Otherwise, if $\langle r, \gamma \rangle$ is a valid signature on M for the identity ID and if both inverses $f_P^{-1}(\hat{e}(P_{pub}, Q_{\mathsf{ID}_A}))$ and $f_{H_2(M,r,\mathsf{ID})}^{-1}(\gamma)$ are actually equal, then $\mathcal{B}$ can compute $\gamma^{\frac{1}{\mu d}}$ which is equal to the solution $\hat{e}(P,P)^{abc}$ of the Gap Bilinear Diffie-Hellman problem (P, aP, bP, cP).

We now assess $\mathcal{B}$'s probability of success. The first way for $\mathcal{B}$ to reach a failure state is to receive a key extraction query on an identity ID_i for which the random variable X_i was 1. A failure also happens if the hash value $H_1(\mathsf{ID})$ of the identity involved in $\mathcal{F}$'s forgery was not set to a known multiple of bP. If q_E denotes the number of key extraction queries made by $\mathcal{F}$, it easily comes that the probability for $\mathcal{B}$ to avoid these failure cases is at least $\delta^{q_E}(1-\delta)$. If $\mathcal{B}$ uses the optimal values $\delta_{opt} = q_E/(q_E+1)$, this probability is greater than $\frac{1}{e(q_E+1)}$.

It is also possible that $\mathcal{F}$ does not query $H_2(M, r, \mathsf{ID})$, where (M, r, ID) is a part of its forgery, during the simulation. One easily sees that the probability for this to happen is smaller than $1/2^k$. Finally, the attacker

$\mathcal{F}$ can also produce a forgery $(M, \mathsf{ID}, \langle r, \gamma \rangle)$ for which she is able to give a correct proof of validity but for which $y = \hat{e}(P_{pub}, Q_{\mathsf{ID}}) = \hat{e}(P, d_{\mathsf{ID}})$, $\gamma = \hat{e}(H_2(M, r, \mathsf{ID}), d'_{\mathsf{ID}})$ with $d_{\mathsf{ID}} \neq d_{\mathsf{ID}'}$. Since $\mathcal{F}$ can provide a proof (U, v, h, S) that $(M, \mathsf{ID}, \langle r, \gamma \rangle)$ is valid, we have

$$\begin{aligned} g_1 &= \hat{e}(P, R) = \hat{e}(P, S)\hat{e}(P, d_{\mathsf{ID}})^{h+v}, \\ g_2 &= \hat{e}(H_2(M, r, \mathsf{ID}), R') = \hat{e}(H_2(M, r, \mathsf{ID}), S)\hat{e}(H_2(M, r, \mathsf{ID}), d'_{\mathsf{ID}})^{h+v} \end{aligned}$$

and then $h = \log_{d_{\mathsf{ID}} - d'_{\mathsf{ID}}}(R - R') - v$. Such a situation only occurs if a hash value $H_3(c, \hat{e}(P, R), \hat{e}(H_2(M, r), R'))$ is set to $\log_{d_{\mathsf{ID}} - d'_{\mathsf{ID}}}(R - R') - v$ by $\mathcal{B}$. The probability for this to happen is not greater than $q_{H_3} 2^{-k}$. Finally, the probability for $\mathcal{B}$ to fail in the simulation of a confirmation/denial is less than $2 \times q_{CD}(q_{H_3} + q_{CD})2^{-k}$ (since we assumed that $q_{H_3} = q_{H_4}$ and lists L_{H_3} and L_{H_4} never contain more than $q_{H_3} + q_{CD}$ and $q_{H_4} + q_{CD}$ elements respectively). This gives us the announced bound

$$\frac{1}{e(q_E + 1)}\Big(\epsilon - \frac{q_S(q_S + q_{H_2})}{2^l} - \frac{q_{CD}(q_{H_3} + q_{CD})}{2^{k-1}}\Big).$$

The bound on the computation time derives from the fact that every request on H_1, H_2 and every signing request or key extraction request requires $\mathcal{B}$ to compute a scalar multiplication in $\mathbb{G}_1$. To handle confirmation/denial and signature queries, $\mathcal{B}$ can avoid pairing evaluations by pre-computing $\hat{e}(P, aP)$, $\hat{e}(P, bP)$, $\hat{e}(P, cP)$, $\hat{e}(aP, cP)$, $\hat{e}(aP, bP)$ and $\hat{e}(bP, cP)$ and performing exponentiations in $\mathbb{G}_2$. Each signing request thus requires an exponentiation in $\mathbb{G}_2$ while each confirmation/denial query entails a call to the decision oracle and 2 multi-exponentiations in $\mathbb{G}_2$ to simulate the confirmation/denial protocol. This yields the bound on $\mathcal{B}$'s running time. □

In order for the proof to hold, we must have $q_{H_2} \ll 2^l$, where l is the size of the random salt r. As usual with identity-based signatures, the reduction is not really efficient: for $l = 160$, if we take $q_E \approx q_{CD} \leq 2^{30}$ and $q_{H_3} < 2^{60}$, we can drop the negligible term $q_{CD}(q_{H_3} + q_{CD})2^{1-k} \approx 2^{-68}$ but we still end up with a bound $\epsilon' \geq 2^{-31}(\epsilon - 2^{-69})$. For technical reasons, we cannot use the technique of Katz and Wang [121] to get rid of the random salt $r \in \{0, 1\}^l$. Actually, it would be incompatible with the proof of theorem 6.2 which claims the scheme's invisibility in the sense of Galbraith and Mao (see [90]) under the Decisional Bilinear Diffie-Hellman assumption.

Theorem 6.2 *In the random oracle model, IBUS satisfies the invisibility property provided the decision Bilinear Diffie-Hellman problem is hard. More formally, if we assume that no algorithm is able to forge a signature in the game of definition 6.1 with a non-negligible probability and if a distinguisher $\mathcal{D}$ has a non-negligible advantage ϵ in the game of definition 6.2 when asking q_E key extraction queries, then there exists a distinguisher $\mathcal{B}$ that has an advantage*

$$\epsilon' \geq \frac{1}{e(q_E+1)}\Big(\epsilon - \frac{q_S(q_S+q_{H_2})+1}{2^l} - \frac{q_{CD}(q_{H_3}+q_{CD})}{2^{k-1}}\Big)$$

for the DBDH problem within a time bounded as in theorem 6.1 except that no decision oracle is used.

Proof. We assume there exists a distinguisher $\mathcal{D}$ that is able to decide whether a signature on a message was actually issued by a signer without the help of the latter. We show that such a distinguisher allows building a PPT algorithm $\mathcal{B}$ that is able to solve the Decisional Bilinear Diffie-Hellman problem with a non-negligible advantage by using $\mathcal{D}$ as a subroutine.

Let (P, aP, bP, cP, z) be a random instance of the problem. $\mathcal{B}$'s goal is to decide whether $z = \hat{e}(P,P)^{abc}$ or not. $\mathcal{B}$ plays the role of $\mathcal{D}$'s challenger in the game of definition 6.2. At the beginning of this game, $\mathcal{B}$ fixes the system-wide parameters as in the proof of theorem 6.1 with $P_{pub} = cP \in \mathbb{G}_1$. These system parameters are given to $\mathcal{D}$ that then performs a polynomially bounded number of queries as explained in definition 6.2. As in the proof of theorem 6.1, we assume that any signature query or confirmation/denial query on an identity is preceded by a H_1 oracle query on that identity. We now detail how $\mathcal{B}$ deals with queries made by $\mathcal{D}$. As in the proof of the previous theorem, $\mathcal{B}$ maintains lists L_{H_1}, L_{H_2} and L_{H_3} to keep track of the answers given to hash oracle queries.

- H_1-queries: are treated as in the proof of theorem 6.1.
- H_2-queries: at any time $\mathcal{D}$ can ask the hash value of a tuple (M, r, ID). When receiving such a query, $\mathcal{B}$ first checks if L_{H_2} contains a tuple $(M, r, \mathsf{ID}, d, Y)$ for some $d \in \mathbb{Z}_q$. If it does, $\mathcal{B}$ returns $dP \in \mathbb{G}_1$ if $Y = 0$ and $d(aP)$ if $Y = 1$. Otherwise, $\mathcal{B}$ picks a random $d \xleftarrow{R} \mathbb{Z}_q$, inserts the tuple $(M, r, \mathsf{ID}, d, 0)$ into L_{H_2} and returns $dP \in \mathbb{G}_1$ as an answer to the query.

- H_3 and H_4-queries are treated in the simplest way by uniformly sampling a random element from $\mathbb{Z}_q^*$.
- key extraction queries are handled as in theorem 6.1.
- signature queries are handled by $\mathcal{B}$ exactly as in the proof of theorem 6.1. As in the latter, $\mathcal{B}$ can always provide a consistent answer to this kind of query.
- confirmation/denial queries: at any time, $\mathcal{D}$ can produce a tuple (M, ID, σ), where $\sigma = \langle r, \gamma \rangle \in \{0,1\}^l \times \mathbb{G}_2$, and ask for a proof that σ is a valid or invalid signature on M for the signer of identity ID. Unlike what happens in the proof of theorem 6.1, $\mathcal{B}$ is able to provide $\mathcal{D}$ with a consistent view with overwhelming probability. Most of the time, it can reconstruct the legitimate signature $\langle r, \gamma' \rangle$ on M for the signer of identity ID with the random string r (this is due to the way that H_2 oracle queries are handled). The confirmation/denial protocol is simulated exactly as in the proof of theorem 6.1.

After a first series of queries, $\mathcal{D}$ produces a message M and an uncorrupted identity ID on which she wishes to be challenged. $\mathcal{B}$ then constructs a challenge signature as follows: it takes a random string $r \in \{0,1\}^l$ and checks if L_{H_2} contains a 4-uple $(M, r, \mathsf{ID}, .)$. If it does, $\mathcal{B}$ aborts (such an event has a probability smaller than $(q_S + q_{H_2})/2^l$ to happen). Otherwise, $\mathcal{B}$ defines the hash value $H_2(M, r, \mathsf{ID})$ to be $d(aP)$ for a randomly chosen $d \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$. The tuple $(M, r, \mathsf{ID}, d, 1)$ is then inserted into L_{H_2}. Because of the assumptions made above, a triple (ID, μ, X) must exist in L_{H_1} for some $\mu \in \mathbb{Z}_q$. If $X = 0$, then $\mathcal{B}$ stops and outputs "failure". Otherwise, $\mathcal{B}$ computes $\gamma = z^{d\mu} \in \mathbb{G}_2$ and sets the challenge signature as $\langle r, \gamma \rangle$. Clearly, if $\mathcal{D}$ is a good distinguisher and if z actually equals $\hat{e}(P,P)^{abc}$, then $\langle r, \gamma \rangle$ must appear as a valid signature for the pair (M, ID).

At the second stage of the game, $\mathcal{D}$ issues a second series of queries with the restriction that she is now disallowed to ask for the private key associated to ID or to perform a confirmation/denial query for the challenge $(M, \mathsf{ID}, \langle r, \gamma \rangle)$. The simulator $\mathcal{B}$ is able to handle confirmation or denial queries with overwhelming probability: any signature

$(M, \mathsf{ID}, \langle r, \gamma' \rangle)$ with $\gamma \neq \gamma'$ is declared invalid and the denial protocol is then simulated. The only case where $\mathcal{F}$ is provided with an inconsistent view on a confirmation/denial query is the situation where (P, aP, bP, cP, z) is not a DBDH tuple and where $\mathcal{D}$ queries the confirmation/denial oracle on a tuple $(M, \mathsf{ID}, \langle r, \gamma' \rangle)$ for which $\langle r, \gamma' \rangle$ is a legitimate signature on M for the identity ID. This event only occurs with negligible probability, since according to theorem 6.1, we assumed that no PPT algorithm can produce a valid signature for a chosen message on a chosen identity without knowing the private key.

Eventually, $\mathcal{D}$ outputs a bit b' that is 0 if she finds that $(M, \mathsf{ID}, \langle r, \gamma \rangle)$ is a valid tuple message-identity-signature and 1 if she finds that $\langle r, \gamma \rangle$ is a random element of the signature space. If $b' = 0$, then $\mathcal{B}$ outputs 1 as a result to indicate that (P, aP, bP, cP, z) is a valid DBDH tuple. If $b' = 1$, it outputs 0, meaning that z is a random element of $\mathbb{G}_2$. One can easily verify that, if $\mathcal{D}$ succeeds in distinguishing whether the challenge was an actual signature, then $\mathcal{B}$ succeeds in distinguishing DBDH tuples.

$\mathcal{B}$'s probability not to achieve a state of failure can be assessed in the same way as in theorem 6.1 and we find that, if ϵ denotes $\mathcal{D}$'s advantage as a distinguisher, then $\mathcal{B}$'s advantage in distinguishing DBDH tuples is at least

$$\epsilon' \geq \frac{1}{e(q_E + 1)} \Big(\epsilon - \frac{q_S(q_S + q_{H_2}) + 1}{2^l} - \frac{q_{CD}(q_{H_3} + q_{CD})}{2^{k-1}} \Big).$$

□

It is possible to directly show that the scheme also satisfies the anonymity property in the random oracle model under the DBDH assumption. However, since anonymity and invisibility are essentially equivalent, the anonymity of our signature derives from its invisibility property.

5. Conclusions

In this chapter, we showed a first construction for a provably secure identity-based undeniable signature and we extended the panel of primitives for identity-based cryptography. We provided a proof of existential unforgeability under the Gap Bilinear Diffie-Hellman assumption. Our construction and the underlying assumption for its security are inspired from those of Chaum-van Antwerpen [53] and Okamoto-Pointcheval

[168]. We also extended the notions of invisibility and anonymity of Galbraith and Mao [90] to the identity-based setting and we proved the invisibility of our scheme in the random oracle model under the Decisional Bilinear Diffie-Hellman assumption.

We leave as an open problem to find a scheme that satisfies the invisibility property and is tightly related to a weaker assumption than the hardness of the Gap-BDH problem.

CHAPTER 7

Joint Signature and Encryption in Identity-Based Cryptography

Abstract. This chapter provides an analysis of several existing protocols ensuring both confidentiality and authentication of messages in the context of identity-based cryptography. It first shows a security flaw in the first scheme of this kind proposed by Malone-Lee. It then presents a new construction and demonstrates its security in a formal security model. The chapter then presents two other improved constructions that were published after ours before describing a new provably secure scheme which is noticeably more efficient than all previous ones.

1. Identity-based signcryption

Two fundamental services of public key cryptography are privacy and authentication. Public key encryption schemes aim at providing confidentiality whereas digital signatures must provide authentication and non-repudiation. Nowadays, many real-world cryptographic applications require those distinct goals to be simultaneously achieved. This motivated Zheng [228] to provide the cryptographer's toolbox with a novel cryptographic primitive which he called 'signcryption'. The purpose of this kind of cryptosystem is to encrypt and sign data in a single operation which has smaller bandwidth requirements and computational costs than those entailed by doing both operations sequentially. Proper signcryption schemes should provide confidentiality as well as authentication and non-repudiation. As in conventional encryption schemes, recovering the plaintext from a signcrypted message must be computationally infeasible without the receiver's private key; as in conventional digital signatures, it must be computationally infeasible to create signcrypted texts without the private key of the sender.

Several identity-based signcryption (IBSC) algorithms have been proposed so far, e.g. [45, 58, 64, 65, 131, 139, 161, 189, 220]. Within

this handful of results, only [45, 58, 64, 65, 131, 220] consider schemes supported by formal models and security proofs in the random oracle model. Actually the first identity-based system simultaneously offering data privacy and sender authentication was suggested by Lynn [137] who extended the Boneh-Franklin scheme to build an efficient authenticated IBE. Unfortunately, his scheme does not provide the non-repudiation property as the recipient of a ciphertext is the only entity to be able to ascertain the origin and validity of the ciphertext. Malone-Lee proposed a method to overcome the latter limitation and achieved an identity-based signcryption solution [139].

One of the contribution of the present chapter is to pinpoint a security flaw in Malone-Lee's system and show how to fix it. Next to this result published at ITW'03, we propose another scheme which is immune to the attack that exists against the Malone-Lee scheme. This chapter also discusses about several papers that followed our work, including the IBSC scheme published by Boyen [45] at Crypto'03 and its improvements put forth by Chen and Malone-Lee [58].

Among all schemes supported by security proofs in formal security models, Chen and Malone-Lee's proposal [58] happens to be the most efficient construction. Another contribution of this chapter is to propose a new identity-based signcryption scheme that even supersedes [58] from an efficiency point of view at the expense of the security resting on stronger but reasonable assumptions.

The new construction was discovered in a joint work with Paulo Barreto and Noel McCullagh. We argue that it can benefit from the most efficient pairing calculation techniques for a larger variety of elliptic curves than previous schemes. Indeed, recent observations [203] pinpointed problems arising for many provably secure pairing-based protocols when implemented using asymmetric pairings and ordinary curves. Our proposal avoids those problems thanks to the fact that it does not require to hash onto an elliptic curve cyclic subgroup. It is actually obtained from an optimal randomness re-use in the DHI-IBS identity-based signature (IBS) that was studied in chapter 5. In an analysis of the scheme's efficiency, we show that it is more than twice as fast as Chen and Malone-Lee's one on the sender's side and about 33% more efficient for receivers. It does not enjoy all of the properties of the Chen-Malone-Lee proposal but we believe that it does suffice to implement

most practical applications.

This chapter is organized as follows. Section 2 presents theoretical concepts and formal security models that suitable IBSC schemes should comply with. The Malone-Lee IBSC is described and analyzed in section 3. We describe our IBSC scheme and prove its security in section 4. The improvements provided by Boyen and Chen-Malone-Lee are recalled in section 5. We then describe our faster scheme in section 6, and compare its efficiency to various schemes in section 6.3.

2. Formal models for identity-based signcryption

We here describe the formal structure that we shall use for identity-based signcryption (IBSC) schemes. We then discuss about related security notions that have been used in several papers in the literature.

2.1. General formalism

Definition 7.1 *Just like encryption-only identity-based systems, IBSC schemes are made of four algorithms which are the following.*

Setup: *is a probabilistic algorithm run by a private key generator (PKG) that takes as input a security parameter to output a master secret key* mk *and public parameters* params *that include a system-wide public key.*

Keygen: *is a key generation algorithm run by the PKG on input of* params *and the master key* mk *to return the private key* d_{ID} *associated to the identity* ID*.*

Sign/Encrypt: *is a probabilistic algorithm that takes as input public parameters* params*, a message* M*, the recipient's identity* ID_R*, and the sender's private key* d_{ID_S}*, and outputs a ciphertext* $\sigma = \mathsf{Sign/Encrypt}(M, d_{\mathsf{ID}_S}, \mathsf{ID}_R)$*.*

Decrypt/Verify: *is a deterministic decryption algorithm that takes as input a ciphertext* σ*, public parameters* params*, the receiver's private key* d_{ID_R} *and (optionally) a sender's identity* ID_S *before returning a plaintext* M *together with auxiliary information allowing a third party to be convinced of its origin or a distinguished symbol* $\perp$ *if* σ *does not properly decrypt into a message accompanied by suitable authenticating information for the sender* ID_S*.*

For consistency purposes, we impose the obvious constraint that if $\sigma = \mathsf{Sign/Encrypt}(m, d_{\mathsf{ID}_S}, \mathsf{ID}_R)$*, then the output of the* $\mathsf{Decrypt/Verify}(.)$ *for the triple* $(\sigma, \mathsf{ID}_S, d_{\mathsf{ID}_R})$ *contains a plaintext* m *and additional information allowing the receiver to convince a third party that the plaintext actually emanates from the sender.*

We mention the existence of schemes such as those proposed in [45, 58] where the Decrypt/Verify algorithm only takes as input the ciphertext and the recipient's private key: the sender's identity is recovered together with the message in the course of the decryption/verification process. This syntactical difference characterizes schemes in which ciphertexts are meant not to convey any clear information identifying their recipient or originator.

2.2. Security notions for IBSC schemes

Malone-Lee defines extended security notions for identity-based signcryption schemes (IBSC). These notions are chosen-ciphertext security (i.e. indistinguishability against adaptive chosen ciphertext attacks) and ciphertext unforgeability against adaptive chosen-message attacks.

Definition 7.2 *An identity-based signcryption scheme (IBSC) satisfies the* ***message confidentiality*** *property (or adaptive chosen-ciphertext security: IND-IBSC-CCA) if no PPT adversary has a non-negligible advantage in the following game.*

1. *The challenger runs the* Setup *algorithm on input of a security parameter* k *and sends the domain-wide parameters* params *to the adversary* $\mathcal{A}$*.*
2. *In a find stage,* $\mathcal{A}$ *starts probing the following oracles:*
   - *Keygen: returns private keys for arbitrary identities.*
   - *Sign/Encrypt: given a pair of identities* ID_S*,* ID_R *and a plaintext* M*, it returns an encryption under the receiver's identity* ID_R *of the message* M *signed in the name of the sender* ID_S*.*
   - *Decrypt/Verify: given a pair of identities* $(\mathsf{ID}_S, \mathsf{ID}_R)$ *and a ciphertext* σ*, it generates the receiver's private key* $d_{\mathsf{ID}_R} = \mathsf{Keygen}(\mathsf{ID}_R)$ *and returns either a pair* (M, s) *made of a plaintext* M *and transferable authenticating information*

for the sender's identity ID_S *or the* $\perp$ *symbol if* σ *does not properly decrypt under the private key* d_{ID_R}.

Once she decides that this stage is over, $\mathcal{A}$ *produces two plaintexts* $M_0, M_1 \in \mathcal{M}$ *and identities* ID_S^* *and* ID_R^*. *She may not have extracted the private key of* ID_R^* *and she obtains* $C = \mathsf{Sign/Encrypt}(M_b, d_{\mathsf{ID}_S^*}, \mathsf{ID}_R^*, \mathsf{params})$, *for a random* $b \stackrel{R}{\leftarrow} \{0,1\}$.

3. *In the guess stage,* $\mathcal{A}$ *issues new queries. This time, she may not issue a key extraction request on* ID_R^* *and she cannot submit* C *to the* Decrypt/Verify *oracle for the target identity* ID_R^*. *Finally,* $\mathcal{A}$ *outputs a bit* b' *and wins if* $b' = b$.

$\mathcal{A}$*'s advantage is defined as* $Adv(\mathcal{A}) := 2 \times \Pr[b' = b] - 1$.

It must be noted that the above definition considers insider attacks in the sense of [9]: the adversary may choose to be challenged on a sender's identity ID_S^* for which she previously learnt the matching private key or to request the latter at some point of the guess stage.

The original definition of message confidentiality given by Malone-Lee [139] does not allow the attacker to corrupt the private key of ID_S^* at any time. This enhancement was introduced by Boyen [45] for the first time. Its motivation is to devise schemes in which an attacker stealing the private key of a user is unable to gain any information about messages previously signed and encrypted by that user[1].

A scheme supporting the latter enhanced form of message confidentiality will be said to be *insider secure* in this chapter. In contrast, a scheme that is only secure in the sense of Malone-Lee's original definition [139] will be said to be *outsider secure*.

Regarding the properties of authentication and non-repudiation, the following definition was introduced in [139] to formalize the inability of any adversary to create a ciphertext containing a message authenticated by some user without knowing the latter's private key.

[1]This property was sometimes referred to as "forward security" in several papers in the literature. We here prefer using the terminology of 'insider security' introduced in [9] to avoid confusion with the quite different forward security property in the sense of [10] for key evolving protocols.

Definition 7.3 *An IBSC scheme is said to be* ***existentially ciphertext unforgeable*** *against adaptive chosen messages attacks (ECUF-IBSC-CMA) if no PPT adversary has a non-negligeable advantage in the following game.*

1. *The challenger runs the* Setup *algorithm with a security parameter k and gives the system parameters to the adversary.*
2. *The adversary $\mathcal{F}$ issues a polynomially bounded number of requests exactly as in the previous definition.*
3. *Finally, $\mathcal{F}$ produces a triple $(\sigma^*, \mathsf{ID}_S^*, \mathsf{ID}_R^*)$ that was not obtained from the* Sign/Encrypt *oracle during the game and for which the private key of ID_S^* was not exposed. The forger wins the game if the result of* Decrypt/Verify$(\sigma^*, d_{\mathsf{ID}_S^*}, \mathsf{ID}_R^*)$ *is not the $\perp$ symbol.*

The adversary's advantage is simply her probability of victory.

We note that the above definition also considers a form of 'insider security' in the sense of [9] since the adversary is allowed to expose the private key corresponding to the recipient's identity ID_R for which the produced ciphertext must be valid. This condition is necessary to obtain the non-repudiation property and to prevent a dishonest recipient to send a ciphertext to himself on Alice's behalf and attempt to convince a third party that Alice was the sender.

Nevertheless, it is natural to wonder whether it is really useful to require the notion of non-repudiation for a ciphertext rather than simply for the plaintext that it contains. Indeed, the former kind of authentication implies that the authenticating information returned by the Decrypt/Verify algorithm does not directly pertain to the clear message embedded in the ciphertext. This might render difficult the task of receivers who want to convince a third party of the sender's authorship for an extracted plaintext. For example, the original public key construction suggested by Zheng [228] in 1997 requires the receiver to produce a (possibly non-interactive) zero-knowledge proof of equality of discrete logarithms and such a requirement was subsequently showed to entail security concerns [110, 175].

On the other hand, if the recipient can extract a regular signature on the plaintext from the ciphertext, the non-repudiation property is easily obtained as noted by Boyen [45] at Crypto'03. The next definition,

given in [45], considers non-repudiation w.r.t. signatures embedded in ciphertexts rather than w.r.t. ciphertexts themselves.

Definition 7.4 *An IBSC scheme is said to be* ***existentially signature-unforgeable*** *against adaptive chosen messages attacks (ESUF-IBSC-CMA) if no PPT adversary can succeed in the following game with a non-negligible advantage:*

1. *The challenger runs the* Setup *algorithm on input k and gives the system-wide public key to the adversary $\mathcal{F}$.*
2. *$\mathcal{F}$ issues a number of queries as in definition 7.2.*
3. *Finally, $\mathcal{F}$ outputs a triple $(\sigma^*, \mathsf{ID}_S^*, \mathsf{ID}_R^*)$ and wins the game if the sender's identity ID_S^* was not corrupted and if the result of the* Decrypt/Verify *oracle on the ciphertext σ^* under the private key associated to ID_R^* is a valid message-signature pair (M^*, s^*) such that no* Sign/Encrypt *query involving M^*, ID_S^* and some receiver ID'_R (possibly different from ID_R^*) resulted in a ciphertext σ' whose decryption under the private key $d_{\mathsf{ID}'_R}$ is the alleged forgery $(M^*, s^*, \mathsf{ID}_S^*)$.*

The adversary's advantage is her probability of victory.

As stressed by [45], considering non-repudiation only w.r.t. signatures is useful if one is willing to devise schemes providing detachable signatures that should be *unlinkable* to the ciphertext they were conveyed in: anyone seeing a valid message-signature pair can use his/her private key to encrypt it into a valid signcryption under his/her public key as formalized by the following definition.

Definition 7.5 ([45]) *An IBSC scheme has the* ***ciphertext unlinkability*** *property if there exists a polynomial time algorithm that, given a receiver's private key d_{ID_R} and a message M bearing a signature s for some sender's name ID_S, can produce a valid ciphertext σ intended to ID_R in such a way that the private key d_{ID_R} decrypts σ into the signature pair (M, s) for the identity ID_S.*

Constructions satisfying the latter criterion allow a sender to always deny having created a given ciphertext even though he always remains committed to the content of the plaintext.

A complementary notion to the latter was termed *ciphertext authentication* and means that a receiver is always convinced that a ciphertext was (jointly) signed and encrypted by the same person.

Definition 7.6 *An IBSC scheme has the* ***ciphertext authentication*** *property against adaptive chosen messages and ciphertexts attacks (AUTH-IBSC-CMA) if no PPT adversary can succeed in the following game with a non-negligible advantage:*

1. *the challenger runs the* Setup *algorithm on input k and gives the system-wide public key to the adversary $\mathcal{F}$.*
2. *$\mathcal{F}$ issues a number of queries as in the previous definition.*
3. *Finally, $\mathcal{F}$ outputs a ciphertext σ^* together with a pair of identities $(\mathsf{ID}_S^*, \mathsf{ID}_R^*)$ and wins the game if the private keys of ID_S^* and ID_R^* were not extracted and if the ciphertext σ^* was not trivially obtained from a request to the* Sign/Encrypt *oracle.*

The adversary's advantage is again her probability of victory.

Intuitively, this notation is complementary to the one of *ciphertext unlinkability* in the sense that it prevents ciphertexts to be subject to man-in-the-middle attacks: no receiver can turn an extracted valid message-signature pair into a ciphertext intended to another receiver without knowing the latter's private key.

In [45], Boyen finally introduced a notion called *ciphertext anonymity* which is close to Bellare et al.'s definition of key privacy [25]. It formalizes the intuitive feature that ciphertexts convey no information about who their sender is nor about whom they are intended to.

Definition 7.7 ([45]) *An IBSC scheme has the* ***ciphertext anonymity*** *property against chosen-ciphertext attacks (or ANON-IBSC-CCA) if no PPT adversary has a non-negligible advantage in the following game.*

1. *The challenger runs the* Setup *algorithm on input of a security parameter k and sends the domain-wide parameters* params *to the adversary $\mathcal{A}$.*
2. *In a find stage, $\mathcal{A}$ starts probing the same oracles as in definition 7.2. Once she decides that this stage is over, $\mathcal{A}$ produces a plaintext $M \in \mathcal{M}$ and four identities $\mathsf{ID}_{S,0}^*, \mathsf{ID}_{S,1}^*$ and $\mathsf{ID}_{R,0}^*, \mathsf{ID}_{R,1}^*$ subject to the rule that she may not have extracted*

the private key of $\mathsf{ID}^*_{R,0}$ *nor* $\mathsf{ID}^*_{R,1}$. *She obtains the challenge* $C = \mathsf{Sign/Encrypt}(M, d_{\mathsf{ID}^*_{S,b}}, \mathsf{ID}^*_{R,b'}, \mathsf{params})$, *for independent random bits* $b, b' \stackrel{R}{\leftarrow} \{0,1\}$.

3. *In the guess stage,* $\mathcal{A}$ *makes new queries but may not issue a key extraction request on* $\mathsf{ID}^*_{R,0}$ *nor* $\mathsf{ID}^*_{R,1}$ *and she cannot submit* C *to the* Decrypt/Verify *oracle for identities* $\mathsf{ID}^*_{R,0}, \mathsf{ID}^*_{R,1}$. *Finally,* $\mathcal{A}$ *outputs a pair of bits* (d, d') *and wins if* $(b, b') = (d, d')$.

$\mathcal{A}$*'s advantage is defined as* $Adv(\mathcal{A}) := Pr[(b, b') = (d, d')] - 1/4$.

When simultaneously ensured, the properties of ciphertext unlinkability and anonymity may be useful for senders, acting as news correspondent in hostile area, who wish to authenticate confidential content without revealing anything about the channel that is used nor the moment or circumstances under which the transmission is taking place.

3. The Malone-Lee signcryption scheme and its (in)security

Setup: given a security parameter k, the PKG chooses symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of primer order $q > 2^k$, a generator P of $\mathbb{G}_1$, hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1$, $H_2 : \{0,1\}^* \to \mathbb{Z}^*_q$, $H_3 : \mathbb{G}_2 \to \{0,1\}^n$. It chooses a master-key $s \stackrel{R}{\leftarrow} \mathbb{Z}^*_q$ and the matching public key is $P_{pub} = sP$. The system-wide parameters are

$$\mathsf{params} := (\mathbb{G}_1, \mathbb{G}_2, n, \hat{e}, P, P_{pub}, H_1, H_2, H_3).$$

Keygen: given an identity ID, the private key is $d_{\mathsf{ID}} = sH_1(\mathsf{ID}) \in \mathbb{G}_1$.

Sign/Encrypt: given a plaintext M, a sender's private key d_{ID_R} and the receiver's public key $Q_{\mathsf{ID}_R} = H_1(\mathsf{ID}_R)$,

(1) Pick $x \stackrel{R}{\leftarrow} \mathbb{Z}^*_q$ to compute $U = xP \in \mathbb{G}_1$.
(2) Compute $r = H_2(M, U) \in \mathbb{Z}^*_q$ and $V = xP_{pub} + rd_{\mathsf{ID}_S} \in \mathbb{G}_1$.
(3) Scramble M into $W = M \oplus H_3(\hat{e}(P_{pub}, Q_{\mathsf{ID}_R})^x) \in \{0,1\}^n$.

The ciphertext is $\langle U, V, W\rangle \in \mathbb{G}_1 \times \mathbb{G}_1 \times \{0,1\}^n$.

Decrypt/Verify: given $\langle U, V, W\rangle$, the sender's identity $Q_{\mathsf{ID}_R} = H_1(\mathsf{ID}_R) \in \mathbb{G}_1$ and the receiver's private key d_{ID_R},

(1) Compute $M = W \oplus H_3(\hat{e}(U, d_{\mathsf{ID}_R}))$ and then $r = H_2(M, U)$.
(2) If $\hat{e}(P, V) = \hat{e}(U + rQ_{\mathsf{ID}_S}, P_{pub})$, return the plaintext M together with the pair $\langle U, V\rangle$. Otherwise, return $\perp$.

FIGURE 7.1. The ML-IBSC scheme

The consistency of the scheme stems from the relation

$$\hat{e}(U, d_{\mathsf{ID}_R}) = \hat{e}(xP, sQ_{\mathsf{ID}_R}) = \hat{e}(xsP, Q_{\mathsf{ID}_R}) = \hat{e}(xP_{pub}, Q_{\mathsf{ID}_R}).$$

Once the receiver has obtained M from the ciphertext $\langle U, V, W\rangle$, he can convince a third party that the message M emanates from the sender as we have

$$\hat{e}(P, V) = \hat{e}(P, rd_{\mathsf{ID}_S} + xP_{pub}) = \hat{e}(sP, xP + rQ_{\mathsf{ID}_S}) = \hat{e}(P_{pub}, U + rQ_{\mathsf{ID}_S})$$

In [139], Malone-Lee claimed that his scheme has the message confidentiality property (against outsider attacks) and gave informal security arguments. In a paper published at ITW'03 [131], we pinpointed a security flaw in the scheme which is even insecure against passive attacks where adversaries have no access to Sign/Encrypt or Decrypt/Verify oracles. The obvious reason is that a signature on the plaintext appears in the ciphertext. We indeed noticed that the scheme is the result of a combination of the simplified version of Boneh and Franklin's IBE cryptosystem (see chapters 2 and 3 for details: this version only has the IND-ID-CPA level of security) with the signature scheme below.

Setup and **Keygen** are the same as above.	
Sign: to sign a message M,	**Verify:** given $\sigma = \langle U, V\rangle$,
(1) Choose $x \xleftarrow{R} \mathbb{Z}_q^*$ and compute $U = xP$	(1) Compute $r = H(M, U)$
(2) Compute $r = H(M, U)$	(2) Accept the signature if
(3) Compute $V = xP_{pub} + rd_{\mathsf{ID}_S}$	$\hat{e}(P, V) = \hat{e}(P_{pub}, U + rQ_{\mathsf{ID}_S})$
The signature on M is $\sigma = \langle U, V\rangle$	

FIGURE 7.2. The IBS scheme underlying ML-IBSC

This signature is a variant of Heß's identity-based signature [111]. The ciphertexts produced by Malone-Lee's scheme may be thought of as a concatenation of a signature and a ciphertext (this approach is sometimes called "encrypt-and-sign" in the literature) except that they aim at taking advantage of a randomness re-use for the sake of efficiency. As a result, the scheme cannot achieve the semantic security as any attacker can simply verify the signature on both of the plaintexts M_0 and M_1 in the game of definition 7.2 and find out which one matches to the challenge ciphertext. Although, the scheme can offer a reasonable security

for some applications, it is not IND-IBSC-CCA (nor even IND-IBSC-CPA) secure in its current version. A similar observation was made in [196] regarding the Bao-Deng signcryption scheme [18] and another one suggested by Yum and Lee [221]. The authors of [196] pointed out that as soon as a signature on the plaintext is visible in the ciphertext, the scheme cannot be semantically secure. All systems resulting from the encrypt-and-sign approach have the same inherent weakness.

In the next section, we explain how to overcome this weakness and describe a signcryption scheme that is secure against (outside) chosen-ciphertext attacks.

4. A new identity-based signcryption scheme

In [228], Zheng showed how to use the randomness of DSA-like signatures for encryption purposes if both the sender and the receiver choose a public key within a cyclic prime order group. Unfortunately, his scheme does not offer non-repudiation. Bao and Deng [18] showed how to overcome the latter shortcoming but their modification renders the scheme semantically insecure as pinpointed in [196]. We here show that a variant of Heß's identity-based signature (which may be regarded as an adaptation of Schnorr's signature using the Tate pairing rather than the discrete exponentiation as a group isomorphism and was shown in [28] to be secure against existential forgery under adaptive chosen-message attacks in the random oracle model) can also be used as a building block to obtain an efficient and secure identity-based signcryption scheme.

4.1. Description of the scheme

The consistency follows from the bilinearity of the map. We have the relations

$$\begin{aligned} \hat{e}(P,S)\hat{e}(P_{pub},Q_{\mathsf{ID}_S})^r &= \hat{e}(P,P_{pub})^x \\ \hat{e}(S,Q_{\mathsf{ID}_R})\hat{e}(Q_{\mathsf{ID}_S},d_{\mathsf{ID}_R})^r &= \hat{e}(P_{pub},Q_{\mathsf{ID}_R})^x. \end{aligned}$$

Any third party (such as a firewall as explained in [95]) can be convinced of the message's origin by recovering k_1 and checking that $r = H(c, k_1)$ according to equation 2. The knowledge of the plaintext m is not required for the public verification of the origin of a ciphertext. On the other hand, it might be difficult for the receiver to convince a

Setup: given security parameters k and λ, the PKG chooses symmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of prime order $q > 2^k$, a generator P of $\mathbb{G}_1$, hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1$, $H_2 : \mathbb{G}_2 \to \{0,1\}^\lambda$, $H_3 : \{0,1\}^n \times \mathbb{G}_2 \to \mathbb{Z}_q^*$. It chooses a master-key $s \xleftarrow{R} \mathbb{Z}_q^*$, computes the matching public key $P_{pub} = sP$ and finally selects a symmetric encryption scheme (E, D) of key length λ. The system-wide parameters are

$$\mathsf{params} := (k, \lambda, \mathbb{G}_1, \mathbb{G}_2, n, \hat{e}, P, P_{pub}, H_1, H_2, H_3)$$

where n denotes the size of plaintexts.

Keygen: given an identity ID, the private key is $d_{\mathsf{ID}} = sH_1(\mathsf{ID}) \in \mathbb{G}_1$.

Sign/Encrypt: given a plaintext $m \in \{0,1\}^n$, his private key d_{ID_R} and the receiver's public key $Q_{\mathsf{ID}_R} = H_1(\mathsf{ID}_R)$, the sender

(1) picks $x \xleftarrow{R} \mathbb{Z}_q^*$ and computes $k_1 = \hat{e}(P, P_{pub})^x \in \mathbb{G}_2$, $k_2 = H_2(\hat{e}(P_{pub}, Q_{\mathsf{ID}_R})^x) \in \{0,1\}^\lambda$,

(2) computes $c = E_{k_2}(m) \in \{0,1\}^n$, $r = H_3(c, k_1) \in \mathbb{Z}_q^*$ and $S = xP_{pub} - rd_{\mathsf{ID}_S} \in \mathbb{G}_1$.

The ciphertext is $\langle c, r, S \rangle \in \{0,1\}^n \times \mathbb{Z}_q \times \mathbb{G}_1$.

Decrypt/Verify: given $\langle c, r, S \rangle$ and the sender's identity $Q_{\mathsf{ID}_R} = H_1(\mathsf{ID}_R) \in \mathbb{G}_1$ and the receiver's private key d_{ID_R}, the receiver computes

$$k_1' = \hat{e}(P, S)\hat{e}(P_{pub}, Q_{\mathsf{ID}_S})^r. \tag{2}$$

and rejects the ciphertext if $r \neq H_3(c, k_1')$. Otherwise, he computes $\tau = \hat{e}(S, Q_{\mathsf{ID}_R})\hat{e}(Q_{\mathsf{ID}_S}, d_{\mathsf{ID}_R})^r$, $k_2' = H_2(\tau) \in \{0,1\}^\lambda$ and the plaintext $m = D_{k_2'}(c) \in \{0,1\}^n$ is recovered. The receiver may then forward m together with the ciphertext $\langle c, r, S \rangle$ and the ephemeral symmetric encryption key k_2' to any third party that can verify that the ciphertext (using equation (2)) and the plaintext m (by checking that $m = D_{k_2'}(c)$) both emanate from the entity of identity ID_S.

FIGURE 7.3. The NewIBSC scheme

third party that the sender is the author of a particular plaintext m. To do this she has to forward the ephemeral decryption key k_2' to the third party. For security reasons, it is mandatory to use a symmetric encryption scheme (E, D) for which it is impossible to find a secret key mapping a chosen plaintext m to a given ciphertext c. Indeed, if the plaintext was simply scrambled by a hash value of $\hat{e}(P_{pub}, Q_{\mathsf{ID}_R})^x$ as

in the Malone-Lee scheme in the signcryption algorithm, the recipient would be unable to convince anyone that the plaintext was created by the sender (as revealing τ would disclose $\hat{e}(Q_{\mathsf{ID}_S}, d_{\mathsf{ID}_R})$ and compromise the secrecy of any subsequent communication between entities ID_S and ID_R). The plaintext non-repudiation procedure thus turns out to be somewhat problematic.

It is important to insist that replacing $r = H_3(c, k_1)$ with $r = H_3(m, k_1)$ in step 2 of the Sign/Encrypt algorithm would induce the same obstacle to the semantic security as in the Malone-Lee scheme. In [95], a similar modification was made to the Bao-Deng construction and it is likely that it enhances its security in the same way.

4.2. Efficiency

This scheme is almost as efficient as Malone-Lee's method (since the pairing $\hat{e}(P, P_{pub})$ does not depend on users or messages and can always be pre-computed) and it can be slightly more efficient when users often have to communicate between each other (pairings $\hat{e}(P_{pub}, Q_{\mathsf{ID}_B})$ and $\hat{e}(Q_{\mathsf{ID}_A}, d_{\mathsf{ID}_B})$ can be pre-computed by the sender and the receiver once and for all and cached in memory). In this case, the most expensive operations of the Sign/Encrypt algorithm are two exponentiations in $\mathbb{G}_2$ and a computation of the type $aP + bQ \in \mathbb{G}_1$. The Decrypt/Verify operation only requires two pairing evaluations and two exponentiations. With pre-computations, both schemes have a similar efficiency for the signature/encryption as well as for the decryption/verification procedure.

4.3. Security

The security proofs provided here assume that the Sign/Encrypt and Decrypt/Verify algorithms always take distinct identities as input. In other words, a sender can never jointly sign and encrypt a message for him/herself. A similar assumption was made by Boyen in his Crypto'03 paper where it was referred to as the "irreflexivity assumption".

4.3.1. Message confidentiality

Theorem 7.1 *In the random oracle model, we assume we have an IND-IDSC-CCA adversary called $\mathcal{A}$ that is able to distinguish ciphertexts during the game of definition 7.1 with an advantage ϵ when running*

in a time t and asking at most q_{H_i} queries to random oracles H_i (for $i = 1, \ldots, 3$), q_{ke} key extraction queries, q_{se} signature/encryption queries and q_{dv} queries to the Decrypt/Verify *oracle. Then, for any $0 \leq \nu \leq 1$, there either exists*

- *an algorithm $\mathcal{B}$ solving the Bilinear Diffie-Hellman problem in a time $t' < t + O(q_{se})t_p + O(q_{ke})t_m$ with an advantage*

$$\epsilon' > \frac{\epsilon - \nu}{(q_{H_2} + q_{se} + q_{dv})(q_{se} + q_{dv})\binom{q_{H_1}}{2}} \left(1 - \frac{(q_{se}+1)(q_{se}+q_{h_3})}{2^k}\right)$$

- *a passive (in the sense of definition 3.3) adversary $\mathcal{B}'$ that breaks the semantic security of the symmetric scheme within a time t' with an advantage ν*

where t_p denotes the computation time of the bilinear map and t_m stands for the cost of a scalar multiplication in $\mathbb{G}_1$.

Proof. Algorithm $\mathcal{B}$ receives a random instance (P, aP, bP, cP) of the Bilinear Diffie-Hellman problem and attempts to compute $\hat{e}(P, P)^{abc}$. It will run $\mathcal{A}$ as a subroutine and act as the challenger in the game of definition 7.2. In order to keep track of answers to $\mathcal{A}$'s queries to random oracles H_1, H_2 and H_3, $\mathcal{B}$ maintains lists L_1, L_2 and L_3 that are initially empty. We assume that any query to Sign/Encrypt or Decrypt/Verify involving a pair of identities happens after that $\mathcal{A}$ queried oracle H_1 on these identities. Random oracle queries on H_1 are also supposed to be distinct. Any key extraction query on an identity is assumed to come after a hash query on the same identity. We also assume that $\mathcal{A}$ never makes a Decrypt/Verify query on a ciphertext obtained from the Sign/Encrypt oracle (as, otherwise, the simulator can coherently answer the request using information stored when emulating the Sign/Encrypt oracle) and that both of the identities involved in the challenge step are submitted to oracle H_1 at some point of the simulation.

At the beginning of the game, $\mathcal{B}$ provides $\mathcal{A}$ with public parameters including $P_{pub} = cP$ (where c is unknown to $\mathcal{B}$). Then, $\mathcal{B}$ randomly chooses two distinct indexes $i, j \stackrel{R}{\leftarrow} \{1, \ldots, q_{H_1}\}$. The attacker $\mathcal{A}$ issues a polynomially bounded number of H_1-queries on identities of her choosing. At the i^{th} H_1-request, $\mathcal{B}$ answers by $H_1(\mathsf{ID}_i) = aP$. At the j^{th}, it responds with $H_1(\mathsf{ID}_j) = bP$. Since aP and bP belong to a random instance of the BDH problem, $\mathcal{A}$'s view will not be modified by

these changes. Hence, the private keys d_{ID_i} and d_{ID_j} (which are not computable by $\mathcal{B}$) are respectively acP and bcP. The solution of the BDH problem is thus given by $\hat{e}(Q_{\mathsf{ID}_i}, d_{\mathsf{ID}_j}) = \hat{e}(d_{\mathsf{ID}_i}, Q_{\mathsf{ID}_j})$. For requests $H_1(\mathsf{ID}_e)$ with $e \neq i, j$, $\mathcal{B}$ chooses $b_e \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$, stores the pair (ID_e, b_e) in list L_1 and returns $H_1(\mathsf{ID}_e) = b_e P$.

We now explain how the other kinds of requests are treated by $\mathcal{B}$.

$\mathbf{H}_2$-queries: for an input $g_e \in \mathbb{G}_2$, $\mathcal{B}$ searches list L_2 for a pair (g_e, R_e). If such a pair exists, $\mathcal{B}$ returns R_e, otherwise it returns a random $R_e \stackrel{R}{\leftarrow} \{0,1\}^n$ and inserts the pair (g_e, R) into L_2.

$\mathbf{H}_3$-queries: for an input (c_e, k_e): $\mathcal{B}$ checks if list L_3 contains a tuple (c_e, k_e, r_e). If it does, $\mathcal{B}$ returns r_e. Otherwise, it responds with $r \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and inserts the triple (c_e, k_e, r) into L_3.

Key extraction requests: when $\mathcal{A}$ asks for the private key of an identity ID_A, if $\mathsf{ID}_A = \mathsf{ID}_i$ or $\mathsf{ID}_A = \mathsf{ID}_j$, then $\mathcal{B}$ fails and stops. If $\mathsf{ID}_A \neq \mathsf{ID}_i, \mathsf{ID}_j$ then the list L_1 must contain a pair (ID_A, d) for some d (meaning that $H_1(\mathsf{ID}_A)$ was set to dP). The private key corresponding to ID_A is then $dP_{pub} = cdP$ which is returned to $\mathcal{A}$.

Sign/Encrypt queries: at any time, the attacker $\mathcal{A}$ may issue a signature/encryption request for a plaintext M and identities ID_A and ID_B. If $\mathsf{ID}_A \neq \mathsf{ID}_i, \mathsf{ID}_j$, $\mathcal{B}$ computes the private key d_{ID_A} associated to ID_A by running the key extraction algorithm and simply runs the algorithm $\mathsf{Sign/Encrypt}(M, d_{\mathsf{ID}_A}, Q_{\mathsf{ID}_B})$.

In the case $\mathsf{ID}_A = \mathsf{ID}_i$ or $\mathsf{ID}_A = \mathsf{ID}_j$ but $\mathsf{ID}_B \neq \mathsf{ID}_i, \mathsf{ID}_j$, $\mathcal{B}$ has to simulate the execution of $\mathsf{Sign/Encrypt}(M, d_{\mathsf{ID}_A}, Q_{\mathsf{ID}_B})$ as follows. It chooses $r \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and $S \stackrel{R}{\leftarrow} \mathbb{G}_1^*$ and computes $k' = \hat{e}(P, S)\hat{e}(P_{pub}, Q_{\mathsf{ID}_A})^r$, $\tau = \hat{e}(S, Q_{\mathsf{ID}_B})\hat{e}(Q_{\mathsf{ID}_A}, d_{\mathsf{ID}_B})^r$ where d_{ID_B} is the private key corresponding to ID_B ($\mathcal{B}$ may obtain it from the key extraction algorithm as $\mathsf{ID}_B \neq \mathsf{ID}_i, \mathsf{ID}_j$). It queries oracle H_2 for itself to obtain $k_2 = H_2(\tau)$ and computes $c = E_{k_2}(M)$. It then checks if L_3 already contains a triple (c, k', r') with $r' \neq r$. In this case, $\mathcal{B}$ fails (this happens with a probability smaller than $q_{se}(q_{h_3} + q_{se})$ throughout the entire simulation). Otherwise, $\mathcal{B}$ stores (c, k', r) in L_3 before returning the ciphertext (c, r, S) which appears valid from $\mathcal{A}$'s point of

view.

If $(\mathsf{ID}_A, \mathsf{ID}_B) = (ID_i, ID_j)$, $\mathcal{B}$ first chooses $r^* \overset{R}{\leftarrow} \mathbb{Z}_q^*$ and $S^* \overset{R}{\leftarrow} \mathbb{G}_1$. It computes

$$k_1' = \hat{e}(P, S^*)\hat{e}(P_{pub}, Q_{\mathsf{ID}_A})^{r^*} = \hat{e}(P, S^*)\hat{e}(cP, aP)^{r^*}$$

and picks $\tau^* \overset{R}{\leftarrow} \mathbb{G}_2$, $k_2' \overset{R}{\leftarrow} \{0,1\}^n$ and calculates $c^* = E_{k_2'}(M)$. It then checks if list L_3 already contains a triple (c^*, k_1', r') such that $r' \neq r^*$. If it does, $\mathcal{B}$ aborts and reports "failure". Otherwise, it stores (c^*, k_1', r^*) into L_3 and (τ^*, k_2') into L_2 and returns the ciphertext $\sigma^* = (c^*, r^*, S^*)$ to $\mathcal{A}$.

The latter will be unable to recognize that σ^* is not a properly signed and encrypted message unless she queries oracle H_2 on $\hat{e}(S^*, Q_{\mathsf{ID}_A})\hat{e}(P, P)^{abcr^*}$. Such an event would allow $\mathcal{B}$ to extract the solution $\hat{e}(P, P)^{abc}$ which it is looking for. In order to be able to take advantage of such an event, $\mathcal{B}$ stores a record (r^*, S^*) into a special list which we call "the critical list" L_C.

Decrypt/Verify queries: when $\mathcal{A}$ requires to decrypt and verify a ciphertext $\sigma' = (c', r', S')$ for identities ID_A and ID_B, the simulator $\mathcal{B}$ first checks its validity (which is publicly verifiable) by calling oracle H_3 on its own. The rejection symbol $\perp$ is answered to $\mathcal{A}$ if σ' does not pass the verification test. Otherwise, we observe that, if $(\mathsf{ID}_A, \mathsf{ID}_B) \neq (\mathsf{ID}_i, \mathsf{ID}_j)$ and $(\mathsf{ID}_A, \mathsf{ID}_B) \neq (\mathsf{ID}_j, \mathsf{ID}_i)$, $\mathcal{B}$ can compute the pairing value $\hat{e}(Q_{\mathsf{ID}_A}, d_{\mathsf{ID}_B})$ which is needed to coherently answer the query. We thus assume $(\mathsf{ID}_A, \mathsf{ID}_B) = (\mathsf{ID}_i, \mathsf{ID}_j)$ (the case $(\mathsf{ID}_A, \mathsf{ID}_B) = (\mathsf{ID}_j, \mathsf{ID}_i)$ is tackled with in the same way). The simulator $\mathcal{B}$ then picks a random $k_2' \overset{R}{\leftarrow} \{0,1\}^\lambda$ to compute $m' = D_{k_2'}(c')$ and returns m' to $\mathcal{A}$. Clearly, $\mathcal{A}$ will not recognize that m' is not the right plaintext extracted from σ' unless she queries H_2 on the input $\hat{e}(S', Q_{\mathsf{ID}_A})\hat{e}(P, P)^{abcr'}$ and such an event would reveal $\hat{e}(P, P)^{abc}$ to $\mathcal{B}$. The latter thus stores the pair (r', S') into the list L_C.

Once $\mathcal{A}$ decides to enter the challenge phase, she chooses a pair of uncorrupted target identities. With a probability at least $1/\binom{q_{H_1}}{2}$, this pair is $(\mathsf{ID}_i, \mathsf{ID}_j)$ (as we assume that $\mathcal{A}$ chooses to be challenged on a pair of

identities for which she asks for the hash value either before or after the challenge phase). If $\mathcal{A}$ does not select ID_i and ID_j as target identities, $\mathcal{B}$ fails.

When $\mathcal{A}$ produces her two plaintexts m_0 et m_1, $\mathcal{B}$ chooses a random bit $b \stackrel{R}{\leftarrow} \{0,1\}$. It also picks $r^* \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$, $S^* \stackrel{R}{\leftarrow} \mathbb{G}_1$ and $k_2' \stackrel{R}{\leftarrow} \{0,1\}^\lambda$ to compute

$$k_1' = \hat{e}(P, S^*)\hat{e}(P_{pub}, Q_{ID_A})^{r^*} = \hat{e}(P, S^*)\hat{e}(cP, aP)^{r^*},$$

and $c_b = E_{k_2'}(m_b)$. It then verifies as above if L_3 already contains an entry (c_b, k_1', r') such that $r' \neq r^*$. If yes, $\mathcal{B}$ aborts and declares "failure" (this happens with a probability smaller than $(q_{se} + q_{h_3})/2^k$). Otherwise, the tuples (c_b, k_1', r^*) and (r^*, S^*) are respectively stored into L_3 and L_C. The ciphertext $\sigma^* = (c_b, r^*, S^*)$ is then returned as a challenge.

$\mathcal{A}$ then performs a second series of queries which is treated in the same way as the first one. At the end of the simulation, she produces a bit b' which is ignored. To produce a result, $\mathcal{B}$ fetches random pairs (τ^*, k_2^*) and (r^*, S^*) respectively from lists L_2 and L_C to compute $(\tau^*/\hat{e}(S^*, bP))^{1/r^*}$ which is produced as a result and matches $\hat{e}(P,P)^{abc}$ with probability

$$\frac{1}{(q_{H_2} + q_{se} + q_{dv})(q_{se} + q_{dv})}$$

(as L_2 and L_C always contain respectively less than $1/(q_{H_2} + q_{se} + q_{dv})$ and $1/(q_{se} + q_{dv})$ elements by construction). Indeed, provided the simulation is indistinguishable from a real game, if $\mathcal{A}$ never issues an H_2-query that pertains to $\hat{e}(Q_{\mathsf{ID}_i}, d_{\mathsf{ID}_j}) = \hat{e}(P,P)^{abc}$, the only way for her to guess the hidden bit b is to succeed in a chosen-plaintext attack on the symmetric encryption scheme (E, D). In more details, let Fail denote the event that $\mathcal{B}$ fails and Ask be the event that a H_2-query related to $\hat{e}(Q_{\mathsf{ID}_i}, d_{\mathsf{ID}_j})$ is issued at some point. When using the notation $\mathrm{pr}[A]$ to denote the conditional probability $\Pr[A|\neg\mathsf{Fail}]$, we can write

$$\begin{aligned} \mathrm{pr}[b' = b] &= \mathrm{pr}[b' = b|\mathsf{Ask}]\mathrm{pr}[\mathsf{Ask}] + \mathrm{pr}[b' = b|\neg\mathsf{Ask}]\mathrm{pr}[\neg\mathsf{Ask}] \\ &\leq \mathrm{pr}[\mathsf{Ask}] + \mathrm{pr}[b' = b|\neg\mathsf{Ask}](1 - \mathrm{pr}[\mathsf{Ask}]). \end{aligned}$$

We note that $\mathrm{pr}[b' = b|\neg\mathsf{Ask}] = (\nu + 1)/2$ is nothing but $\mathcal{A}$'s probability to succeed in a chosen-plaintext attack against the symmetric scheme (E, D) for a random key. It then comes that

$$\frac{\epsilon + 1}{2} \leq \frac{\nu + 1}{2} + \frac{1 - \nu}{2}\mathrm{pr}[\mathsf{Ask}] \leq \frac{\nu + 1}{2} + \frac{1}{2}\mathrm{pr}[\mathsf{Ask}]$$

which yields $\text{pr}[\mathsf{Ask}] \geq \epsilon - \nu$ and thus $\Pr[\mathsf{Ask} \wedge \neg\mathsf{Fail}] \geq (\epsilon - \nu)\Pr[\neg\mathsf{Fail}]$.

We now have to assess $\mathcal{B}$'s probability of success. We observed that $\mathcal{B}$ fails in the following situations

- E_1**:** $\mathcal{A}$ asks for the private key associated to ID_i or ID_j before the challenge phase.
- E_2**:** $\mathcal{A}$ does not choose the pair $(\mathsf{ID}_i, \mathsf{ID}_j)$ as target identities.
- E_3**:** a collision on H_3 occurs when $\mathcal{B}$ at a $\mathsf{Sign}/\mathsf{Encrypt}$ query or at the challenge phase.

We have $\Pr[E_3] \leq (q_{se}+1)(q_{se}+q_{h_3})/2^k$. We already observe that $\Pr[\neg E_2] = \binom{q_{h_1}}{2}$ and we note that $\neg E_2$ implies $\neg E_1$. We thus find

$$\Pr[\neg\mathsf{Fail}] = \Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3] = \Big(1 - \frac{(q_{se}+1)(q_{se}+q_{h_3})}{2^k}\Big)\frac{1}{\binom{q_{H_1}}{2}}$$

which yields the announced bound. □

The obtained bound on algorithm $\mathcal{B}$'s advantage is quite loose. We note that, in terms of tightness, a more efficient (but still loose) reduction can be achieved under the stronger Decisional Bilinear Diffie-Hellman assumption recalled in chapter 1. We do not give the details here.

4.3.2. Ciphertext unforgeability

Setup and **Keygen** are the same in our scheme.	
Sign: to sign a message M,	**Verify:** given $\sigma = \langle r, S\rangle$,
(1) Choose $x \xleftarrow{R} \mathbb{Z}_q^*$ and compute $u = \hat{e}(P,P)^x$	(1) Compute $u' = \hat{e}(P,S)\hat{e}(P_{pub}, Q_{\mathsf{ID}_S})^r$
(2) Compute $r = H_3(M,u)$	(2) Accept the signature if $r = H_3(M,u')$
(3) Compute $S = xP - rd_{\mathsf{ID}_S}$	
The signature on M is $\sigma = \langle r, S\rangle$	

FIGURE 7.4. The Heß-IBS scheme

The ciphertext unforgeability property appears to directly derive from the existential unforgeability of Heß's identity-based signature (for which formal proofs were given in [111, 81, 28]). Indeed, in a ciphertext (c, r, S), the components (r, S) are a Heß-like signature on the encrypted message c (though the scheme is different from a basic encrypt-then-sign composition in that the symmetric encryption is performed using a key derived from a quantity depending on the same randomness as the one used in the signature part (r, S)). Forging a ciphertext (c, r, S) in the

name of an uncorrupted signer's identity thus looks like producing a message-signature pair for Heß's signature.

However, we have to formally establish this fact in the sense of definition 7.3. To do so, we adopt the same proof strategy as Cha and Cheon [51] who first show that a successful adversary in the game of definition 7.3 implies a forger in a weaker attack where the adversary is challenged on a given sender's identity selected by the challenger at the outset of the game.

Lemma 7.1 *If there is a forger $\mathcal{A}_0$ for an adaptively chosen message and identity attack having advantage ϵ_0 over our IBSC scheme when running in a time t_0 and making q_{h_i} queries to random oracle h_i ($i = 1, 2, 3$), then there exists an algorithm $\mathcal{A}_1$ for an adaptively chosen message and given identity attack which has advantage $\epsilon_1 \geq \epsilon_0(1 - \frac{1}{2^k})/q_{h_1}$ within a running time $t_1 \leq t_0$. Moreover, $\mathcal{A}_1$ performs the same number of key extraction queries, signature/encryption queries and H_2, H_3-queries as $\mathcal{A}_0$ does.*

Again, the proof is quite similar to Cha and Cheon's one [51]. In a second step, lemma 7.2 shows that an adversary in the weaker attack scenario implies an algorithm solving the Diffie-Hellman problem in the random oracle model. The proof uses the forking lemma [182, 183].

Lemma 7.2 *Let us assume that there is an adaptively chosen message and given identity attacker $\mathcal{F}$ that makes q_{h_i} queries to random oracles H_i ($i = 1, 2, 3$) and q_{se} queries to the signature/encryption oracle and q_{dv} calls to the decryption/verification oracle. Assume that, within a time t, $\mathcal{F}$ produces a forgery with probability $\epsilon \geq 10(q_{se} + 1)(q_{se} + q_{h_2})/2^k$. Then, there exists an algorithm $\mathcal{B}$ that is able to solve the CDHP in $\mathbb{G}_1$ within an expected time*

$$t' \leq 120686 q_{h_3}(t + O((q_{se} + q_{dv})\tau_p))/\epsilon$$

where τ_p is the cost of a pairing evaluation.

Proof. Algorithm $\mathcal{B}$ takes as input a random Diffie-Hellman instance $(A = aP, b = bP) \in \mathbb{G}_1 \times \mathbb{G}_1$. It starts $\mathcal{F}$ on the system-wide public key $P_{pub} = aP$ and for a fixed sender's identity ID_S. We first show how it can provide $\mathcal{F}$ with a consistent environment and we then explain how

it uses the forking lemma to extract $abP \in \mathbb{G}_1$. Adversarial queries are dealt with as follows.

$\mathbf{H}_1$-queries: for an identity $\mathsf{ID}_i \in \{0,1\}$ (we let ID_i denote the input of the i^{th} H_1-query), $\mathcal{B}$ returns bP if $\mathsf{ID}_i = \mathsf{ID}_S$. Otherwise, it chooses $\mu_i \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and returns $\mu_i P$.

$\mathbf{H}_2$-queries: for an input $g_e \in \mathbb{G}_2$, $\mathcal{B}$ returns the previously defined value if it exists and a random string $R_e \stackrel{R}{\leftarrow} \{0,1\}^n$ otherwise.

$\mathbf{H}_3$-queries: for an input (c,k): $\mathcal{B}$ checks if list L_3 contains a triple (c,k,r). If it does, $\mathcal{B}$ returns r. Otherwise, it responds with a random $r \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and inserts (ID, c, k, r) into L_3.

Key extraction requests: when $\mathcal{F}$ asks for the private key of ID_i (which differs from ID_S), $\mathcal{B}$ retrieves the value μ_i for which $H_1(\mathsf{ID}_i) = \mu_i P$ and returns $\mu_i P_{pub} = \mu_i(aP)$.

Sign/Encrypt queries: at any time, the attacker $\mathcal{A}$ may issue a signature/encryption request for a plaintext M and a pair of identities ID_A and ID_B. If $\mathsf{ID}_A \neq \mathsf{ID}_S$, $\mathcal{B}$ knows the private key d_{ID_A} corresponding to ID_A and can simply run the Sign/Encrypt algorithm according to its specification. We thus assume $\mathsf{ID}_A = \mathsf{ID}_S$. Because of the irreflexivity assumption, we know that $\mathsf{ID}_B \neq \mathsf{ID}_S$ and $\mathcal{B}$ thus knows the matching private key d_{ID_B} for ID_B. It can thus compute $\gamma = \hat{e}(Q_{\mathsf{ID}_A}, d_{\mathsf{ID}_B}) = \hat{e}(bP, d_{\mathsf{ID}_B})$. It then simulates the Sign/Encrypt algorithm by choosing $r, S \stackrel{R}{\leftarrow} \mathbb{Z}_q^* \times \mathbb{G}_1$ and computing $\tau = \hat{e}(S, Q_{\mathsf{ID}_B})\gamma^r$, $k_1 = \hat{e}(P,S)\hat{e}(P_{pub}, Q_{\mathsf{ID}_A})^r$. It obtains $k_2 = H_2(\tau)$ through simulation of H_2, computes $c = E_{k_2}(M)$ and backpatches to set the hash value $H_3(c, k_1)$ to r (the simulation fails if H_3 is already defined but such an event occurs with negligible probability taken into account by the bounds given by the forking lemma). The ciphertext (c, r, S) is then returned to $\mathcal{F}$.

Decrypt/Verify queries: when $\mathcal{A}$ requires to decrypt and verify a ciphertext $\sigma' = (c', r', S')$ for identities ID_A and ID_B, the simulator $\mathcal{B}$ first checks its publicly verifiable validity by calling oracle H_3 on its own. A rejection notification is sent to $\mathcal{A}$ if σ' does not pass the verification test. Otherwise, we observe that, if $\mathsf{ID}_B \neq \mathsf{ID}_S$, $\mathcal{B}$ can answer the query with probability

1 as it knows the receiver's private key d_{ID_B}. We thus assume $\mathsf{ID}_B = \mathsf{ID}_S$ and hence $\mathsf{ID}_A \neq \mathsf{ID}_S$ (thanks to the irreflexivity assumption) which implies that $\mathcal{B}$ knows the sender's private key d_{ID_A} and can calculate $\gamma = \hat{e}(d_{\mathsf{ID}_A}, Q_{\mathsf{ID}_B}) = \hat{e}(d_{\mathsf{ID}_A}, bP)$, $\tau' = \hat{e}(S', Q_{\mathsf{ID}_B})\gamma^{r'}$, obtain $k_2' = H_2(\tau')$ via simulation of H_2 before finishing the job by returning $m = D_{k_2'}(c)$.

We have showed how to simulate $\mathcal{F}$'s environment in a chosen-message and given identity attack scenario. We are ready to apply the forking lemma as a forged ciphertext (c, r, S) produced by the forger for a receiver's identity can be parsed into a signature (r, S) onto the message c. Let us assume that the attacker $\mathcal{F}$ forges a ciphertext (c, r, S) for a recipient's identity ID_R in a time t with probability $\epsilon \geq 10(q_{se}+1)(q_{se}+q_{h_3})/2^k$ when making q_{se} signature/encryption requests and q_{h_3} random oracle calls. As all oracles can be simulated without knowing the private key of the challenge identity ID_S, then there exists a Turing machine $\mathcal{F}'$ that uses $\mathcal{F}$ to produce two valid signatures (c, k_1, r_1, S_1), (c, k_2, r_2, S_2) for the same receiver's identity ID_R[2], where $k_1 = k_2 = \hat{e}(P, S_i)\hat{e}(P_{pub}, Q_{\mathsf{ID}_S})^{r_i}$ for $i = 1, 2$ and with $r_1 \neq r_2$, in expected time $t' \leq 120686 q_{h_3} t/\epsilon$.

The verification equation then leads us to write

$$\hat{e}(P, S_1)\hat{e}(P_{pub}, Q_{\mathsf{ID}_S})^{r_1} = \hat{e}(P, S_2)\hat{e}(P_{pub}, Q_{\mathsf{ID}_S})^{r_2}$$

which implies $\hat{e}(P, (r_2 - r_1)^{-1}(S_1 - S_2)) = \hat{e}(P_{pub}, Q_{\mathsf{ID}_S}) = \hat{e}(aP, bP)$. The simulator $\mathcal{B}$ finally extracts $abP = (r_2 - r_1)^{-1}(S_1 - S_2) \in \mathbb{G}_1$ as in the security proofs of Heß's signature [111, 81, 28]. □

When combined, lemmas 7.1 and 7.2 establish the following theorem.

[2]Actually, $\mathcal{F}'$ might produce forgeries for distinct receivers' identities but the deterministic behaviour of $\mathcal{F}$ before her crucial H_3-query (ensured by identical answers to previous H_3-queries and the identical random tape in the replay phase) implies that, in the replay phase, the crucial H_3-query is made for the same c (and thus the same ID_R) as in the original run. Both forgeries will be related to the same ID_R with identical probabilities to those provided by the probabilistic analysis of the forking lemma.

Theorem 7.2 *Let us assume that there is an adaptively chosen message and identity attacker $\mathcal{F}$ that makes q_{h_i} queries to random oracles H_i $(i = 1, 2, 3)$ and q_{se} queries to the signature/encryption oracle and q_{dv} calls to the decryption/verification oracle. Assume that, within a time t, $\mathcal{F}$ produces a forgery with probability $\epsilon \geq 10(q_{se} + 1)(q_{se} + q_{h_2})/2^k$. Then, there exists an algorithm $\mathcal{B}$ that is able to solve the CDHP in $\mathbb{G}_1$ within an expected time*

$$t' \leq 120686 q_{h_1} q_{h_3} (t + O((q_{se} + q_{dv})\tau_p))/\big(\epsilon(1 - \frac{1}{2^k})\big)$$

where τ_p stands for the cost of a pairing evaluation.

4.4. Limitations of the scheme

We showed how to overcome the security problems encountered in Malone-Lee's scheme and our scheme turned out to be the first identity-based signcryption scheme ensuring chosen-ciphertext security in the sense of definition 7.2 with the restriction that it only withstands outsider attacks (that is, the adversary is not allowed to expose the sender's private key under which the challenge ciphertext is created). This is the first limitation of our scheme. The second and main one is the problematic non-repudiation procedure for a receiver wishing to prove to a third party that the sender is the author of a given plaintext. The problem is that nothing prevents a malicious receiver to forward the ciphertext (c, r, S) together with a random symmetric key k_2' in an attempt to convince a third party that the sender created the message $m' = D_{k_2'}(c)$. It might be harmless for the sender as m' is very likely to be a meaningless message. In order to instantiate the system with a simple symmetric scheme such as a one-time pad (i.e. a symmetric scheme $E_{k_2}(m) = m \oplus k_2$), special precautions are needed. In this case the symmetric key should be obtained as $k_2 = H_2(H_2(\hat{e}(P_{pub}, Q_{\mathsf{ID}_R})))$ and the receiver has to transmit $k_2' = H_2(\tau)$ to a third party which then checks that $m = D_{H_2(k_2')}(c)$ in the non-repudiation procedure. This modification prevents a cheating receiver to convince a judge of the sender's authorship for an arbitrarily chosen message she did not send. Indeed, framing the sender for a chosen message would lead the dishonest receiver to invert H_2.

Another shortcoming is that it does not provide anonymous ciphertexts in the sense of definition 7.7 as their origin is publicly verifiable.

The latter feature might be a desirable property in some situations where firewalls are required to ascertain that encrypted data emanate from an authorized sender without learning anything about their content as exemplified in [95]. However, this property complicates the task of a receiver wishing to convince a third party of the sender's authorship of the plaintext. The inherent nature of the scheme therefore leads us to consider the non-repudiation property for entire ciphertexts instead of just the plaintext they contain. As a result, senders do not enjoy the ability to deny being the author of issued ciphertexts as they are in the scheme of [45].

Actually, the construction proposed by Boyen [45] overcomes all of the aforementioned limitations present in our scheme. Those improvements are obtained without noticeable additional computational costs w.r.t. to our scheme. In order to allow the reader to concretely figure out the relative efficiencies of both schemes, the next subsection provides a formal description of Boyen's IBSC.

5. Boyen's scheme and the Chen-Malone-Lee variant

The formal description given here is the one provided by Boyen in his Crypto'03 paper and it does not perfectly comply with the syntax that we set out in definition 7.1. Indeed, the scheme was presented in [45] as a two-layer design of probabilistic signature generation followed by a deterministic and randomness re-using encryption algorithm. Such a description has the advantage of clearly showing how the scheme was constructed from the Cha-Cheon IBS [51] through a secure randomness re-use in encryption purposes. Nevertheless, in security proofs of all IBSC schemes (including Boyen's one), an attacker can never observe data transiting from the signature layer to its encryption successor in the middle of a signature/encryption process: she always queries the Sign/Encrypt and Decrypt/Verify oracles as monolithic oracles and never issues inquiries to a signature or a decryption oracle individually. That is why we prefer using our single-layer formalism captured by definition 7.1 in the forthcoming sections although the present paragraph reproduces Boyen's original presentation. After all, the protocol can be described using our syntax as well (with the difference that the Decrypt/Verify algorithm would only take as inputs the receiver's private key and the ciphertext since the sender's identity is retrieved along the course of the

Setup: is the same as in Boneh and Franklin's IBE except that five hash functions are needed: $H_0 : \{0,1\}^* \to \mathbb{G}_1$, $H_1 : \{0,1\}^{n_0} \times \mathbb{G}_1 \to \mathbb{Z}_q^*$, $H_2 : \mathbb{G}_2 \to \{0,1\}^\ell$, $H_3 : \mathbb{G}_2 \to \mathbb{Z}_q^*$ and $H_4 : \mathbb{G}_1 \to \{0,1\}^{n_0+n_1}$ where ℓ is the size of the representation of $\mathbb{G}_1$'s elements, n_0 is the size of plaintexts and n_1 the maximum length of identifiers.

Keygen: given $\mathsf{ID} \in \{0,1\}^{n_1}$ and $s \in \mathbb{Z}_q^*$, return $d_{\mathsf{ID}} = sH_0(\mathsf{ID}) \in \mathbb{G}_1$.

Sign: given a message M and the sender's private key d_{ID_S},

(1) Pick $r \xleftarrow{R} \mathbb{Z}_q^*$, compute $U = rQ_{\mathsf{ID}_S} = rH_0(\mathsf{ID}_S)$, $h = H_1(M, U) \in \mathbb{Z}_q^*$ and $V = (r+h)d_{\mathsf{ID}_S} \in \mathbb{G}_1$.
(2) Return $\langle M, r, U, V, \mathsf{ID}_S, d_{\mathsf{ID}_S}\rangle$ to algorithm **Encrypt**.

Encrypt: given $\langle M, r, U, V, \mathsf{ID}_S, d_{\mathsf{ID}_S}\rangle$ and the recipient's identity $Q_{\mathsf{ID}_R} = H_0(ID_R) \in \mathbb{G}_1$,

(1) Compute $\omega = \hat{e}(d_{\mathsf{ID}_S}, Q_{\mathsf{ID}_R})$ and $\lambda = H_3(\omega) \in \mathbb{Z}_q^*$.
(2) Set $X = \lambda U \in \mathbb{G}_1$ and $Y = V \oplus H_2(\omega^{\lambda r}) \in \{0,1\}^\ell$.
(3) Compute $Z = (M||\mathsf{ID}_S) \oplus H_4(V) \in \{0,1\}^{n_0+n_1}$.
(4) Return the final ciphertext $\langle X, Y, Z\rangle$.

Decrypt: given $\langle X, Y, Z\rangle$ a private key d_{ID_R}

(1) Compute $V = Y \oplus H_2(\hat{e}(X, d_{\mathsf{ID}_R}))$ and then $(M||\mathsf{ID}_S) = Z \oplus H_4(V)$.
(2) Compute $Q_{\mathsf{ID}_S} = H_0(\mathsf{ID}_S) \in \mathbb{G}_1$, $\omega = \hat{e}(Q_{\mathsf{ID}_S}, d_{\mathsf{ID}_R})$ and then $\lambda = H_3(\omega) \in \mathbb{Z}_q^*$.
(3) Compute $U = \lambda^{-1}X \in \mathbb{G}_1$ and return $\langle M, \mathsf{ID}_S, U, V\rangle$ to the **Verify** algorithm.

Verify: given $\langle M, \mathsf{ID}_S, U, V\rangle$,

(1) Compute $Q_{\mathsf{ID}_S} = H_0(\mathsf{ID}_S) \in \mathbb{G}_1$.
(2) Return 1 if $\hat{e}(P, V) = \hat{e}(P_{pub}, U + hQ_{\mathsf{ID}_S})$ and 0 otherwise.

FIGURE 7.5. The Boyen IBSC scheme

decrypt/verify operation). Finally, similarly to other schemes studied further in this chapter, the monolithic description still enables a standard IBS on the plaintext to be detached from the ciphertext so that non-repudiation issues for extracted plaintexts can always be easily settled by third parties.

We observe that the scheme has about the same efficiency as ours: a single pairing must be computed by the Sign/Encrypt algorithm while four pairing calculations are needed upon decryption/verification.

Unlike our scheme, the present one offers message confidentiality even against insider attacks (that is, security in the strongest sense of definition 7.2): an attacker exposing a sender's private key is provably unable to gain any information about messages that were previously signed and encrypted by that sender.

Contrary to our scheme, the non-repudiation property is only required for plaintexts embedded in ciphertexts in accordance with definition 7.4. Although the receiver is convinced that the message was signed and encrypted by the same person throughout a single signature/encryption operation (and was not subject to a kind of man-in-the-middle attack), he/she is unable to convince any third party of this fact: in accordance with definition 7.5, the sender can always deny having produced the ciphertext as the receiver might have encrypted a message-signature pair to himself using his private key.

At PKC'05, Chen and Malone-Lee [58] published a simplification of Boyen's construction. Their simplification resides in the Encrypt and Decrypt algorithms. The former produces ciphertexts of the form $\langle U, W, Z\rangle$, where $U = rQ_{\mathsf{ID}_A}$, $W = V \oplus H_2(\omega^r)$ and $Z = (M||\mathsf{ID}_S) \oplus H_4(V)$. The latter avoids a pairing calculation at step (2) and a scalar multiplication at step (3). In other words, they removed a layer in the encryption operation and leave ciphertexts and embedded signatures share a common component U.

A consequence of their modification is that the ciphertext unlinkability property is achieved in a weaker sense than in [45]: an external observer is able to decide with a better probability than $1/2$ whether or not a given ciphertext encrypts a given message-signature pair since the ciphertext and the extracted signature share a common component. Nevertheless, the scheme still satisfies the criterion of definition 7.5 and the sender can still deny being the author of a specific ciphertext.

6. A fast identity-based signcryption scheme

This section presents the scheme that we discovered with Paulo Barreto and Noel McCullagh [21] and which we name FastIBSC. It is obtained by suitably combining the DHI-IBS signature scheme that we studied in chapter 5 with a simplified version of the Sakai-Kasahara IBE [189]. We are able to jointly sign and encrypt a plaintext without computing any pairing whereas the decryption/verification algorithm only

entails two pairing calculations.

Our scheme thus outperforms all previously known IBSC constructions: the signature/encryption algorithm performs more than twice as fast as in the Chen-Malone-Lee scheme whereas the Decrypt/Verify algorithm gains about 33% of efficiency if both schemes are implemented using similar parameters and algorithms (see section 6.3 for details). Besides, unlike previous IBSC systems, ours does not require to hash onto cyclic elliptic curve subgroups. As argued in section 6.3, it thus avoids problems that might prevent other IBSC schemes to benefit from the most optimized pairing calculation algorithms for ordinary curves when they are implemented using asymmetric pairings (unless one accepts to leave their security proofs rely on somewhat unnatural assumptions).

We actually propose two variants of FastIBSC. The first one works with named ciphertexts, meaning that encrypted message-signature pairs have to be accompanied with their sender's identity. It thus fail to satisfy the property of ciphertext anonymity formalized in definition 7.7. For applications that would require the latter property, we show a variant of FastIBSC that enjoys the same efficiency and supports anonymous ciphertexts in which the sender's identity is masked together with the signed plaintext. The disadvantage of the anonymous variant of FastIBSC is that its security reductions are significantly more expensive than those of the original FastIBSC.

To be fair, we mention that FastISBC does not feature all of the properties of Boyen or Chen-Malone-Lee's systems. For example, it does not have the ciphertext unlinkability property (see definition 7.5): given a correct message-signature pair, it seems infeasible for anyone to use his private key to embed it into a proper ciphertext addressed to himself. Nevertheless, we believe that FastIBSC does satisfy the main requirements that are desired in real-life applications. Its great efficiency thus renders it more than interesting for identity-based cryptography.

6.1. The scheme

Our construction consists in using a hash value of the commitment part of DHI-IBS to hide the plaintext in such a way that a single additional scalar multi-exponentiation is needed to ensure the confidentiality of the message in accordance with the model of definition 7.3.

If required, the anonymity property is obtained by scrambling the

Setup: given k, the PKG chooses *asymmetric* bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ of prime order $q > 2^k$ and generators $Q \in \mathbb{G}_2$, $P = \psi(Q) \in \mathbb{G}_1$, $g = e(P, Q) \in \mathbb{G}_T$. It then chooses a master key $s \xleftarrow{R} \mathbb{Z}_q^*$, a system-wide public key $Q_{pub} = sQ \in \mathbb{G}_2$ and hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$, $H_2 : \{0,1\}^* \times \mathbb{G}_T \to \mathbb{Z}_q^*$ and $H_3 : \mathbb{G}_T \to \{0,1\}^n$. The public parameters are $\mathsf{params} := \{q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P, Q, g, Q_{pub}, e, \psi, H_1, H_2, H_3\}$

Keygen: for an identity ID, the private key is $d_{\mathsf{ID}} = \frac{1}{H_1(\mathsf{ID})+s} Q \in \mathbb{G}_2$.

Sign/Encrypt: given a message $M \in \{0,1\}^*$, a receiver's identity ID_R and a sender's private key d_{ID_S},

(1) Pick $x \xleftarrow{R} \mathbb{Z}_q^*$, compute $r = g^x$ and $c = M \oplus H_3(r) \in \{0,1\}^n$.
(2) Set $h = H_2(M, r) \in \mathbb{Z}_q^*$.
(3) Compute $S = (x + h)\psi(d_{\mathsf{ID}_S})$.
(4) Compute $T = x(H_1(\mathsf{ID}_R)P + \psi(Q_{pub}))$.

The ciphertext is $\sigma = \langle c, S, T \rangle \in \{0,1\}^n \times \mathbb{G}_1 \times \mathbb{G}_1$.

Decrypt/Verify: given $\sigma = \langle c, S, T \rangle$, and some sender's identity ID_S,

(1) Compute $r = e(T, d_{\mathsf{ID}_R})$, $M = c \oplus H_3(r)$, and $h = H_2(M, r)$.
(2) Accept the message if $r = e(S, H_1(\mathsf{ID}_S)Q + Q_{pub})g^{-h}$. If this condition holds, return the message M together with the signature $(h, S) \in \mathbb{Z}_q^* \times \mathbb{G}_1$.

FIGURE 7.6. The FastIBSC scheme

sender's identity ID_S together with the message at step 1 of Sign/Encrypt in such a way that the recipient retrieves it at the first step of the reverse operation. This change does not imply any computational penalty in practice but induces more expensive security reductions. In order for the proof to hold, ID_S must be appended to the inputs of H_2.

6.2. Security results

The signature unforgeability property is proved under the p-Diffie-Hellman Inversion assumption introduced in chapter 6. The message confidentiality provably relies on the intractability of the p-Bilinear Diffie-Hellman Inversion problem introduced in chapter 4. In asymmetric bilinear map groups, the latter problem is defined as follows.

Definition 7.8 ([37, 35]) *Let us consider asymmetric bilinear map groups* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ *and generators* $P \in \mathbb{G}_1$ *and* $Q \in \mathbb{G}_2$. *The* **p-*Bilinear Diffie-Hellman Inversion*** *problem (p-BDHIP) in* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ *consists in, given* $\langle P, Q, \alpha Q, \alpha^2 Q, \ldots, \alpha^p Q\rangle$, *computing* $e(P,Q)^{1/\alpha} \in \mathbb{G}_T$.

The following theorems show the security of the scheme in the random oracle model under the same irreflexivity assumption as Boyen's scheme [45]: the signature/encryption algorithm is assumed to always take distinct identities as inputs (in other words, a principal never encrypts a message bearing his signature using his own identity).

Theorem 7.3 *Assume that an IND-IBSC-CCA adversary* $\mathcal{A}$ *has an advantage* ϵ *over our scheme when running in time* τ, *asking* q_{h_i} *queries to random oracles* H_i $(i = 1, 2, 3)$, q_{se} *signature/encryption queries and* q_{dv} *queries to the decryption/verification oracle. Then there is an algorithm* $\mathcal{B}$ *to solve the p-BDHIP for* $p = q_{h_1}$ *with probability*

$$\epsilon' > \frac{\epsilon}{q_{h_1}(2q_{h_2} + q_{h_3})} \left(1 - q_{se}\frac{q_{se} + q_{h_2}}{2^k}\right)\left(1 - \frac{q_{dv}}{2^k}\right)$$

within a time $\tau' < \tau + O(q_{se} + q_{dv})\tau_p + O(q_{h_1}^2)\tau_{mult} + O(q_{dv}q_{h_2})\tau_{exp}$ *where* τ_{exp} *and* τ_{mult} *are respectively the costs of an exponentiation in* $\mathbb{G}_T$ *and a multiplication in* $\mathbb{G}_2$ *whereas* τ_p *is the complexity of a pairing calculation.*

Proof. Algorithm $\mathcal{B}$ takes as input $\langle P, Q, \alpha Q, \alpha^2 Q, \ldots, \alpha^p Q\rangle$ and attempts to extract $e(P,Q)^{1/\alpha}$ from its interaction with $\mathcal{A}$.

In a preparation phase, $\mathcal{B}$ selects $\ell \stackrel{R}{\leftarrow} \{1, \ldots, q_{h_1}\}$, elements $I_\ell \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and $w_1, \ldots, w_{\ell-1}, w_{\ell+1} \ldots, w_{q_{h_1}} \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$. For $i = 1, \ldots, \ell-1, \ell+1, \ldots, q_{h_1}$, it computes $I_i = I_\ell - w_i$. As in the technique of [37] and in lemma 5.2, it sets up generators $G_2 \in \mathbb{G}_2$, $G_1 = \psi(G_2) \in \mathbb{G}_1$ and another $\mathbb{G}_2$ element $U = \alpha G_2$ such that it knows $q_{h_1} - 1$ pairs $(w_i, H_i = (1/(w_i + \alpha))G_2)$ for $i \in \{1, \ldots, q_{h_1}\}\backslash\{\ell\}$. The system-wide public key Q_{pub} is chosen as

$$Q_{pub} = -U - I_\ell G_2 = (-\alpha - I_\ell)G_2$$

so that its (unknown) private key is implicitly set to $x = -\alpha - I_\ell \in \mathbb{Z}_q^*$. For all $i \in \{1, \ldots, q_{h_1}\}\backslash\{\ell\}$, we have $(I_i, -H_i) = (I_i, (1/(I_i + x))G_2)$.

$\mathcal{B}$ then initializes a counter ν to 1 and starts the adversary $\mathcal{A}$ on input of (G_1, G_2, Q_{pub}). Throughout the game, we assume that H_1-queries are distinct, that the target identity ID_R^* is submitted to H_1 at some point

and that any query involving an identity comes after a H_1-query on that identity:

- H_1-queries (let us call ID_ν the input of the ν^{th} one of such queries): $\mathcal{B}$ answers I_ν and increments ν.
- H_2-queries on input (M, r): $\mathcal{B}$ returns the defined value if it exists and a random $h_2 \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ otherwise. To anticipate possible subsequent Decrypt/Verify requests, $\mathcal{B}$ additionally simulates random oracle H_3 on its own to obtain $h_3 = H_3(r) \in \{0,1\}^n$ and stores in list L_2 the information
$$(M, r, h_2, c = M \oplus h_3, \gamma = r \cdot e(G_1, G_2)^{h_2}).$$
- H_3-queries for an input $r \in \mathbb{G}_T$: $\mathcal{B}$ returns the previously assigned value if it exists and a random $h_3 \stackrel{R}{\leftarrow} \{0,1\}^n$ otherwise. In the latter case, the input r and the response h_3 are stored in a list L_3.
- Keygen queries on an input ID_ν: if $\nu = \ell$, then the simulator fails. Otherwise, it knows that $H_1(\mathsf{ID}_\nu) = I_\nu$ and returns $-H_\nu = (1/(I_\nu + x))\, G_2 \in \mathbb{G}_2$.
- Sign/Encrypt queries for a plaintext M and identities $(\mathsf{ID}_S, \mathsf{ID}_R) = (\mathsf{ID}_\mu, \mathsf{ID}_\nu)$ for $\mu, \nu \in \{1, \ldots, q_{h_1}\}$: we observe that, if $\mu \neq \ell$, $\mathcal{B}$ knows the sender's private key $d_{\mathsf{ID}_\mu} = -H_\mu$ and can answer the query by following the specification of Sign/Encrypt. We thus assume $\mu = \ell$ and hence $\nu \neq \ell$ by the irreflexivity assumption. Observe that $\mathcal{B}$ knows the receiver's private key $d_{\mathsf{ID}_\nu} = -H_\nu$ by construction. The difficulty is to find a random triple $(S, T, h) \in \mathbb{G}_1 \times \mathbb{G}_1 \times \mathbb{Z}_q^*$ for which

$$e(T, d_{\mathsf{ID}_\nu}) = e(S, Q_{\mathsf{ID}_\ell}) e(G_1, G_2)^{-h} \tag{3}$$

where $Q_{\mathsf{ID}_\ell} = I_\ell G_2 + Q_{pub}$. To do so, $\mathcal{B}$ randomly chooses elements $t, h \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and computes $S = t\psi(d_{\mathsf{ID}_\nu}) = -t\psi(H_\nu)$, $T = t\psi(Q_{\mathsf{ID}_\ell}) - h\psi(Q_{\mathsf{ID}_\nu})$ where $Q_{\mathsf{ID}_\nu} = I_\nu G_2 + Q_{pub}$ in order to obtain the desired equality

$$\begin{aligned} r = e(T, d_{\mathsf{ID}_\nu}) &= e(S, Q_{\mathsf{ID}_\ell}) e(G_1, G_2)^{-h} \\ &= e(\psi(d_{\mathsf{ID}_\nu}), Q_{\mathsf{ID}_\ell})^t e(G_1, G_2)^{-h} \end{aligned}$$

before patching the hash value $H_2(M, r)$ to h ($\mathcal{B}$ fails if H_2 is already defined but this happens with a probability smaller than $(q_{se} + q_{h_2})/2^k$). The ciphertext $\sigma = \langle M \oplus H_3(r), S, T\rangle$ is returned.

- Decrypt/Verify queries on a ciphertext $\sigma = \langle c, S, T\rangle$ for identities $(\mathsf{ID}_S, \mathsf{ID}_R) = (\mathsf{ID}_\mu, \mathsf{ID}_\nu)$: we assume that $\nu = \ell$ (and hence $\mu \neq \ell$ by the irreflexivity assumption), because otherwise $\mathcal{B}$ knows the receiver's private key $d_{\mathsf{ID}_\nu} = -H_\nu$ and can normally run the Decrypt/Verify algorithm. Since $\mu \neq \ell$, $\mathcal{B}$ has the sender's private key d_{ID_μ} and also knows that, for all valid ciphertexts, $\log_{d_{\mathsf{ID}_\mu}}(\psi^{-1}(S) - h d_{\mathsf{ID}_\mu}) = \log_{\psi(Q_{\mathsf{ID}_\nu})}(T)$, where $h = H_2(M, r)$ is the hash value obtained in the Sign/Encrypt algorithm and $Q_{\mathsf{ID}_\nu} = I_\nu G_2 + Q_{pub}$. Hence, we have the relation

$$e(T, d_{\mathsf{ID}_\mu}) = e(\psi(Q_{\mathsf{ID}_\nu}), \psi^{-1}(S) - h d_{\mathsf{ID}_\mu}) \tag{4}$$

which yields $e(T, d_{\mathsf{ID}_\mu}) = e(\psi(Q_{\mathsf{ID}_\nu}), \psi^{-1}(S)) e(\psi(Q_{\mathsf{ID}_\nu}), d_{\mathsf{ID}_\mu})^{-h}$. We observe that the latter equality can be tested without inverting ψ as $e(\psi(Q_{\mathsf{ID}_\nu}), \psi^{-1}(S)) = e(S, Q_{\mathsf{ID}_\nu})$. The query is thus handled by computing $\gamma = e(S, Q_{\mathsf{ID}_\mu})$, where $Q_{\mathsf{ID}_\mu} = I_\mu G_2 + Q_{pub}$, and searching through list L_2 for records of the form $(M_i, r_i, h_{2,i}, c, \gamma)$ indexed by $i \in \{1, \ldots, q_{h_2}\}$. If none is found, σ is rejected. Otherwise, each one of them is further examined: for the corresponding indexes, $\mathcal{B}$ checks if

$$e(T, d_{\mathsf{ID}_\mu})/e(S, Q_{\mathsf{ID}_\nu}) = e(\psi(Q_{\mathsf{ID}_\nu}), d_{\mathsf{ID}_\mu})^{-h_{2,i}} \tag{5}$$

(the pairings are computed only once and at most q_{h_2} exponentiations are needed), meaning that (4) is satisfied. If the unique $i \in \{1, \ldots, q_{h_2}\}$ satisfying (5) is detected, the matching pair $(M_i, \langle h_{2,i}, S\rangle)$ is returned. Otherwise, σ is rejected. Overall, an inappropriate rejection occurs with probability smaller than $q_{dv}/2^k$ across the whole game.

At the challenge phase, $\mathcal{A}$ outputs messages (M_0, M_1) and identities $(\mathsf{ID}_S, \mathsf{ID}_R)$ for which she never obtained ID_R's private key. If $\mathsf{ID}_R \neq \mathsf{ID}_\ell$, $\mathcal{B}$ aborts. Otherwise, it picks $\xi \xleftarrow{R} \mathbb{Z}_q^*$, $c \xleftarrow{R} \{0,1\}^n$ and $S \xleftarrow{R} \mathbb{G}_1$ to return the challenge $\sigma^* = \langle c, S, T\rangle$ where $T = -\xi G_1 \in \mathbb{G}_1$. If we define $\rho = \xi/\alpha$ and since $x = -\alpha - I_\ell$, we can check that

$$T = -\xi G_1 = -\alpha\rho G_1 = (I_\ell + x)\rho G_1 = \rho I_\ell G_1 + \rho\psi(Q_{pub}).$$

$\mathcal{A}$ cannot recognize that σ^* is not a proper ciphertext unless she queries H_2 or H_3 on $e(G_1, G_2)^\rho$. Along the guess stage, her view is simulated as before and her eventual output is ignored. Standard arguments can show that a successful $\mathcal{A}$ is very likely to query H_2 or H_3 on the input $e(G_1, G_2)^\rho$ if the simulation is indistinguishable from a real attack environment.

To produce a result, $\mathcal{B}$ fetches a random record (M, r, h_2, c, γ) or $(r, .)$ from the lists L_2 or L_3. With probability $1/(2q_{h_2} + q_{h_3})$ (as L_3 contains no more than $q_{h_2} + q_{h_3}$ records by construction), the chosen record will contain the right element $r = e(G_1, G_2)^\rho = e(P, Q)^{f(\alpha)^2\xi/\alpha}$, where $f(z) = \sum_{i=0}^{p-1} c_i z^i$ is the polynomial for which $G_2 = f(\alpha)Q$. The p-BDHIP solution can be extracted by noting that, if $\gamma^* = e(P, Q)^{1/\alpha}$, then

$$e(G_1, G_2)^{1/\alpha} = \gamma^{*(c_0^2)} e\big(\sum_{i=0}^{p-2} c_{i+1}(\alpha^i P), c_0 Q\big) e\big(G_1, \sum_{j=0}^{p-2} c_{j+1}(\alpha^j)Q\big).$$

In an analysis of $\mathcal{B}$'s advantage, we note that it only fails in providing a consistent simulation because one of the following independent events:

- E_1: $\mathcal{A}$ does not choose to be challenged on ID_ℓ.
- E_2: a key extraction query is made on ID_ℓ.
- E_3: $\mathcal{B}$ aborts in a Sign/Encrypt query because of a collision on H_2.
- E_4: $\mathcal{B}$ rejects a valid ciphertext at some point of the game.

We clearly have $\Pr[\neg E_1] = 1/q_{h_1}$ and we know that $\neg E_1$ implies $\neg E_2$. We also already observed that $\Pr[E_3] \leq q_{se}(q_{se} + q_{h_2})/2^k$ and $\Pr[E_4] \leq q_{dv}/2^k$. We thus find that

$$\Pr[\neg E_1 \wedge \neg E_3 \wedge \neg E_4] \geq \frac{1}{q_{h_1}} \left(1 - q_{se}\frac{q_{se} + q_{h_2}}{2^k}\right)\left(1 - \frac{q_{dv}}{2^k}\right).$$

We obtain the announced bound by noting that $\mathcal{B}$ selects the correct element from L_2 or L_3 with probability $1/(2q_{h_2}+q_{h_3})$. Its workload is dominated by $O(q_{h_1}^2)$ multiplications in the preparation phase, $O(q_{se} + q_{dv})$ pairing calculations and $O(q_{dv}q_{h_2})$ exponentiations in $\mathbb{G}_T$ in its emulation of the Sign/Encrypt and Decrypt/Verify oracles. □

Theorem 7.4 *Assume there exists an ESUF-IBSC-CMA attacker $\mathcal{A}$ that makes q_{h_i} queries to random oracles H_i $(i = 1, 2, 3)$, q_{se} signature/encryption queries and q_{dv} queries to the decryption/verification oracle. Assume also that, within a time τ, $\mathcal{A}$ produces a forgery with probability*

$\epsilon \geq 10(q_{se}+1)(q_{se}+q_{h_2})/2^k$. *Then, there is an algorithm* $\mathcal{B}$ *that is able to solve the p-DHIP for* $p = q_{h_1}$ *in expected time*

$$\tau' \leq 120686 q_{h_1} q_{h_2} \frac{\tau + O((q_{se}+q_{dv})\tau_p) + q_{dv} q_{h_2} \tau_{exp}}{\epsilon(1-1/2^k)(1-q/2^k)} + O(q^2 \tau_{mult})$$

where τ_{mult}, τ_{exp} *and* τ_p *denote the same quantities as in theorem 7.3.*

PROOF. The proof is almost the same as that of theorem 5.1. Namely, it shows that a forger in the ESUF-IBSC-CMA game implies a forger in a chosen-message and *given* identity attack. Using the forking lemma [182, 183], the latter is in turn shown to imply an algorithm to solve the p-Diffie-Hellman Inversion problem. More precisely, queries to the Sign/Encrypt and Decrypt/Verify oracles are answered as in the proof of theorem 7.3 and, at the outset of the game, the simulator chooses public parameters in such a way that it can extract private keys associated to any identity but the one which is given as a challenge to the adversary. By doing so, thanks to the irreflexivity assumption, it is able to extract clear message-signature pairs from ciphertexts produced by the forger (as it knows the private key of the receiving identity ID_R^*).

□

We now restate theorem 7.3 for the variant of our scheme with anonymous ciphertexts. The simulator's worst-case running time is affected by the fact that, when handling Decrypt/Verify requests, senders' identities are not known in advance. The reduction involves a number of pairing calculations which is quadratic in the number of adversarial queries.

Theorem 7.5 *Assume that an IND-IBSC-CCA adversary* $\mathcal{A}$ *has an advantage* ϵ *over our scheme with anonymous ciphertexts when running in time* τ, *asking* q_{h_i} *queries to random oracles* H_i $(i = 1, 2, 3)$, q_{se} *signature/encryption queries and* q_{dv} *queries to the decryption/verification oracle. Then there is an algorithm* $\mathcal{B}$ *to solve the p-BDHIP for* $p = q_{h_1}$ *with probability*

$$\epsilon' > \frac{\epsilon}{q_{h_1}(2q_{h_2}+q_{h_3})} \left(1 - q_{se}\frac{q_{se}+q_{h_2}}{2^k}\right)\left(1 - \frac{q_{dv}}{2^k}\right)$$

within a time $\tau' < \tau + O(q_{se}+q_{dv}q_{h_2})\tau_p + O(q_{h_1}^2)\tau_{mult} + O(q_{dv}q_{h_2})\tau_{exp}$ *where* τ_{exp}, τ_{mult} *and* τ_p *denote the same quantities as in theorem 7.3.*

PROOF. The simulator is the same as in theorem 7.3 with the following differences (recall that senders' identities are provided as inputs to H_2).

- H_2-queries on input (ID_S, M, r): $\mathcal{B}$ returns the previously defined value if it exists and a random $h_2 \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ otherwise. To anticipate subsequent Decrypt/Verify requests, $\mathcal{B}$ simulates oracle H_3 to obtain $h_3 = H_3(r) \in \{0,1\}^{n+n_0}$ (where n_0 is the maximum length of identity strings) and stores $(\mathsf{ID}_S, M, r, h_2, c = (M \| \mathsf{ID}_S) \oplus h_3, \gamma = r \cdot e(G_1, G_2)^{h_2})$ in list L_2.
- Decrypt/Verify queries: given a ciphertext $\sigma = \langle c, S, T \rangle$ and a receiver's identity $\mathsf{ID}_R = \mathsf{ID}_\nu$, we assume that $\nu = \ell$ because otherwise $\mathcal{B}$ knows the receiver's private key. The simulator $\mathcal{B}$ does not know the sender's identity ID_S but knows that $\mathsf{ID}_S \neq \mathsf{ID}_\nu$. It also knows that, for the private key d_{ID_S}, $\log_{d_{\mathsf{ID}_S}}(\psi^{-1}(S) - h d_{\mathsf{ID}_S}) = \log_{\psi(Q_{\mathsf{ID}_\nu})}(T)$, and hence

$$e(T, d_{\mathsf{ID}_S}) = e(\psi(Q_{\mathsf{ID}_\nu}), \psi^{-1}(S) - h d_{\mathsf{ID}_S}), \tag{6}$$

where $h = H_2(\mathsf{ID}_S, M, r)$ is the hash value obtained in the Sign/ Encrypt algorithm and $Q_{\mathsf{ID}_\nu} = I_\nu G_2 + Q_{pub}$. The query is handled by searching through list L_2 for records of the form $(\mathsf{ID}_{S,i}, M_i, r_i, h_{2,i}, c, \gamma_i)$ indexed by $i \in \{1, \ldots, q_{h_2}\}$. If none is found, the ciphertext is rejected. Otherwise, each one of these tuples for which $\mathsf{ID}_{S,i} \neq \mathsf{ID}_\nu$ is further examined by checking whether $\gamma_i = e(S, H_1(\mathsf{ID}_{S,i})Q + Q_{pub})$ and

$$e(T, d_{\mathsf{ID}_{S,i}}) / e(S, Q_{\mathsf{ID}_\nu}) = e(\psi(Q_{\mathsf{ID}_\nu}), d_{\mathsf{ID}_{S,i}})^{-h_{2,i}} \tag{7}$$

(at most $3q_{h_2} + 1$ pairings and q_{h_2} exponentiations must be computed), meaning that equation (6) is satisfied and that the ciphertext contains a valid message signature pair if both relations hold. If $\mathcal{B}$ detects an index $i \in \{1, \ldots, q_{h_2}\}$ satisfying them, the matching triple $(M_i, \langle h_{2,i}, S \rangle, \mathsf{ID}_{S,i})$ is returned. Otherwise, σ is rejected and such a wrong rejection again occurs with an overall probability smaller than $q_{dv}/2^k$.

□

Theorem 7.4 can be similarly restated as its reduction cost is affected in the same way.

The ciphertext anonymity property of the anonymized version of FastIBSC remains to be formally established by the following theorem.

Theorem 7.6 *Assume that an ANON-IBSC-CCA adversary $\mathcal{A}$ has an advantage ϵ over the anonymous variant of our scheme when running in time τ, asking q_{h_i} queries to random oracles H_i $(i = 1, 2, 3)$, q_{se} signature/encryption queries and q_{dv} queries to the decryption/verification oracle. Then there is an algorithm $\mathcal{B}$ to solve the p-BDHIP for $p = q_{h_1}$ with probability*

$$\epsilon' > \frac{4\epsilon}{3q_{h_1}(2q_{h_2} + q_{h_3})} \left(1 - q_{se}\frac{q_{se} + q_{h_2}}{2^k}\right)\left(1 - \frac{q_{dv}}{2^k}\right)$$

within a time $\tau' < \tau + O(q_{se} + q_{dv}q_{h_2})\tau_p + O(q_{h_1}^2)\tau_{mult} + O(q_{dv}q_{h_2})\tau_{exp}$ where τ_{exp}, τ_{mult} and τ_p denote the same quantities as in theorems 7.5.

Proof. Algorithm $\mathcal{B}$ takes as input $\langle P, Q, \alpha Q, \alpha^2 Q, \ldots, \alpha^p Q\rangle$ and attempts to find $e(P, Q)^{1/\alpha}$ using $\mathcal{A}$.

In a preparation phase, $\mathcal{B}$ selects $\ell \xleftarrow{R} \{1, \ldots, q_{h_1}\}$, elements $I_\ell \xleftarrow{R} \mathbb{Z}_q^*$ and $w_1, \ldots, w_{\ell-1}, w_{\ell+1} \ldots, w_{q_{h_1}} \xleftarrow{R} \mathbb{Z}_q^*$. For $i = 1, \ldots, \ell-1, \ell+1, \ldots, q_{h_1}$, it computes $I_i = I_\ell - w_i$. As in the proof of theorem 7.3, it chooses generators $G_2 \in \mathbb{G}_2$, $G_1 = \psi(G_2) \in \mathbb{G}_1$ and another $\mathbb{G}_2$ element $U = \alpha G_2$ such that it knows $q_{h_1} - 1$ pairs $(w_i, H_i = (1/(w_i + \alpha))G_2)$ for $i \in \{1, \ldots, q_{h_1}\}\backslash\{\ell\}$. The public key Q_{pub} is chosen as

$$Q_{pub} = -U - I_\ell G_2 = (-\alpha - I_\ell)G_2$$

so that its private key is implicitly set to $x = -\alpha - I_\ell \in \mathbb{Z}_q^*$.

$\mathcal{B}$ initializes a counter ν to 1 and starts $\mathcal{A}$ on input of (G_1, G_2, Q_{pub}). We again assume that H_1-queries are distinct, that the target identities $\mathsf{ID}_{R,0}^*, \mathsf{ID}_{R,1}^*$ are submitted to H_1 at some point and that any query involving an identity comes after a H_1-query on it. All oracles are simulated as in the proof of theorem 7.5.

At the challenge phase, $\mathcal{A}$ outputs two pairs of identities $(\mathsf{ID}_{S,0}^*, \mathsf{ID}_{S,1}^*)$ and $(\mathsf{ID}_{R,0}^*, \mathsf{ID}_{R,1}^*)$ among which $\mathsf{ID}_{R,0}^*$ and $\mathsf{ID}_{R,1}^*$ were never submitted to the key extraction oracle. If $\mathsf{ID}_{R,0}^*, \mathsf{ID}_{R,1}^* \neq \mathsf{ID}_\ell$, $\mathcal{B}$ aborts. Otherwise, we may assume w.l.o.g. that $\mathsf{ID}_{R,0}^* = \mathsf{ID}_\ell$ (the case $\mathsf{ID}_{R,1}^* = W_\ell$ is treated in the same way). It picks $\xi \xleftarrow{R} \mathbb{Z}_q^*$, $S \xleftarrow{R} \mathbb{G}_1$ and $c \xleftarrow{R} \{0, 1\}^n$ to return the challenge $\sigma^* = \langle c, S, T\rangle$ where $T = -\xi G_1 \in \mathbb{G}_1$. If we define $\rho = \xi/\alpha$ and since $x = -\alpha - I_\ell$, we can check that

$$T = -\xi G_1 = -\alpha\rho G_1 = (I_\ell + x)\rho G_1 = \rho I_\ell G_1 + \rho\psi(Q_{pub}).$$

$\mathcal{A}$ cannot recognize that σ^* is not a proper ciphertext unless she queries H_2 on either

$$\begin{aligned} r_0^* &= e\big(T, \frac{1}{x + H_1(\mathsf{ID}_{R,0}^*)} G_2\big) = e(G_1, G_2)^\rho \\ \text{or } r_1^* &= e\big(T, \frac{1}{x + H_1(\mathsf{ID}_{R,1}^*)} G_2\big). \end{aligned}$$

Along the second stage, her view is simulated as before and her eventual output is ignored. Standard arguments can show that a successful $\mathcal{A}$ is very likely to query H_2 on either r_0^* or r_1^* if the simulation is indistinguishable from a real attack environment. Let AskH_2 denote this event. In a real attack, we have

$$\Pr[\mathcal{A} \text{ wins}] \leq \Pr[\mathcal{A} \text{ wins}|\neg\mathsf{AskH}_2]\Pr[\neg\mathsf{AskH}_2] + \Pr[\mathsf{AskH}_2].$$

Clearly, $\Pr[\mathcal{A} \text{ wins}|\neg\mathsf{AskH}_2] = 1/4$ and $\Pr[\mathcal{A} \text{ wins}] \leq 1/4 + (3/4)\Pr[\mathsf{AskH}_2]$. On the other hand, we have $\Pr[\mathcal{A} \text{ wins}] = \epsilon + 1/4$. It comes that $\Pr[\mathsf{AskH}_2] \geq 4\epsilon/3$. Hence, provided the simulation is consistent, $\mathcal{A}$ issues a H_2-query on either r_0^* or r_1^* at some point of the game with probability at least $4\epsilon/3$. With probability $2\epsilon/3$, a H_2-query involving $r_0^* = e(G_1, G_2)^\rho$ will thus be issued.

To produce a result, $\mathcal{B}$ picks a random record from L_2 or L_3. With probability $1/q_{h_2}$, the chosen record contains the right element $r_0^* = e(G_1, G_2)^\rho = e(P, Q)^{f(\alpha)^2 \xi/\alpha}$, where $f(z) = \sum_{i=0}^{p-1} c_i z^i$ is the polynomial for which $G_2 = f(\alpha)Q$. The p-BDHIP solution can be extracted by noting that, if $\gamma^* = e(P, Q)^{1/\alpha}$, then

$$e(G_1, G_2)^{1/\alpha} = \gamma^{*(c_0^2)} e\big(\sum_{i=0}^{p-2} c_{i+1}(\alpha^i P), c_0 Q\big) e\big(G_1, \sum_{j=0}^{p-2} c_{j+1}(\alpha^j)Q\big).$$

In an analysis of $\mathcal{B}$'s advantage, we note that it only fails in providing a consistent simulation because one of the following independent events:

- E_1: $\mathsf{ID}_{R,0}^*, \mathsf{ID}_{R,1}^* \neq \mathsf{ID}_\ell$.
- E_2: a key extraction query is made on ID_ℓ.
- E_3: $\mathcal{B}$ aborts in a $\mathsf{Sign/Encrypt}$ query because of a collision on H_2.
- E_4: $\mathcal{B}$ rejects a valid ciphertext at some point of the game.

We clearly have $\Pr[\neg E_1] = (q_{h_1} - 1)/\binom{q_{h_1}}{2} = 2/q_{h_1}$ and we know that $\neg E_1$ implies $\neg E_2$. As in the proof of theorem 7.3, we have $\Pr[E_3] \leq q_{se}(q_{se} + q_{h_2})/2^k$ and $\Pr[E_4] \leq q_{dv}/2^k$. We thus find that

$$\Pr[\neg E_1 \wedge \neg E_3 \wedge \neg E_4] \geq \frac{2}{q_{h_1}} \left(1 - q_{se}\frac{q_{se}+q_{h_2}}{2^k}\right)\left(1 - \frac{q_{dv}}{2^k}\right).$$

□

6.3. Efficiency discussions and comparisons

In [203], Smart and Vercauteren pointed out problems that arise when several pairing-based protocols are implemented with asymmetric pairings. They showed the difficulty of finding groups $\mathbb{G}_2$ allowing the use of the most efficient pairing calculation techniques for ordinary curves [22] if arbitrary strings should be *efficiently* hashed onto them and an efficient isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ must be available at the same time. As a consequence, several protocols have to be implemented with groups for which no efficient isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ is computable and their security eventually has to rely on somewhat unnatural assumptions involving an oracle that computes groups isomorphisms between $\mathbb{G}_2$ and $\mathbb{G}_1$.

Except [189] that has no security proof (and actually has several known security problems [144]), all known identity-based signcryption schemes would require to hash onto $\mathbb{G}_2$ if they were instantiated with asymmetric pairings. Our scheme avoids this problem since it does not require to hash onto a cyclic group. It thus more easily benefits from optimized pairing calculation algorithms. For example, section 4 of [203] yields an example of group $\mathbb{G}_2$ for which techniques of [22] can be used and where efficient isomorphisms are available.

We now assess the comparative efficiency of several identity-based signcryption schemes, implemented in accordance with their original descriptions. Table 5.1 summarises the number of relevant basic operations underlying several identity-based signcryption and signature schemes, namely, $\mathbb{G}_T$ exponentiations, scalar point multiplications, and pairing evaluations, and compares the processing times (in milliseconds) observed by Paulo Barreto for a supersingular curve of embedding degree $k = 6$ over $\mathbb{F}_{3^{97}}$, using implementations written in C++ and run on an Athlon XP 2 GHz. Subtleties in the algorithms determine somewhat different running times even when the operation counts for those algorithms are equal. We see from these results that our scheme even beats the fastest other ones. The Sign/Encrypt algorithm is more than twice as fast as in the Chen-Malone-Lee scheme.

	Sign/Encrypt				Decrypt/Verify			
IBSC schemes	exp	mul	pairings	time (ms)	exp	mul	pairings	time (ms)
Boyen [45]	1	3	$1^{\dagger}$	11.09		2	$4^{\dagger}$	18.44
Chow et al.¶ [64]		2	$2^{\star}$	10.24		1	$4^{\star}$	17.65
NewIBSC$^{\P\spadesuit}$ [131]		2	$2^{\star}$	10.24		1	$4^{\star}$	17.65
Nalla-Reddy$^{\diamondsuit\bowtie}$ [161]	1	2	$1^{\dagger}$	10.06	1		$3^{\dagger}$	13.44
Malone-Lee$^{\clubsuit}$ [139]		3	$1^{\ddagger}$	7.03		1	3	13.44
Chen-Malone-Lee [58]		3	$1^{\ddagger}$	7.03		1	3	13.44
Sakai-Kasahara$^{\clubsuit}$ [189]	2	$1+1^{\S}$		6.56	1		2	12.35
FastIBSC	1	2		2.65	1		2	9.06

(†) One pairing is precomputable, incurring for each user a storage cost of one $\mathbb{G}_T$ element for each other user in the system.
(‡) One pairing is precomputable, incurring for each user a storage cost of one $\mathbb{G}_T$ element for each other user in the system, plus one $\mathbb{G}_T$ exponentiation.
($\star$) Two pairings are precomputable, incurring for each user a storage cost of one $\mathbb{G}_T$ element for each user in the system, plus two $\mathbb{G}_T$ exponentiations.
(§) One of the scalar multiplications is done in $\langle Q \rangle$ rather than $\langle P \rangle$.
(¶) Universally verifiable scheme (i.e. supports public ciphertext validation).
($\clubsuit$) These schemes suffer from security problems as mentioned in [144] and in this chapter.
($\spadesuit$) This scheme does not provide insider-security for the message-confidentiality criterion.
($\diamondsuit$) This scheme has no security proof.
($\bowtie$) This construction can only authenticate messages from the receiver's point of view.

TABLE 7.1. Efficiency comparisons

7. Conclusions

This chapter analyzed several identity-based protocols jointly performing signature and encryption. We pinpointed a flaw in the original scheme proposed by Malone-Lee and we showed a (not entirely satisfying) method to repair it.

We also proposed a new scheme that employs the key generation technique of Sakai and Kasahara [189] and which is constructed on our new efficient identity-based signature described in chapter 5. The new scheme is showed to outperform all previously known systems providing combined encryption and authentication in identity-based cryptography.

Part 4

Other Contributions

CHAPTER 8

Joint Signature and Encryption with Traditional Public Keys

Abstract. We propose here three new constructions of authenticated public key encryption schemes based on discrete logarithm related assumptions. Each of those constructions has its own advantages and disadvantages. Two of them are extensions of pairing-based digital signature schemes. The third one can be implemented with general groups and is constructed on the Schnorr signature. We consider for them stringent security models and give security proofs in the random oracle model under plausible (although sometimes very recently suggested) computational assumptions.

1. Combined public key encryption and authentication

As discussed in the previous chapter, many cryptographic applications, such as secure e-mail or secure channel establishment protocols, need the requirements of confidentiality and authentication to be simultaneously fulfilled. To achieve them in the asymmetric setting, the concept of public key authenticated encryption, or 'signcryption', was introduced by Zheng in 1997 [228]. This kind of primitive aims at efficiently performing encryption and signature in a single logical step in order to obtain confidentiality, integrity, authentication and non-repudiation. We recall that the basic encrypt-then-sign composition is generally insecure (except for some particular constructions such as [8] or [117] or if special precautions are taken as in [9]) against chosen-ciphertext attacks as well as the encrypt-and-sign approach. The drawback of most of the latter compositions is to expand the final ciphertext's size and increase the sender and receiver's computing time.

Five years after Zheng's introducing paper, his original discrete logarithm related signcryption proposal [228] was proved secure [15] by

Baek et al. who described a formal security model in a multi-user setting. In [204], Steinfeld and Zheng proposed another scheme for which the unforgeability of ciphertexts relies on the intractability of the integer factoring problem but they provided no proof of chosen-ciphertext security.

The previously cited schemes have the shortcoming not to offer easy non-repudiation of ciphertexts: a recipient cannot convince a third party that a plaintext actually emanates from its sender. A method was proposed in [18] to overcome this limitation but it was shown [196] to leak information on the plaintext as another scheme described in [221]. This weakness can easily be fixed by slightly modifying the schemes as suggested by Malone-Lee in [138]. Unfortunately, even such a modification is not sufficient to render the schemes robust against insider attacks discussed by An, Dodis and Rabin [9] for the confidentiality criterion: a chosen-ciphertext attacker can learn some user's private key and break the confidentiality of messages previously signcrypted by that user.

Actually, formal security notions against outside attackers were first considered by An in [8] where general composition methods for asymmetric encryption and digital signatures are analyzed. The first security models capturing the scenario of a network of multiple users where attackers may be members of the network were introduced in [9] where the security of generic compositions of signature and encryption schemes was analyzed. In the same paper, they addressed the approach consisting in performing signature and encryption in parallel: a plaintext is first transformed into a commitment/de-commitment pair (c, d) in such a way that c reveals no information about m while the pair (c, d) allows recovering m. The signer can then jointly sign c and encrypt d in parallel using appropriate encryption and signature schemes. The reverse operation is also achieved by the recipient in a parallel fashion: the signature on c is verified while the decryption reveals d and the pair (c, d) is finally used to recover the plaintext.

After the work of An et al. [9], several researchers attempted to devise schemes providing a better efficiency than sequential compositions. A first secure discrete logarithm related signcryption scheme in an appropriate model of chosen-ciphertext security was described in [196] but no proof of unforgeability was given for it. An RSA-based scheme was described by Malone-Lee and Mao [140] who provided proofs for

unforgeability under chosen-message attacks and chosen-ciphertext security but they only considered the security in a two-user setting rather than the more realistic multi-user setting. Furthermore, the security of that scheme is only loosely related to the RSA assumption. However, none of the aforementioned schemes is provably secure against insider attacks: in some of them, an attacker learning some user's private key can even recover all the messages previously sent by that user.

The parallel signcryption approach of An et al. [9] was further investigated by Pieprzyk and Pointcheval [179] who proposed to use a commitment scheme based on a $(2, 2)$-Shamir secret sharing of an appropriately salted plaintext: the first resulting share s_1, which does not individually reveal any information on m, is used as a commitment and is signed while the second share s_2 is encrypted as a de-commitment. That method also provides a construction allowing to integrate any one-way encryption system (such as the basic RSA) with a weakly secure signature (a non-universally forgeable signature) into a CCA2-secure and existentially unforgeable scheme.

Dodis et al. [79] recently proposed other possibly parallel signcryption techniques, one of which which can be viewed as a generalization of existing probabilistic paddings such as OAEP, OAEP+ or PSS-R. They showed that their constructions allow optimal exact security as well as compatibility with PKCS standards and have other interesting properties. In fact, the latter schemes are probably the most practical ones among all solutions based on trapdoor functions.

This chapter summarizes three constructions that we proposed and for which we gave security proofs under discrete logarithm related assumptions. Two of these constructions are built on pairing-based signature schemes and can be instantiated with either symmetric or asymmetric pairings. The first one, published at PKC'04 [132], enjoys tight security reductions from a fairly standard variant of the Diffie-Hellman problem in groups equipped with bilinear mappings. It is proved to satisfy strong security notions (namely, notions of chosen-ciphertext security and 'ciphertext strong unforgeability' even against insider attacks) and provide anonymous ciphertexts (i.e. a ciphertext contains no information identifying its originator nor its recipient). Unfortunately, this Diffie-Hellman based scheme, that is obtained from Boneh et al.'s short signature [43] in a randomized version, offers very few computational

savings w.r.t. sequential compositions: it only spares one elliptic curve scalar multiplication on the receiver's side as well as a 160-bit bandwidth overhead in the ciphertext size when compared to a sequential composition of the BLS signature with an ElGamal encryption scheme padded with either of the Fujisaki-Okamoto [89, 88] conversions.

We thus subsequently proposed [135] a more efficient Diffie-Hellman based signcryption solution that satisfies the same strong security requirements: this second scheme has essentially the same cost as a mere ElGamal encryption on the sender's side while only one pairing evaluation and three exponentiations are required for the simultaneous decryption/verification tasks. This improves the efficiency of the signature/encryption operation of our first scheme by about 33% while the decryption/verification algorithm is almost twice as fast. The price to pay for such improvements is a security that relies on stronger assumptions than our original Diffie-Hellman based scheme: the chosen-ciphertext security is proved under the p-Diffie-Hellman Inversion assumption already considered in [37], [35] and [227] while the unforgeability property relies on the p-Strong Diffie-Hellman assumption introduced by Boneh and Boyen [37]. This second construction additionally features ciphertexts from which short signatures can be extracted if the system is instantiated using suitable ordinary curves.

We finally describe a third construction which is not based on pairings but rather on the Schnorr signature [191]. Its advantage is to be compatible with much more general groups than those equipped with pairings. It is also much more efficient for receivers who can extract standard Schnorr signatures from ciphertexts. The disadvantage of the latter construction is an inefficient reduction under a rather non-standard assumption in the proof of unforgeability.

Before describing our schemes, this chapter will first explain the motivations for the design of signcryption schemes and the study of signcryption as a cryptographic primitive. It will then explain the formal syntax of our schemes and the security models that will be used to analyze their security. The description of our schemes will follow in sections 4, 5 and 6 where security proofs in the random oracle model will also be given.

2. Motivations for the design of signcryption schemes

The main motivation of Zheng to study signcryption as a separate primitive was a gain in efficiency. It was indeed observed in [228] that confidentiality and authentication might be achieved significantly more cheaply than by combining encryption and signature tools. But efficiency improvements are far from being the only reason to study this kind of public key primitive. Several recent papers such as [9] or [196] showed that, even given digital signatures and public key encryption schemes that are each secure in their model, it was not trivial to combine them into a secure cryptographic scheme providing both authentication and privacy. Indeed, the trivial solution of appending a signature on the plaintext to a ciphertext (which was sometimes called "encrypt-and-sign" in the literature) is insecure in an indistinguishability scenario: given a challenge ciphertext

$$C = \langle \mathsf{Sig}_{\mathsf{sk}_S}(m), \mathsf{Enc}_{\mathsf{pk}_R}(m) \rangle$$

where sk_S and pk_R respectively denote the sender's private key and the receiver's public key and $\mathsf{Sig}_{\mathsf{sk}}(.)$ and $\mathsf{Enc}_{\mathsf{pk}}(.)$ are respectively signature and encryption algorithms, an adversary can easily decide whether C is a signature/encryption of two messages m_0 or m_1 by simply verifying the signature on both m_0 or m_1.

Another approach is called "encrypt-then-sign" and consists of appending a signature on the ciphertext so that messages sequentially encrypted and signed have the form

$$C = \langle \mathsf{Sig}_{\mathsf{sk}_S}(\mathsf{Enc}_{\mathsf{pk}_R}(m)), \mathsf{Enc}_{\mathsf{pk}_R}(m), \mathsf{ID}_S, \mathsf{ID}_R \rangle$$

where ID_S and ID_R are identities of the sender and receiver. The latter approach suffers from a kind of "identity fraud attack" which prevents it from being secure in a chosen-ciphertext scenario in the multi-user setting unless special precautions are taken (such as encrypting the sender's identity along with the plaintext and signing the receiver's identity together with the second part of the ciphertext) as explained in [9]. Indeed, anyone intercepting the ciphertext can simply replace the signature $\mathsf{Sig}_{\mathsf{sk}_S}(\mathsf{Enc}_{\mathsf{pk}_R}(m))$ with his own signature $\mathsf{Sig}_{\mathsf{sk}'_S}(\mathsf{Enc}_{\mathsf{pk}_R}(m))$ and obtain another signcryption

$$C = \langle \mathsf{Sig}_{\mathsf{sk}'_S}(\mathsf{Enc}_{\mathsf{pk}_R}(m)), \mathsf{Enc}_{\mathsf{pk}_R}(m), \mathsf{ID}'_S, \mathsf{ID}_R \rangle$$

of the same message and thus break the non-malleability of the scheme in some sense. It is unclear whether the latter form of malleability is actually damaging in practice, but this composition suffers from an additional disadvantage: it might complicate the task of a receiver wishing to convince a third party that a ciphertext actually emanates from the receiver.

The reversed-order sequential composition called "Sign-then-encrypt" does not suffer from the latter non-repudiation issue. It ends up with signed and encrypted messages of the form

$$C = \langle \mathsf{Enc}_{\mathsf{pk}_R}(m, \mathsf{Sig}_{\mathsf{sk}_S}(m, \mathsf{ID}_R), \mathsf{ID}_S), \mathsf{ID}_S, \mathsf{ID}_R \rangle.$$

It was shown in [9] how to ensure privacy and authentication against insider-attacks (that is adversaries have access to the receiver's private key in the game modelling the unforgeability property and to the sender's private key in the game modelling privacy) in a suitable model. Including both parties' identities at crucial points of the process is important to prevent identity fraud attacks as discussed by An, Dodis and Rabin [9].

The same authors also introduced a third approach named "Commit-then-encrypt-and-sign" which is to first split a message into a commitment and a matching de-commitment in such a way that the former can be signed while the latter is encrypted in parallel. Ciphertexts have the shape

$$C = \langle \mathsf{Sig}_{\mathsf{sk}_S}(c, \mathsf{ID}_R), \mathsf{Enc}_{\mathsf{pk}_R}(d, \mathsf{ID}_S), \mathsf{ID}_S, \mathsf{ID}_R \rangle$$

where c is a commitment to the plaintext and d is the corresponding de-commitment enabling m to be retrieved from the pair (c, d). This method was shown to satisfy the security model of [9] provided the employed commitment scheme satisfies some reasonable security requirement. The "Commit-then-encrypt-and-sign" approach has the advantage to be compatible with a parallel implementation of the signature and encryption operations. The length of the whole signcryption operation may thus be reduced to the maximal time-complexity of the encryption and signature subroutines. Following [9], the idea of 'parallel signcryption' was further investigated by Pierpzyk and Pointcheval [179] and more recently by Dodis et al. [79] who devised efficient paddings for joint signature and encryption using trapdoor permutations and a special kind of commitment schemes.

The paper by An, Dodis and Rabin [9] was actually the first one to address the security of modular approaches to construct signcryption protocols from secure underlying encryption and signature schemes. Independently of [15], they were the first ones to consider formal security models for signcryption schemes in the multi-user setting. In the present chapter, we slightly depart from those security models because we also focus on ensuring the anonymity of ciphertexts: as mentioned in the introduction of this chapter, we want to avoid transmitting the identities of corresponding parties. We shall thus employ a security model that is tailored to an anonymous setting.

3. Our model of signcryption schemes

The present section first explains the interface of the various algorithms upon which our signcryption schemes are made. It then considers formal models of security which are slightly different from other ones given in [15], [9] or [138].

3.1. Formal components

Definition 8.1 *A signcryption scheme is made of four algorithms which are the following.*

Common-Keygen: *is a probabilistic algorithm that takes as input a security parameter to output public information I that will be used by all users of the scheme. Such a public information may include the description of plaintext and ciphertext spaces, public and private key spaces, a set of hash functions employed in the scheme, etc. This algorithm may be run either by an authority or by a designated user.*

Keygen: *is a probabilistic key generation algorithm independently run by each user on input of the common public information I to produce a key pair $(\mathsf{sk}, \mathsf{pk})$.*

Sign/Encrypt: *is a probabilistic algorithm that takes as input the common public key I, a plaintext M, the recipient's public key pk_R, and the sender's private key sk_S, and outputs a ciphertext $\sigma = \mathsf{Sign/Encrypt}(M, \mathsf{sk}_S, \mathsf{pk}_R)$.*

Decrypt/Verify: *is a deterministic algorithm that takes as input a ciphertext σ, public information I and the receiver's private*

key sk *to return a plaintext* M *together with a sender's public key* pk_S *and auxiliary non-repudiation information (allowing to convince a third party of the origin of the message) or a distinguished symbol* $\perp$ *if* σ *does not properly decrypt into a message accompanied by suitable authenticating information for the sender.*

For consistency, we require that if $\sigma = \mathsf{Sign/Encrypt}(m, \mathsf{sk}_S, \mathsf{pk}_R)$*, then the output of the* $\mathsf{Decrypt/Verify}(.)$ *for the pair* (σ, sk_R) *contains a plaintext* m*, the public key* pk_S *and additional information allowing the receiver to convince a third party that the plaintext actually emanates from the sender.*

Our model considers schemes having the same interface as a sequential composition in the "sign-then-encrypt" order where the sender's public key is encrypted together with the plaintext. The receiver is not assumed to know in advance who the sender is. The latter's identity is retrieved in the decryption/verification process. This difference with models of [9], [15], [79] or [138] is motivated by the compatibility with anonymous ciphertexts. We stress here that we do not claim the above model to be more relevant than those of [9, 15, 138, 79]. We just say that it might be more appropriate for specific applications where it is of interest to have anonymous ciphertexts.

We have to mention that Boyen's work [45] was the first one to extend the notion of key privacy [25] to the signcryption setting. Somewhat surprisingly, it was only considered in the identity based setting. A similar notion was never formalized for traditional public keys. The models described in the next section thus aim at filling this gap.

3.2. Security notions

We recall the two usual security notions: the security against chosen ciphertext attacks and the unforgeability against chosen-message attacks. In the former, we consider a multi-user security model as already done in [9, 15, 79, 179] and [45] to let the adversary obtain ciphertexts created with the attacked private key under arbitrary public keys. We also consider "inside attackers" that are allowed to choose the private key under which the challenge is signcrypted: for confidentiality purposes,

we require the owner of a private key to be unable to find any information on a ciphertext created with that key without knowing which random coins were used to produce that ciphertext. This further allows us to show that an attacker stealing a private key does not threaten the confidentiality of messages previously signed and encrypted using that private key.

Definition 8.2 *We say that a signcryption scheme ensures* ***message privacy against chosen-ciphertext attacks*** *(we call this security notion IND-SC-CCA) if no PPT adversary has a non-negligible advantage in the following game:*

1. *The challenger $\mathcal{CH}$ generates a private/public key pair (sk_U, pk_U). sk_U is kept secret while pk_U is given to the adversary $\mathcal{A}$.*
2. *$\mathcal{A}$ performs a first series of queries of the following kinds:*
   - *Signature/encryption queries: $\mathcal{A}$ produces a message $m \in \mathcal{M}$ and an arbitrary public key pk_R (which may differ from pk_U) and requires the result $\mathsf{Sign/Encrypt}(m, sk_U, pk_R)$ of the signature/encryption oracle.*
   - *Decryption/verification queries: $\mathcal{A}$ produces a ciphertext σ and requires the result of $\mathsf{Decryt/Verify}(\sigma, sk_U)$ which consists of a signed plaintext together with a sender's public key if the obtained signed plaintext is valid for the recovered sender's public key and the $\perp$ symbol otherwise (indicating that the ciphertext was not properly formed).*

   These queries can be asked adaptively: each query may depend on the answers to previous ones. After a number of queries, $\mathcal{A}$ produces two plaintexts m_0, $m_1 \in \mathcal{M}$ and an arbitrary private key sk_S. $\mathcal{CH}$ flips a coin $b \overset{R}{\leftarrow} \{0,1\}$ to compute an authenticated encryption $\sigma = \mathsf{Sign/Encrypt}(m_b, sk_S, pk_U)$ of m_b with the sender's private key sk_S under the attacked public key pk_U. The ciphertext σ is sent to $\mathcal{A}$ as a challenge.
3. *$\mathcal{A}$ performs new queries as in step 2 but she may not ask the decryption/verification of the challenge σ. At the end of the game, she outputs a bit b' and wins if $b' = b$.*

$\mathcal{A}$'s advantage is defined to be $Adv^{\mathtt{ind\text{-}cca}}(\mathcal{A}) := 2 \times Pr[b' = b] - 1$.

Similarly to several other works [9, 117, 79, 140], our PKC'04 paper considered the non-repudiation property with respect to the entire ciphertext as formalized by the following definition.

Definition 8.3 *We say that a signcryption scheme is* ***strongly existentially ciphertext-unforgeable*** *against chosen-message attacks (SC-SUF-CMA) if no PPT adversary has a non-negligible advantage in the following game:*

1. *The challenger generates a key pair* (sk_U, pk_U) *and* pk_U *is given to the forger* $\mathcal{F}$*.*
2. *The forger* $\mathcal{F}$ *performs queries to oracles* $\mathsf{Sign/Encrypt}(., sk_U, .)$ *and* $\mathsf{Decrypt/Verify}(., sk_U)$ *exactly as in the previous definition. Again, these queries can also be produced adaptively.*
3. *Eventually,* $\mathcal{F}$ *produces a ciphertext* σ *and a key pair* (sk_R, pk_R) *and wins the game if the result* $\mathsf{Decrypt/Verify}(\sigma, sk_R)$ *is a tuple* (m, s, pk_U) *such that* (m, s) *is a valid signature for the public key* pk_U *such that* σ *was not the output of a signature/encryption query* $\mathsf{Sign/Encrypt}(m, sk_U, pk_R)$ *made during the game.*

As in [9, 117, 79] and many other works, the forger is allowed to have obtained the forged ciphertext as the result of a signature/encryption query for a different receiver's public key than the one corresponding to the claimed forgery. The only constraint is that, for the message m obtained by decryption/verification of the alleged forgery with the chosen private key sk_R, the produced ciphertext σ was not obtained as the result of a $\mathsf{Sign/Encrypt}(m, sk_U, pk_R)$ query.

In [45], Boyen proposed additional security notions for signcryption schemes. One of the most important ones was the notion of ciphertext anonymity that can be viewed as an extension to authenticated encryption schemes of the notion of key privacy already considered by Bellare et al in [25]. Intuitively, in the context of public key encryption, a scheme is said to have the key privacy property if ciphertexts convey no information about the public key that was used to create them. In the signcryption setting, we say that the ciphertext anonymity (or key privacy) property is satisfied if ciphertexts contain no information about who created them nor about whom they are intended to. This notion is a transposition into the non-identity based setting of the one presented

by Boyen in [45].

Definition 8.4 *We say that a signcryption scheme satisfies the* ***ciphertext anonymity property*** *(also called key privacy or key indistinguishability: we call this notion SC-INDK-CCA for short) if no PPT distinguisher has a non-negligible advantage in the following game:*

1. *The challenger generates two distinct key pairs* $(sk_{R,0}, pk_{R,0})$ *and* $(sk_{R,1}, pk_{R,1})$*. The distinguisher* $\mathcal{D}$ *is provided with* $pk_{R,0}$ *and* $pk_{R,1}$*.*
2. $\mathcal{D}$ *adaptively performs queries* $\mathsf{Sign/Encrypt}(m, sk_{R,c}, pk_R)$*, for arbitrary receivers' public keys* pk_R*, and* $\mathsf{Decrypt/Verify}(\sigma, sk_{R,c})$ *for* $c = 0$ *or* $c = 1$*. Once stage 2 is over,* $\mathcal{D}$ *outputs two private keys* $sk_{S,0}$ *and* $sk_{S,1}$ *and a plaintext* $m \in \mathcal{M}$*. The challenger then flips two coins* $b, b' \leftarrow_R \{0,1\}$ *and computes a challenge ciphertext* $\sigma = \mathsf{Sign/Encrypt}(m, sk_{S,b}, pk_{R,b'})$ *for* $\mathcal{D}$*.*
3. $\mathcal{D}$ *adaptively performs new queries as in stage 2 with the restriction that, this time, she is disallowed to ask for the decryption/verification of the challenge* σ *for the private keys* $sk_{R,0}$ *or* $sk_{R,1}$*. At the end of the game,* $\mathcal{D}$ *outputs bits* d, d' *and wins if* $(d, d') = (b, b')$*. Her advantage is defined to be* $Adv^{\mathtt{indk\text{-}cca}}(D) := Pr[(d, d') = (b, b')] - \frac{1}{4}$*.*

Again, this notion captures the security against insider attacks. Indeed, the distinguisher is allowed to choose the pair of private keys among which the one used to create the challenge ciphertext is picked by the challenger. The above definition can be viewed as a transposition to the non-identity based setting of the definition of ciphertext anonymity proposed by Boyen [45] as well as an extension of the definition of key privacy [25] to the authenticated encryption context.

4. A scheme based on the co-Diffie-Hellman problem

This section presents a signcryption scheme whose security is tightly related to the hardness of the co-Diffie-Hellman problem introduced in [43] and recalled in chapter 1.

Our solution relies on the digital signature algorithm of Boneh, Lynn and Shacham [43], recalled in chapter 2, where signatures have the form

$\sigma = xH(m) \in \mathbb{G}_1$ (where the hash function H maps arbitrary messages onto the cyclic group $\mathbb{G}_1$) for public keys $Y = xP_2 \in \mathbb{G}_2$ and can be verified by checking that $e(\sigma, P_2) = e(H(m), Y)$. In order to enhance the concrete security of the reduction in the proof of ciphertext unforgeability, a random quantity U that is used for encryption purposes also acts as a random salt to provide a tighter security reduction to the co-Diffie-Hellman problem in $(\mathbb{G}_1, \mathbb{G}_2)$.

The scheme may be viewed as a composition of a digital signature scheme which is existentially unforgeable against chosen-message attacks (EUF-CMA) [105] with a public key encryption scheme that is only secure against chosen-plaintext attacks. In [9], An, Dodis and Rabin showed that, in a multi-user setting, a composition (in any order) of an EUF-CMA signature with an IND-CCA public key encryption scheme yields a signcryption scheme that is secure in a generalized model of security against chosen-ciphertext attacks provided some precautions are taken (as a mere encrypt-then-sign composition is insecure in the multi-user setting). This section gives an example of secure composition of an EUF-CMA signature with an IND-CPA encryption scheme that yields a CCA-secure signcryption system in the sense of definition 8.2 instead of a relaxed model of CCA-security. Such a composition is not always secure in general but our example shows that it can be in some particular settings. As we will see, our scheme is secure because redundancies needed to achieve the CCA-security are embedded in the signature.

The version of the scheme that is presented here is not the original one depicted in our PKC'04 paper. The latter version of the scheme was showed to suffer from a slight security flaw that prevents it from being CCA-secure against insider attacks as pointed out in [218]. However, repairing the original scheme was quite straightforward. The version of the system that we describe here is slightly different from the one of [218]: it uses fewer random oracles and makes use of a symmetric encryption scheme. It is also described in terms of asymmetric pairings and may thus be implemented with ordinary curves.

4.1. The scheme

The consistency of the scheme is easy to verify. We note that, in the signature/encryption algorithm, the recipient's public key must be hashed together with the pair (m, U) in order to achieve the provable

Common-Keygen: given a security parameter k, this algorithm outputs a k-bit prime q, asymmetric bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ of order q such that the lengths ℓ_1 and ℓ_2 of elements from $\mathbb{G}_1$ and $\mathbb{G}_2$ are polynomial in k, an efficiently computable (but non-necessarily invertible) isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$, generators $P_2 \in \mathbb{G}_2$, $P_1 = \psi(P_2)$ and hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1$, $H_2 : \{0,1\}^* \to \{0,1\}^{\ell_1}$ and $H_3 : \{0,1\}^* \to \{0,1\}^\lambda$ where n denotes the size of plaintexts (i.e. the message space is $\mathcal{M} = \{0,1\}^n$). A symmetric encryption scheme (E, D) of keylength λ is also chosen. The common key is then

$$I = \{q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e, H_1, H_2, E, D, n, \ell_1, \ell_2\}.$$

Keygen: user u picks a random $x_u \xleftarrow{R} \mathbb{Z}_q^*$ and sets his public key to $Y_u = x_u P_2 \in \mathbb{G}_2$. His private key is x_u. We will denote the sender and the receiver respectively by $u = S$ and $u = R$ and their key pair by (x_S, Y_S) and (x_R, Y_R).

Sign/Encrypt: given a plaintext $m \in \{0,1\}^n$ intended to R, the sender S uses the following procedure

(1) Pick a random $r \xleftarrow{R} \mathbb{Z}_q^*$ and compute $U = rP_1 \in \mathbb{G}_1$.
(2) Compute $V = x_S H_1(m||U||Y_R) \in \mathbb{G}_1$ and scramble it into $W = V \oplus H_2(U||r\psi(Y_R))$.
(3) Compute $\kappa = H_3(U||V||r\psi(Y_R)) \in \{0,1\}^\lambda$ and then $Z = E_\kappa(m||Y_S) \in \{0,1\}^{n+\ell_1+\ell_2}$.

The ciphertext is given by $\sigma = \langle U, W, Z\rangle \in \mathbb{G}_1 \times \{0,1\}^{n+\ell_1+\ell_2}$.

Decrypt/Verify: when receiving a ciphertext $\sigma = \langle U, W, Z\rangle$, the receiver R has to perform the steps below:

(1) Compute $V = W \oplus H_2(U||x_R U)$ and return $\perp$ if $V \notin \mathbb{G}_1$.
(2) Compute $\kappa = H_3(U||V||x_R U) \in \{0,1\}^\lambda$.
(3) Compute $(m||Y_S) = D_\kappa(Z) \in \{0,1\}^{n+\ell_1+\ell_2}$.
(4) Compute $H = H_1(m||U||Y_R) \in \mathbb{G}_1$ and then check if $e(H, Y_S) = e(V, P_2)$. If this condition does not hold, reject the ciphertext.

FIGURE 8.1. The co-DH-signcryption scheme

strong unforgeability (as shown in the proof of theorem 8.2).

When the recipient wishes to convince a third party that a ciphertext actually emanates from the sender, he has to forward m, U, the detached signature V and his public key Y_R to that third party. As public keys

lie in group $\mathbb{G}_2$ whose elements have a large representation, it might be more convenient to implement the scheme with symmetric pairings (and thus supersingular cuves) to reduce the communication cost of the non-repudiation procedure in environments where bandwidth is a primary concern.

From an efficiency point of view, only three scalar multiplications in $\mathbb{G}_1$ are required for the signature-encryption operation while 1 multiplication and 2 pairings must be performed in the decryption/verification process. We can verify that the scheme is at least as efficient and more compact than any sequential composition of the BLS signature [43] with any one of the CCA-secure Diffie-Hellman based encryption schemes proposed in [12, 16, 71, 89, 88, 180, 200]. For example, a sequential combination of the BLS signature with the encryption scheme proposed in [12] would involve an additional multiplication at decryption because of the re-encryption phase entailed by the validity checking of the decryption algorithm. If we take $\ell_1 \approx k \geq 160$ and $\ell_2 = 1024$ (by working with an ordinary curve), we see that ciphertexts are about 1344 bits longer than plaintexts. A composition of the BLS signature with the so-called 'length-saving ElGamal encryption scheme' proposed in [12] (which is nothing but the result of applying the Fujisaki-Okamoto transformation of [88] to a variant of ElGamal) would result in ciphertexts that would be at least 1504 bits longer than plaintexts.

We observe that the scheme looks like a sequential composition of the BLS signature with the Hybrid ElGamal encryption scheme proven secure by Cramer and Shoup [74] and recalled in chapter 3. Actually, the scheme is more than just a sequential composition. Indeed, the Hybrid ElGamal scheme *must* be implemented with an IND-CCA symmetric encryption (and is trivially CCA-insecure if instantiated with an IND-CPA symmetric scheme). In contrast, our co-DH signcryption system only needs a symmetric scheme that satisfies the very weak requirement to be semantically secure against passive attacks (that is an attack where the adversary has no encryption or decryption oracle in an indistinguishability scenario such as the one of definition 3.3). For example, the symmetric encryption could simply be a "one-time pad" with a hash value of the triple $(U, Y_R, r\psi(Y_R))$. The scheme may actually be regarded as a sequential composition of an existentially unforgeable signature with a public key encryption scheme that is only secure

against chosen-plaintext attacks. All the redundancies needed to ensure the security in the sense of definition 8.2 are contained in the signature embedded in the ciphertext. In [9], it was already observed that a sequential composition in the "sign-then-encrypt" order can amplify rather than simply preserve the security properties of the underlying signature and encryption building blocks. Our co-Diffie-Hellman based construction gives another example of this fact.

4.2. Security

The original version [132] of this system (where κ was obtained by hashing v alone) was found [206, 218] not to meet its intended security properties. Although a chosen-ciphertext attack was also given [207] against the modification suggested in [218], its present variant is immune to attacks reported in [218, 206, 207] (and countermeasures do not incur any significant additional cost).

In the random oracle model, the scheme is secure under tight reductions assuming the hardness of a natural variant of the Diffie-Hellman problem.

Theorem 8.1 *Assume that an adversary $\mathcal{A}$ has non-negligible advantage ϵ over the IND-SC-CCA security of the above scheme when running in time t and performing q_{se} signature/encryption queries, q_{dv} decryption/verification queries and q_{H_i} queries to oracles H_i (for $i = 1, 2$). Then, for any $0 \leq \nu \leq \epsilon$, there either exists*

- *an algorithm $\mathcal{B}$ that can solve the co-CDH problem in groups $(\mathbb{G}_1, \mathbb{G}_2)$ with probability $\epsilon' \geq \epsilon - \nu - q_{dv}/2^k$ within time*

$$t' < t + O(q_{dv} + q_{H_2} + q_{H_3})t_p$$

  where t_p denotes the time required for a pairing evaluation.
- *a passive adversary breaking the semantic security of the symmetric scheme (E, D) with advantage ν within time t'.*

Proof. We show how to build an algorithm $\mathcal{B}$ that runs the attacker $\mathcal{A}$ as a subroutine to solve the co-Diffie-Hellman problem in a polynomial time. Let $(aP_1, bP_2) \in \mathbb{G}_1 \times \mathbb{G}_2$ be a random instance of the co-CDH problem. Algorithm $\mathcal{B}$ starts $\mathcal{A}$ with $Y_u = bP_2 \in \mathbb{G}_2$ as a challenge public key. $\mathcal{A}$ then adaptively performs queries that are handled using lists L_1, L_2 and L_2' to keep track of the answers given to random oracle

queries on H_1 and H_2.

- H_1-queries: when a hash query $H_1(m||U||Y_R)$ is made, $\mathcal{B}$ first looks if the value of H_1 was previously defined for the input $m||U||Y_R$. If it was, the previously defined value is returned. Otherwise, $\mathcal{B}$ picks a random $t \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$, returns $tP_1 \in \mathbb{G}_1$ as an answer and inserts the tuple (m, U, Y_R, t) into L_1.
- H_2-queries on inputs $U_i||R_i \in \mathbb{G}_1^2$: $\mathcal{B}$ checks if (P_2, U_i, Y_u, R_i) is a valid co-Diffie-Hellman quadruple (in our notation, we write $R_i = \text{co-DH}_{P_2}(U_i, Y_u)$) by checking whether $e(R_i, P_2) = e(U_i, Y_u)$. If it is then
  - if $U = aP_1$, $\mathcal{B}$ halts and outputs $R_i = abP_1 \in \mathbb{G}_1$ which is the solution that it was looking for.
  - $\mathcal{B}$ checks if L_2' contains an entry of the shape $(U_i, ., h_{2,i})$ for some $h_{2,i} \in \{0,1\}^{\ell_1}$. In this case, $h_{2,i}$ is returned and a record $(U_i, R_i, h_{2,i}, 1)$ is stored in L_2. If no entry $(U_i, ., h_{2,i})$ exists in L_2', $\mathcal{B}$ returns a random string $h_{2,i} \stackrel{R}{\leftarrow} \{0,1\}^{\ell_1}$ and inserts $(U_i, R_i, h_{2,i}, 1)$ in L_2.

  If (P_2, U_i, Y_u, R_i) is not a co-DH tuple, the simulator stores the 5-uple $(U_i, R_i, h_{2,i}, 0)$ in L_2.
- H_3-queries on triples $U_i||V_i||R_i$: $\mathcal{B}$ proceeds as for answering H_2-queries, using lists L_3 and L_3' to maintain the consistency and checking if (P_2, U_i, Y_u, R_i) is a co-DH tuple. Again, the simulator halts and outputs R_i if a co-DH tuple (aP_1, V_i, R_i) is queried.
- For a signature/encryption query on a plaintext m with a recipient's public key Y_R both chosen by $\mathcal{A}$, $\mathcal{B}$ first picks a random $r \stackrel{R}{\leftarrow} \mathbb{Z}_q$, computes $U = rP_1 \in \mathbb{G}_1$ and checks if L_1 contains a tuple (m, U, Y_R, t) indicating that $H_1(m||U||Y_R)$ was previously set to tP_1. If no such tuple is found, $\mathcal{B}$ picks $t \stackrel{R}{\leftarrow} \mathbb{Z}_q$ and stores the entry (m, U, Y_R, t) in L_1. It then computes $V = t\psi(Y_u) = t(bP_1) \in \mathbb{G}_1$. The rest follows as in the normal process: $\mathcal{B}$ computes $r\psi(Y_R)$ (for the Y_R specified by the adversary), simulates H_2 and H_3 to obtain $h_2 = H_2(U||r\psi(Y_R))$,

$\kappa = H_3(U||V||r\psi(Y_R))$ and then computes $W = V \oplus h_2$ and $Z = E_\kappa(m||Y_u)$. The ciphertext $\langle U, W, Z\rangle$ is then returned.

- Decryption/verification queries for a ciphertext $C = \langle U, W, Z\rangle$: $\mathcal{B}$ checks if L_2 contains the sole possible 5-uple $(U, Y_u, R, h_2, 1)$ for some elements $R \in \mathbb{G}_1$ and $h_2 \in \{0,1\}^{\ell_1}$ (meaning that $R = \text{co-DH}_{P_2}(U, Y_u)$ and that $H_2(U||R)$ was set to $h_2 \in \{0,1\}^\lambda$):
  - if it does, $\mathcal{B}$ obtains $V = Z \oplus h_2$ and rejects C if $V \notin \mathbb{G}_1$. Otherwise, $\mathcal{B}$ obtains $\kappa = H_3(U||V||R) \in \{0,1\}^\lambda$ (by simulating H_3), sets $(m||Y_S) = D_\kappa(Z)$ and rejects C if $Y_S \notin \mathbb{G}_2$. Otherwise, it computes $H = H_1(m||U||Y_R) \in \mathbb{G}_1$ and checks whether $e(V, P_2) = e(H, Y_S)$. If yes, the information (m, V, Y_S, Y_R) is returned. Otherwise, C is declared invalid.
  - if not, $\mathcal{B}$ draws $h_2 \xleftarrow{R} \{0,1\}^{\ell_1}$ and stores a record $(U, ., h_2)$ in L_2' so as to answer h_2 to a subsequent H_2-query on the input $U||\text{co-DH}_{P_2}(U, Y_u)$. It sets $V = W \oplus h_2 \in \{0,1\}^{\ell_1}$ (and rejects C if $V \notin \mathbb{G}_1$) and inserts a record $(U, V, ., \kappa)$ in L_3' so that a future hash query $H_3(U||V||\text{co-DH}_{P_2}(U, Y_u))$ will get κ as an answer. Finally, $D_\kappa(Z)$ is computed and parsed into $(m||Y_S) \in \{0,1\}^{n+\ell_2}$. The ciphertext C is declared invalid if $Y_S \notin \mathbb{G}_2$ or $e(V, P_2) \neq e(H, Y_S)$ where $H = H_1(m||U||Y_R)$.

At the end of the first stage, $\mathcal{A}$ outputs plaintexts m_0 and m_1 together with an arbitrary sender's private key x_S and requires a challenge ciphertext built under the recipient's public key Y_u. $\mathcal{B}$ then picks a random bit $d \xleftarrow{R} \{0,1\}$ and strings $\kappa^* \xleftarrow{R} \{0,1\}^\lambda$, $W \xleftarrow{R} \{0,1\}^{\ell_1}$. It obtains $H^* = H_1(m_d||aP_1||Y_u) \in \mathbb{G}_1$ by simulating H_1 to compute the challenge ciphertext $\sigma = \langle U, W, Z\rangle = \langle aP_1, W, E_{\kappa^*}(m_d||x_S P_2)\rangle$ which is sent to $\mathcal{A}$. The latter then issues a second series of queries which are handled as in the first stage. If the symmetric scheme (E, D) is semantically secure against passive attacks, it is easy to show that $\mathcal{A}$'s view is independent from the hidden bit $d \in \{0,1\}$ unless she queries a tuple containing abP_1 to random oracles H_2 or H_3. In either case, the solution of the co-Diffie-Hellman problem is detected when answering H_2 and H_3-queries.

Now to assess $\mathcal{B}$'s probability of success, let us denote by AskH the event that $\mathcal{A}$ asks the hash value of abP_1) during the simulation. As done

in several papers in the literature (see [200, 40, 45] for instance), as long as the simulation of the attack's environment is perfect, the probability for AskH to happen is the same as in a real attack (i.e. an attack where $\mathcal{A}$ interacts with real oracles). In a real attack we have

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

where ν denotes the maximal advantage of any passive adversary against the semantic security of the symmetric scheme (E, D). Since by definition, $\epsilon = 2\times\Pr[b=b']-1$, we may write $\Pr[\mathsf{AskH}]\geq\epsilon-\nu$.

We note that the simulation only fails if the random challenge ciphertext was submitted to the Decrypt/Verify oracle before the challenge phase. Such a very unlikely event happens with probability smaller than $q_{dv}/2^k$. The bound on $\mathcal{B}$'s computation time derives from the fact that decryption/verification queries and H_2-queries each require 2 pairing evaluations. □

We observe that the reduction is very tight in that, up to negligible terms, algorithm $\mathcal{B}$ has the same probability to solve the co-CDH problem as the adversary's advantage. Moreover, the cost of the reduction is bounded by an expression which is linear in the number of adversarial queries. That is the reason why U is included among the arguments of H_2. The scheme remains secure is V is concealed by a hash value of $r\psi(Y_R)$ alone but the reduction then entails a number of pairing evaluation that is quadratic in the number of adversarial queries (more precisely, up to $2q_{h_2}$ pairings might have to be calculated for each decryption/verification queries).

The reductions are also efficient in the proofs of ciphertext unforgeability and anonymity.

Theorem 8.2 *Let an adversary $\mathcal{F}$ having non-negligible advantage ϵ over the SC-SUF-CMA security of the scheme when running in time t, making q_{se} signature/encryption queries, q_{dv} decryption/verification queries and q_{H_i} queries on oracles H_i (for $i=1,2$). Then there is an algorithm $\mathcal{B}$ solving the co-CDH problem in $(\mathbb{G}_1,\mathbb{G}_2)$ with probability*

$\epsilon' > \epsilon - (q_{se}(q_{H_1} + q_{se}) + 1)/2^k$ in time $t' < t + O(q_{H_2} + q_{H_3} + q_{dv})t_p$, t_p being the time required for a pairing evaluation.

PROOF. $\mathcal{B}$ receives a random instance (aP_1, bP_2) of the co-Diffie-Hellman problem. It uses $\mathcal{F}$ as a subroutine to solve that instance and plays the role of $\mathcal{F}$'s challenger. The forger $\mathcal{F}$ is initialized with $Y_u = bP_2$ as a challenge public key. She then performs adaptive queries that are handled as explained below (using lists L_1, L_2 and L_2' as in the proof of theorem 8.1):

- H_2-queries: are dealt with as in the proof of theorem 8.1.
- H_1-queries on a tuple $m||U||Y_R$: if the latter was previously queried, $\mathcal{B}$ returns the defined value. For a query on a new tuple $m||U||Y_R$, $\mathcal{B}$ picks $t \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and defines $H_1(m||U||Y_R) = t(aP_1) \in \mathbb{G}_1$. The list L_1 is updated accordingly.
- Signature/encryption queries for a message m and a receiver's public key Y_R chosen by $\mathcal{F}$: $\mathcal{B}$ picks $r \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$, computes $U = rP_1 \in \mathbb{G}_1$. If H_1 is already defined on $m||U||Y_R$, $\mathcal{B}$ outputs "failure" and halts. Otherwise, $\mathcal{B}$ picks $t \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$, sets $H_1(m||U||Y_R) = tP_1 \in \mathbb{G}_1$ and updates L_1 accordingly. It then computes $V = t\psi(Y_u) \in \mathbb{G}_1$, $h_2 = H_2(U||r\psi(Y_R)) \in \{0,1\}^\lambda$, $W = V \oplus h_2 \in \{0,1\}^{\ell_1}$, $\kappa = H_3(U||V||r\psi(Y_R)) \in \{0,1\}^\lambda$ and $Z = E_\kappa(m||Y_u) \in \{0,1\}^{n+\ell_2}$. The ciphertext $\langle U, W, Z\rangle$ is then returned to $\mathcal{F}$.
- Decryption/verfication queries: are handled exactly as in the proof of theorem 8.1.

At the end of the game, $\mathcal{F}$ produces a ciphertext $\langle U^*, W^\star, Z^*\rangle$ and a recipient's public/private key pair (x_R^*, Y_R^*). At that moment, $\mathcal{B}$ can perform the decrytion/verification operation using x_R^* and, if the ciphertext is a valid forged ciphertext for the sender's public key Y_u, $\mathcal{B}$ can extract the message-key pair $(m^*||Y_u)$ and the signature V^*. If the hash value $H_1(m^*||U^*||Y_R^*)$ was not asked by $\mathcal{F}$ during the simulation, $\mathcal{B}$ reports "failure" and stops. Otherwise, $H_1(m^*||U^*||Y_R^*)$ must have been set to $t^*(aP_1)$, for some known $t^* \in \mathbb{Z}_q^*$, and V' must be equal to $t^*(abP_1)$ which yields the co-Diffie-Hellman solution $t^{*-1}V'$. It is easy to see that the probability for $\mathcal{B}$ to fail in answering a signature/encryption query is not greater than $q_{se}(q_{h_1} + q_{se})/2^k$ (since at each signature/encryption query, there is at most $q_{H_1} + q_{se}$ elements in L_1). Besides, the probability

that $\mathcal{F}$ succeeds in her attempt without asking the query $H_1(m^*, U^*, Y_R^*)$ is at most $1/2^k$.

□

Theorem 8.3 *Assume there exists a PPT distinguisher $\mathcal{D}$ that has non-negligible advantage over the SC-INDK-CCA security of the scheme when running in time t, performing q_{se} signature/encryption queries, q_{dv} decryption/verification queries and q_{H_i} queries to oracle H_i (for $i = 1, 2$). Then, for any $0 \leq \nu \leq \epsilon$, there either exists*

- *an algorithm $\mathcal{B}$ that solves the co-CDH problem with advantage*
$$\epsilon' > \epsilon - \nu - \frac{q_{dv}}{2^k}$$
when running in time $t' < t + O(q_{dv} + q_{H_2} + q_{H_3})t_p$ where t_p denotes the time required for a pairing calculation.
- *a passive attacker breaking the semantic security of (E, D) with advantage $2\nu - \frac{1}{2}$ within time t'.*

Proof. Let (aP_1, bP_2) be an instance of the co-CDH problem. $\mathcal{B}$ uses $\mathcal{A}$ to solve that instance and plays the role of $\mathcal{D}$'s challenger in the game of definition 8.4. To do this, $\mathcal{B}$ picks random elements $x, y \in \mathbb{Z}_q^*$ and initializes $\mathcal{D}$ with $pk_{u,0} = Y_{u,0} = x(bP_2)$ and $pk_{u,1} = Y_{u,1} = y(bP_2)$. The distinguisher $\mathcal{D}$ then issues queries as explained in definition 8.4. To deal with these queries, $\mathcal{B}$ maintains lists L_1, L_2 and L_2' as in previous theorems. When oracle H_2 is queried, $\mathcal{B}$ performs the same manipulations as in previous proofs to maintain consistency with decryption/verification queries. All queries are treated exactly in the same way as in the proof of theorem 8.1 except that two distinct private keys $Y_{u,0}$ and $Y_{u,1}$ may be involved in calls to oracles Sign/Encrypt and Decrypt/Verify. It can be checked that it does not change anything to the strategy of the simulator that never fails.

Once the first stage is over, $\mathcal{D}$ outputs two private key $sk_{S,0} = x_{S,0} \in \mathbb{Z}_q^*$, $sk_{S,1} = x_{S,1} \in \mathbb{Z}_q^*$ and a plaintext $m \in \{0,1\}^n$. $\mathcal{B}$ then chooses two independent bits $(d, d') \in \{0,1\} \times \{0,1\}$ and random strings $\kappa \xleftarrow{R} \{0,1\}^\lambda$, $W \xleftarrow{R} \{0,1\}^{\ell_1}$, obtains $H = H_1(m||aP||Y_{u,d'}) \in \mathbb{G}_1$ and computes $Z = E_\kappa(m||x_{S,d}P_2) \in \{0,1\}^{n+\ell_2}$ before sending the challenge $\sigma = \langle aP, W, Z\rangle$ to $\mathcal{D}$. Clearly, if (E, D) is a semantically secure symmetric scheme, as long as no H_2 or H_3-query is made on a tuple containing

$x(abP_1)$ or $y(abP_1)$, $\mathcal{F}$'s view is independent from the hidden bits (d, d'). Moreover, such H_2 or H_3-queries would provide $\mathcal{B}$ with the co-CDH solution abP_1 that would be detected when handling hash queries.

$\mathcal{D}$'s second series of queries is handled as in the first stage. Eventually, $\mathcal{D}$ outputs a guess (d, d') which is ignored. We note that in a real attack, we have

$$\begin{aligned}\Pr[(d,d')=(b,b')] &\leq \Pr[(d,d')=(b,b')|\neg\mathsf{AskH}]\Pr[\neg\mathsf{AskH}]+\Pr[\mathsf{AskH}]\\ &= \big(\nu+\frac{1}{4}\big)(1-\Pr[\mathsf{AskH}])+\Pr[\mathsf{AskH}]\\ &\leq \nu+\frac{1}{4}+\frac{3}{4}\Pr[\mathsf{AskH}]\end{aligned}$$

where ν is the maximal advantage of a passive adversary against the symmetric scheme in distinguishing between the encryptions of four messages of her choosing (this advantage being defined as the probability of a correct guess minus 1/4). Indeed, the second equality derives from the fact that $\Pr[(d, d') = (b, b')|\neg\mathsf{AskH}]$ is nothing but the probability of a passive adversary to distinguish between the encryption of four plaintexts for an unknown random symmetric key. As $\Pr[(d, d') = (b, b')] = \epsilon + 1/4$ by definition, we thus find $\Pr[\mathsf{AskH}] \geq \frac{4}{3}(\epsilon - \nu) > (\epsilon - \nu)$. As in theorem 8.1, one can show that, as long as the simulation is perfect, the probability for AskH to happen is the same in the simulation as in a real attack and as in theorem 8.1, the simulator only fails with negligible probability. Its strategy to solve the co-CDH problem is to check two candidates co-CDH tuples $(aP_1, x(bP_2), .)$ or $(aP_1, y(bP_2), .)$ for each H_2 or H_3-query involving aP_1 and the bound on its running time follows.

The announced bound for the advantage in breaking the semantic security of the symmetric scheme in the event $\neg\mathsf{AskH}$ stems from the fact that an attacker distinguishing between the encryption of 4 messages with probability $\nu+1/4$ implies a distinguisher winning a traditional indistinguishability game with the same probability and thus an advantage $2(\nu + 1/4) - 1 = 2\nu - 1/2$. □

5. A scheme providing short detachable signatures

Despite its good concrete security, the scheme depicted in the previous section remains of moderate interest because it only offers very

few computational or bandwidth savings over a sequential composition of signature and encryption schemes. It indeed requires two pairing calculations upon decryption/verification and three elliptic curve scalar multiplications on the sender's side whereas Zheng's original scheme features a signcryption operation that entails a single group exponentiation (but it is far from having all of the useful properties of our scheme: it is much less practical from a non-repudiation point of view for instance).

It is thus natural to wonder whether it is possible to devise a system having the same properties as co-DH signcryption with an increased efficiency. The present section proposes a more efficient Diffie-Hellman based signcryption solution published at SCN'04 [135] that satisfies the same strong security requirements: the new scheme has essentially the same cost as a mere ElGamal encryption on the sender's side while only one pairing evaluation and three exponentiations are required for the simultaneous decryption/verification tasks. This is a real efficiency improvement: the signature/encryption operation is roughly 33% faster than in co-DH signcryption while the decryption/verification algorithm is almost twice as efficient. In addition, the scheme does provide the non-repudiation property in a very simple and natural way as it allows receivers to extract ordinary signatures on the plaintext from received ciphertexts.

The system is built on a digital signature scheme independently studied by Boneh-Boyen [37] and Zhang et al. [227]. The latter signature scheme was also implicitly used in the private key extraction algorithm of the Sakai-Kasahara IBE [189]. It provides signatures of the form

$$\sigma = \frac{1}{x + h(m)} P_1,$$

for public keys $Y = xP_2$, that are verified via a single pairing calculation: the right hand side of the verification equation $e(\sigma, H(m)P_2 + Y) = e(P_1, P_2)$ is included as a part of the signer's public key.

As a result, our system features detachable signatures made of a single group element. When implemented with ordinary curves such as MNT curves [155], the protocol provides detachable signatures of only 171-bits. To the best of our knowledge, our proposal is the first practical signcryption protocol supporting short detachable signatures. The price to pay for such improvements is that our scheme's security relies

on stronger assumption than the one depicted in section 4: the message privacy is proved in the random oracle model under the p-Diffie-Hellman Inversion assumption already considered in [37, 35, 227] and also employed in chapter 5 while the scheme's unforgeability relies on the p-Strong Diffie-Hellman assumption introduced by Boneh and Boyen [37]. Both assumptions are recalled in the following definition:

Definition 8.5 *Let us consider bilinear map groups* $(\mathbb{G}_1, \mathbb{G}_2)$ *of generators* P *and* Q*.*

- *The* **p-*Diffie-Hellman Inversion problem*** *(q-DHI) consists in, given a tuple* $(P, Q, xQ, x^2Q, \ldots, x^pQ) \in \mathbb{G}_1 \times \mathbb{G}_2^{p+1}$*, computing* $\frac{1}{x}P_1 \in \mathbb{G}_1$*.*
- *The* **p-*Strong Diffie-Hellman problem*** *(q-SDH) consists in, given a tuple* $(P, Q, xQ, x^2Q, \ldots, x^pQ) \in \mathbb{G}_1 \times \mathbb{G}_2^{p+1}$*, coming up with a pair* $(c, \frac{1}{x+c}P) \in \mathbb{Z}_q \times \mathbb{G}_1$*.*

One of the steps towards the realization of our scheme was to change our security model for the non-repudiation property. Similarly to what Boyen [45] did in the identity based setting, we do no longer require the unforgeability property against insider adversaries but rather consider the non-repudiation for embedded plaintexts.

5.1. Considering non-repudiation for detached signatures only

Beside our co-Diffie-Hellman based scheme presented in section 4, a lot of provably unforgeable signcryption schemes [9, 15, 79, 117, 179] provide non-repudiation with respect to the whole ciphertext. As noticed in [45], in many contexts, it is sufficient to only consider the non-repudiation with respect to the signature embedded in the ciphertext. Even though we still doubt on whether non-repudiation with respect to entire ciphertexts is a relevant or useful security notion, for applications that would be requiring it, we will show how to turn our scheme into a ciphertext-existentially unforgeable one at the cost of increasing the size of the detachable signatures by a factor of more than 3 (but without any other loss of efficiency). The notion of unforgeability w.r.t. embedded signatures, that was introduced for the first time in [45], is recalled below.

Definition 8.6 *A signcryption scheme is* ***existentially signature-unforgeable*** *against chosen-message attacks (or has the ESUF-SC-CMA security) if no PPT adversary has a non-negligible advantage against a challenger $\mathcal{CH}$ in the following game:*

1. *$\mathcal{CH}$ generates a key pair (sk_U, pk_U) and pk_U is given to the forger $\mathcal{F}$.*
2. *$\mathcal{F}$ adaptively performs queries to the same oracles as in definition 8.2.*
3. *$\mathcal{F}$ produces a ciphertext σ and a key pair (sk_R, pk_R) and wins if the result of $\mathsf{Decrypt/Verify}(\sigma, sk_R)$ is a triple (m, s, pk_U) such that the pair (m, s) is valid for the public key pk_U and no query to the signature/encryption oracle involving the message m and some receiver's public key pk'_R resulted in a ciphertext σ' for which the output of $\mathsf{Decrypt/Verify}(\sigma', sk'_R)$ is (m, s, pk_U).*

As stressed by Boyen [45], considering non-repudiation only w.r.t. signatures is useful for schemes providing detachable signatures that should be *unlinkable* to the ciphertext they were conveyed in: anyone seeing a valid message-signature pair can use his/her private key to encrypt it into a valid signcryption under his/her public key.

A complementary notion to the latter was also introduced in [45] in the identity based setting and recalled in chapter 7. It was called *ciphertext authentication* and means that a receiver is always convinced that a ciphertext was jointly signed and encrypted by the same person and was not subject to a kind of man-in-the-middle attack.

Definition 8.7 *We say that a signcryption scheme satisfies the* ***ciphertext authentication*** *property (AUTH-SC-CMA) if no PPT adversary has a non-negligible advantage in the following game:*

1. *The challenger generates two key pairs (sk_S, pk_S), (sk_R, pk_R) and pk_S and pk_R are given to the forger.*
2. *The forger $\mathcal{F}$ performs queries to the oracles $\mathsf{Sign/Encrypt}(., sk_U, .)$ and $\mathsf{Decrypt/Verify}(., sk_U)$, for both $U = S$ and $U = R$, exactly as in the previous definition. Again, these queries can also be produced adaptively.*
3. *At the end of the game, $\mathcal{F}$ produces a ciphertext σ and wins the game if the result of the operation $\mathsf{Decrypt/Verify}(\sigma, sk_R)$ is*

a triple (m, s, pk_S) *such that* (m, s) *is a valid signature for the public key* pk_S *such that no query to the signature/encryption oracle involving the message* m *and the receiver's public key* pk_R *resulted in the ciphertext* σ*.*

5.2. The scheme

Common-Keygen: given a security parameter k,this algorithm outputs a k-bit prime number p and the description of bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ of order q. Let ℓ_1 and ℓ_2 be polynomials in k respectively denoting the bitlength of elements from $\mathbb{G}_1$ and $\mathbb{G}_2$. The algorithm also chooses generators $P_1 \in \mathbb{G}_1$ and $P_2 \in \mathbb{G}_2$ with $P_1 = \psi(P_2)$, hash functions $h_1 : \{0,1\}^* \to \mathbb{Z}_q$, $h_2 : {\mathbb{G}_1}^3 \to \{0,1\}^{k+1}$ and $h_3 : \{0,1\}^k \to \{0,1\}^\lambda$ as well as a pseudo-random function $h' : \{0,1\}^* \to \{0,1\}$. The common public parameters are

$$\mathtt{param} = \{q, \mathbb{G}_1, \mathbb{G}_2, P_1, P_2, h_1, h_2, h_3, h', n\}$$

where n stands for the length of plaintexts.

Keygen: each user picks $x_u \xleftarrow{R} \mathbb{Z}_q^*$ and computes $Y_u = x_u P_2 \in \mathbb{G}_2$ obtains a public/private key pair $(pk_u, sk_u) = (Y_u, x_u)$.

Sign/Encrypt: given a message $m \in \{0,1\}^n$, the recipient's public key Y_R and her private key x_S, the sender does the following:

(1) Pick $\gamma \xleftarrow{R} \mathbb{Z}_q^*$ and compute $r = \frac{\gamma}{h_1(b_m||m||Y_S)+x_S} \bmod q$ where $b_m = h'(x_S, m) \in \{0,1\}$.

(2) Set $C_1 = rP_1 \in \mathbb{G}_1$, $C_2 = (\gamma||b_m) \oplus h_2\big(C_1||Y_R||r\psi(Y_R)\big) \in \{0,1\}^{k+1}$ and then $C_3 = (m||Y_S) \oplus h_3(\gamma) \in \{0,1\}^{n+\ell_2}$.

The ciphertext is

$$\langle C_1, C_2, C_3\rangle = \langle rP_1, (\gamma||b_m) \oplus h_2\big(rP_1||Y_R||r\psi(Y_R)\big), (m||Y_S) \oplus h_3(\gamma)\rangle$$

Decrypt/Verify: given $C = \langle C_1, C_2, C_3\rangle$,

(1) Compute $(\gamma||b_m) = C_2 \oplus h_2(C_1||Y_R||x_R C_1) \in \{0,1\}^k$. Return $\perp$ if $\gamma \notin \mathbb{Z}_q^*$.

(2) Compute $(m||Y_S) = C_3 \oplus h_3(\gamma) \in \{0,1\}^{n+\ell_2}$.

(3) Compute $\sigma = \gamma^{-1} C_1 \in \mathbb{G}_1$ and accept the message if

$$e\big(\sigma, Y_S + h_1(b_m||m||Y_S)P_2\big) = e(P_1, P_2). \tag{8}$$

FIGURE 8.2. The q-DH-signcryption scheme

The protocol relies on a signature scheme independently proposed by Zhang et al. [227] and Boneh and Boyen [37]. In the latter papers, this scheme was shown to efficiently produce 160-bit signatures without requiring the use of a special hash function mapping messages to be signed onto an elliptic curve subgroup (unlike the original BLS short signature proposed in [43]). In [37], it was also showed that this scheme has a more efficient security reduction in the random oracle model under the p-strong Diffie-Hellman assumption than the reduction given by Zhang et al. [227] under the p-Diffie-Hellman-Inversion assumption.

The protocol makes use of such a (masked) signature as an ElGamal like ephemeral key as well as a checksum showing that a message was properly encrypted in chosen-ciphertext security concerns: the sender first computes a multiplier $r = \gamma/(h_1(b_m||m||Y_S) + x_S) \in \mathbb{Z}_q^*$ where γ is randomly chosen from $\mathbb{Z}_q^*$, $m \in \{0,1\}^*$ is the message to sign and encrypt and b_m is a message-dependent bit computed as a pseudo-random function of m and the private key x_S according to Katz and Wang's proof technique [121] (that aims at achieving tight security reductions without random salts). This multiplier r is then used to compute an ephemeral Diffie-Hellman key rP_1 as in the ElGamal cryptosystem [96] and to scramble the secret γ using a hash value of $r\psi(Y_R) \in \mathbb{G}_1$ while a digest of γ is used to conceal the message m together with the sender's public key.

The use of a masked signature as a "one-time" Diffie-Hellman key allows sparing one exponentiation (actually an elliptic curve scalar multiplication) w.r.t. a sequential signature/encryption composition.

When computing the second component of the ciphertext, the receiver's public key and the first component (which is an embedded signature as well as a Diffie-Hellman ephemeral key) are hashed together with the "one-time" Diffie-Hellman key $r\psi(Y_R)$ in order to achieve a more efficient reduction in the security proof.

In order to convince a third party that a recovered message m emanates from the sender S, the receiver reveals σ, the message m and the associated bit b_m to the third party who can run the regular signature verification algorithm as done in step 3 of the decryption/verification algorithm. The scheme thus provides detachable signatures that cannot be linked to their original ciphertext: the signature is masked by a randomly chosen factor γ and anyone observing a valid message-signature

pair can use his/her private key to build a signcryption of that message-signature pair under his/her public key. Hence, the scheme provides *ciphertext unlinkability* in the sense of Boyen [45] in a very simple manner.

As Boyen's identity based scheme, the present one is obviously not existentially ciphertext-unforgeable in the sense of [132] (because of its inherent ciphertext unlinkability property). As already mentioned in chapter 7, we believe that it is actually useless to consider ciphertext non-repudiation (that appears as being antagonist to the useful notion of ciphertext unlinkability and might even be undesirable) rather than the mere signature non-repudiation: an adversary should not be rewarded for achieving the trivial task of using a valid signature and a randomly chosen x'_R as a recipient's private key to output a claimed forged ciphertext under the public key $Y'_R = x'_R P_1$.

5.3. Efficiency discussions.

As mentioned above, the scheme is efficient since, beside a modular inversion, the sender only computes two scalar multiplications in $\mathbb{G}_1$. The receiver's workload is dominated by one pairing computation (as $e(P_1, P_2)$ can be computed once-and-for-all and cached in memory), two scalar multiplications in $\mathbb{G}_1$ and one multiplication in $\mathbb{G}_2$. For both the sender and the receiver, the protocol is much more efficient than the scheme described in the previous section.

The scheme is described in terms of asymmetric pairings and requires the existence of a publicly computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$. Similarly to the fast identity based scheme described in chapter 7, it does not require hashing arbitrary strings onto cyclic elliptic curve subgroups. Hence, the kind of groups suggested in section 4 of [203] may be employed here and allows performing the last step of the Decrypt/Verify algorithm at a reasonable speed by using specialized techiques for ordinary curves due to Barreto, Lynn and Scott [43].

Interestingly, unlike what appears at first glance, the two exponentiations that are the bulk of the sender's workload can be computed offline (i.e. before knowing the message to be sent). Indeed, in an offline phase, the sender can already pick a random $r \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$, compute $c_1 = rP_1$ and $\omega = r\psi(Y_R)$, store them in memory and then, once

the message m is known, compute $\gamma = r(h_1(b_m||m||Y_S) + x_S) \bmod q$, $C_2 = (\gamma||b_m) \oplus h_2(C_1||Y_R||\omega) \in \{0,1\}^k$ and $C_3 = (m||Y_S) \oplus h_3(\gamma)$. In this case, care must be taken not to re-use the same r to sign and encrypt distinct messages because this would expose the private key. This is not a problem since all signatures obtained through the Fiat-Shamir heuristic [84] have this feature. In the absence of reliable pseudo-random generators, the pre-computation phase may choose random powers r as digests (computed using a collision-resistant hash function) of the message and the signer's public/private key pair according to a technique suggested in [158].

Similarly to the Boneh-Boyen signature [37], our scheme can benefit from a great *on-the-fly* efficiency although the signature scheme on which it is constructed does not support the use of "coupons" which is here allowed thanks to the presence of the blinding factor γ .

From a bandwidth point of view, we note that receivers' public keys that are scrambled along with plaintexts in the last part of ciphertexts are $\mathbb{G}_2$ elements of rather long representation. Indeed, for recommended parameters [43] their length is usually at least $\ell = 1024$. However, assuming that the scheme is supported by a public key infrastructure, it is fairly reasonable to encrypt a short (less than 128 ASCII characters) sender-identifier ID_S instead of a public key together with the plaintext in the second step of the Sign/Encrypt algorithm. The receiver then has to perform an online lookup in a public repository to enquire for the associated public key.

Finally, the present version of the protocol is devised to sign and encrypt messages of bounded size n. In applications that require to handle messages of variable length, the message-public key pair may be symmetrically encrypted using a hash value of γ as a secret key instead of being concealed by a "one-time pad". The employed symmetric encryption scheme then only has to be semantically secure against passive attacks (i.e. attacks must be unable to distinguish two encryptions of plaintexts of their choosing without having access to decryption nor encryption oracles) which is a very weak requirement. The security proof of section 5.5 is easily adaptable to the latter hybrid setting.

5.4. Short detachable signatures.

We have to mention that implementing the scheme with symmetric pairings $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ (and thus supersingular curves) might result in a faster Decrypt/Verify algorithm in some cases. Indeed, the implementation suggested in section 5 entails a scalar multiplication in $\mathbb{G}_2$ in step 3 of Decrypt/Verify and arithmetic operations in $\mathbb{G}_2$ are known to be more expensive than in $\mathbb{G}_1$ (recall that, if $\mathbb{G}_1$ is a subgroup of a curve $E(\mathbb{F}_r)$ of embedding degree α, $\mathbb{G}_2$ is usually a cyclic subgroup of $E(\mathbb{F}_{r^\alpha})$ or a subgroup of the twisted curve $\overline{E}(\mathbb{F}_{r^{\alpha/2}})$). For applications where speed is primary concern, using supersingular curves and the group $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$ (where the isomorphism ψ is the identity map) or ordinary curves of low embedding degree (ideally $\alpha = 2$) might be preferable.

On the other hand, ordinary curves of larger embedding degree (say $\alpha = 6$ for instance) should be preferred for applications where bandwidth considerations have priority. Indeed, because of the increased efficiency of particular algorithms such as [67] for solving discrete logarithms in finite fields of small characteristic, supersingular curves in characteristic 3 are known to offer fewer security than ordinary ones for the same length of group elements. As explained in [43], they are thus not recommended for applications requiring short signatures: examples given in [171] show that, for the security level of a 1024-bit RSA modulus, supersingular curves disallow signatures shorter than 260 bits whereas ordinary MNT curves [155] allow signatures of 171 bits with the Boneh-Lynn-Shacham scheme [43]. For the security level of a 1842-bit RSA modulus, supersingular curves require signatures of 560 bits whereas 308 bits are sufficient with MNT curves.

Those reasons motivated us to describe the scheme in terms of asymmetric pairings. In such an algebraic setting, receivers can extract signatures of 171 bits from ciphertexts. Assuming the deployment of a suitable PKI, ciphertexts may even avoid conveying sender's encrypted public keys: in step 2 of the signature/encryption algorithm, senders may just append their short identifier ID_S to the plaintext. In order to prevent the threat of impersonators replacing ID_S with their own identifier in public repositories, senders may also append ID_S to the plaintext

when computing the multiplier r in such a way that C_1 contains a signature on the extended message $m||\mathsf{ID}_S$.

From a computational point of view, all the sender's arithmetic operations take place in $\mathbb{G}_1$ while, besides a pairing evaluation, receivers have to perform a single scalar multiplication in $\mathbb{G}_2$.

5.5. Security

The original version of the scheme [135] was shown in [208] to be vulnerable to a chosen-ciphertext attack taking advantage of a key substitution attack [205] on the underlying signature scheme [227, 37]. However, protecting the scheme against the attack of [208] was rather straightforward using a standard countermeasure to immunize signature schemes from key substitution attacks: hashing the signer's public key along with the message to be signed suffices.

The security properties called *message confidentiality* and *existential signature unforgeability* respectively rely (in the random oracle model [29]) on the intractability of the following problems introduced in [37, 35] which extend ideas from [154, 189].

Theorem 8.4 *Assume that an adversary $\mathcal{A}$ has non-negligible advantage ϵ in breaking the IND-SC-CCA security of the scheme when running in time τ, asking q_{h_i} queries to random oracles h_i (for $i = 1, 2, 3$), q_{se} signature/encryption queries and q_{dv} decryption/verification queries. Then there exists a PPT algorithm $\mathcal{B}$ to solve the p-Diffie-Hellman Inversion problem for $p = q_{h_1}$ with advantage*

$$\epsilon' > \epsilon - \frac{q_{dv}}{2^k} - \frac{q_{h_3}}{2^{n+\ell}}$$

when running in time $\tau' < \tau + O(q_{h_1}^2 \tau_m) + 2q_{h_2}\tau_p$ where τ_m is the maximal cost of a scalar multiplication in $\mathbb{G}_1$ and $\mathbb{G}_2$, τ_p being the time for a bilinear map evaluation.

Proof. We actually show how $\mathcal{B}$ can use $\mathcal{A}$ as a subroutine to solve a random instance $(P, Q, xQ, x^2Q, \ldots, x^pQ)$ of the $(p+1)$-exponent problem problem in $(\mathbb{G}_1, \mathbb{G}_2)$ which is equivalent to the p-DHI problem and consists in computing $x^{p+1}P \in \mathbb{G}_1$. We can assume w.l.o.g. that $q_{se} = q_{h_1} - 1$ since, otherwise, $\mathcal{B}$ can issue dummy signcryption queries for itself. In a preparation phase, $\mathcal{B}$ uses its input to compute generators $H \in \mathbb{G}_2$ and $G = \psi(H) \in \mathbb{G}_1$ together with a public key $X = xH \in \mathbb{G}_2$

such that it knows $q_{se} = p - 1$ pairs $(w_i, \frac{1}{w_i+x}G)$ for $w_i \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ as in the proof technique of [37]. To do so, $\mathcal{B}$ picks $w_1, \ldots, w_{p-1} \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$, expands the polynomial $f(z) = \prod_{i=0}^{p-1}(z + w_i) = \sum_{i=0}^{p-1} c_i z^i$. A generator $H \in \mathbb{G}_2$ and the public key X are then obtained as

$$H = \sum_{i=0}^{p-1} c_i(x^i Q) = f(x)Q \quad \text{and} \quad X = \sum_{i=1}^{p} c_{i-1}(x^i Q) = xf(x)Q = xH$$

(as in the proof of lemma 1 in [37]). As in [37], pairs $(w_i, \frac{1}{w_i+x}G)$ are obtained by expanding $f_i(z) = f(z)/(z+w_i) = \sum_{i=0}^{p-2} d_i z^i$ and computing

$$G_i = \sum_{j=0}^{p-2} d_j \psi(x^j Q) = f_i(x)P = \frac{f(x)}{z + w_i}P = \frac{1}{z + w_i}G$$

for $i = 1, \ldots, p - 1$.

The adversary $\mathcal{A}$ is then initialized with the generators $H \in \mathbb{G}_2$ and $G = \psi(H) \in \mathbb{G}_1$ and on the public key $X \in \mathbb{G}_2$. She starts probing the oracles she is given access to. These oracles are simulated by $\mathcal{B}$ as explained below. The queries to oracle h_2 need to be simulated using two lists L_2, L_2' that are initially empty.

- h'-queries on an input $(\alpha_i, m_i) \in \mathbb{Z}_q^* \times \{0,1\}^*$: $\mathcal{B}$ first checks if $X = \alpha_i H$. In this case, we are done and $\mathcal{B}$ can easily compute $x^{p+1}P$. Otherwise, it responds with a random bit $b_{m_i} \stackrel{R}{\leftarrow} \{0,1\}$.

- h_1-queries: these queries are indexed by a counter t that is initially set to 1. When a triple $d||m||X \in \{0,1\} \times \{0,1\}^n \times \mathbb{G}_2$ is submitted in a h_1-query for the first time, $\mathcal{B}$ checks whether d equals the bit b_m (which is set at the first time the message m is submitted in a $h_1(.)$ query). If $d = b_m$, $\mathcal{B}$ returns w_t and increments t (in such a way that $\mathcal{B}$ is able to create a valid signature on m). Otherwise, $\mathcal{B}$ returns a random $c \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ stores (d, m, X, c) in L_1.

- h_2-queries on input $Y_{1,i}||Y_{2,i}||Y_{3,i} \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1$: $\mathcal{B}$ checks if the 4-uple $(H, Y_{1,i}, Y_{2,i}, Y_{3,i})$ is a valid co-Diffie-Hellman tuple (in our notation, we shall write $Y_{3,i} = \text{co-DH}_H(Y_{1,i}, Y_{2,i})$) by verifying if

$$e(Y_{1,i}, Y_{2,i}) = e(Y_{3,i}, H).$$

If it is, $\mathcal{B}$ checks if L'_2 contains an entry of the form $(Y_{1,i}, Y_{2,i}, ., \tau_i)$ for some $\tau_i \in \{0,1\}^{k+1}$. In this case, τ_i is returned and an entry $(Y_{1,i}, Y_{2,i}, Y_{3,i}, \tau_i, 1)$ is added in L_2. If no entry of the shape $(Y_{1,i}, Y_{2,i}, ., \tau_i)$ is in L'_2, $\mathcal{B}$ returns a string $\tau_i \xleftarrow{R} \{0,1\}^{k+1}$ and inserts $(Y_{1,i}, Y_{2,i}, Y_{3,i}, \tau_i, 1)$ in L_2. If $(H, Y_{1,i}, Y_{2,i}, Y_{3,i})$ is not a co-DH tuple, the entry $(Y_{1,i}, Y_{2,i}, Y_{3,i}, \tau_i, 0)$ is added in L_2. At most $2q_{h_2}$ pairings must be computed overall.

- h_3-queries: are answered with random elements from $\{0,1\}^{n+\ell_2}$.

- Signature/encryption queries on a plaintext m, for an arbitrary receiver's key Y: we assume that m was previously submitted in a h_1-query and that the message-dependent bit b_m was previously defined. Since $h_1(b_m||m||X)$ was (or will be) defined to be w_j for some $j \in \{1, \ldots, t\}$, $\mathcal{B}$ knows that the previously computed $G_j = (1/(w_j + x))G$ appears as a valid signature on m from $\mathcal{A}$'s view. So, it computes $C_1 = \gamma G_j \in \mathbb{G}_1$ for some $\gamma \xleftarrow{R} \mathbb{Z}_q^*$, obtains $\kappa = h_3(\gamma) \in \{0,1\}^{n+\ell_2}$ through h_3-simulation and computes $C_3 = (m||X) \oplus \kappa \in \{0,1\}^{n+\ell_2}$. It then checks if L_2 contains an entry $(C_1, Y, Y_3, \tau, 1)$ (indicating that $Y_3 = \text{co-DH}_H(C_1, Y)$). If this entry exists, $\mathcal{B}$ returns $C = \langle C_1, C_2, C_3 \rangle$ with $C_2 = (\gamma||b_m) \oplus \tau \in \{0,1\}^{k+1}$. Otherwise it returns $C = \langle C_1, C_2, C_3 \rangle$ for a random $C_2 \xleftarrow{R} \{0,1\}^{k+1}$ and inserts $\big(C_1, Y, ., (\gamma||b_m) \oplus C_2\big)$ in the special list L'_2.

- Decryption/verification queries: when $\mathcal{A}$ submits a ciphertext $C = \langle C_1, C_2, C_3 \rangle$, $\mathcal{B}$ checks whether L_2 contains the unique entry $(C_1, X, Y, \tau, 1)$ for some $Y \in \mathbb{G}_1$ and $\tau \in \{0,1\}^{k+1}$ (indicating that $Y = \text{co-DH}_H(C_1, X)$):
  - if it does, $\mathcal{B}$ obtains $(\gamma||b_m) = C_2 \oplus \tau \in \{0,1\}^{k+1}$, $\kappa = h_3(\gamma)$ (via simulation of h_3) and finally $(m||X_S) = C_3 \oplus \kappa \in \{0,1\}^{n+\ell_2}$ (C is also rejected if X_S is not a $\mathbb{G}_2$ element). Finally, $\mathcal{B}$ extracts $\sigma = \gamma^{-1} C_1$ and returns the plaintext $m \in \{0,1\}^n$ and the associated signature σ together with the sender's public key $X_S \in \mathbb{G}_2$ if the verification equation (1) holds.
  - if it does not, $\mathcal{B}$ picks a random $\tau \xleftarrow{R} \{0,1\}^{k+1}$, inserts $(C_1, X, ., \tau)$ into the special list L'_2 (so that a subsequent

h_2-query on $(C_1, X, \text{co-DH}_H(C_1, X))$ will receive τ as an answer), before finishing the job with the randomly chosen τ: it computes $(\gamma||b_m) = C_2 \oplus \tau \in \{0,1\}^{k+1}$, $\kappa = h_3(\gamma)$, and so on. The extracted signature $\sigma = \gamma^{-1}C_1$ is checked as above.

After the find stage, $\mathcal{A}$ outputs messages m_0, m_1 and a sender's private key $x_S \in \mathbb{Z}_q^*$. At this moment, $\mathcal{B}$ chooses a random $a \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and computes $C_1^* = (x+a)G \in \mathbb{G}_1$ as $C_1^* = \psi(X) + aG$. It also expands the polynomial $f'(z) = f(z)(z+a) = \sum_{j=0}^{p} f_j z^j \in \mathbb{Z}_q[z]$ and returns the challenge $C^* = \langle C_1^*, C_2^*, C_3^* \rangle$, where $b^* \stackrel{R}{\leftarrow} \{0,1\}$, $C_2^* \stackrel{R}{\leftarrow} \{0,1\}^{k+1}$ and $C_3^* = (m_d||x_S H) \oplus \kappa$ for a random bit $d \stackrel{R}{\leftarrow} \{0,1\}$ and a random $\kappa \stackrel{R}{\leftarrow} \{0,1\}^{n+\ell_2}$. In the extremely unlikely case (its probability is much smaller than $q_{dv}/2^k$) that C^* was submitted in a Decrypt/Verify query before the challenge phase, $\mathcal{B}$ aborts.

Clearly, if κ does not hit the output of an h_3-query (the probability for this to occur is at most $q_{h_3}/2^{n+\ell_2}$), the adversary's view is independent from the bit d unless the hash value $h_2\big(C_1^*||X||\text{co-DH}_H(C_1^*, X)\big)$ is asked during the simulation. Such an event, that we call AskH_2, is easily detected by the h_2 simulator and is very likely to happen: as in the proofs of theorems 3.1 and 3.2, one can easily show that in a real attack, $\Pr[\mathsf{AskH}_2]$ is at least ϵ if $\mathcal{A}$'s advantage in definition 8.2 is ϵ. Furthermore, as long as $\mathcal{A}$ is provided with a consistent view, $\Pr[\mathsf{AskH}_2]$ is the same in the simulation as in the real world.

Queries made by $\mathcal{A}$ in the second stage are handled as above and, as already argued, the h_2-simulator must detect the AskH_2 event with high probability. At this moment, $\mathcal{B}$ obtains $Z = \text{co-DH}_H(C_1^*, X) = x(x+a)G = xf(x)(x+a)P$. Since we have $f(z)(z+a)z = zf'(z) = \sum_{j=0}^{p} f_j z^{j+1}$ and, since $Z = \sum_{j=0}^{p} f_j \psi(x^{j+1}Q)$, $\mathcal{B}$ can compute

$$(x^{p+1})P = \frac{1}{f_p}\big[Z - \sum_{j=0}^{p-1} f_j \psi(x^{j+1}Q)\big] \in \mathbb{G}_1$$

which is the solution to the $(p+1)$-exponent problem. At that moment, we are done since the latter is known to be equivalent to the p-Diffie-Hellman Inversion problem (as explained in [154] and [35]).

From a computational point of view, $\mathcal{B}$'s running time is dominated by $p+2$ multi-exponentiations with q elements that reach an overall cost

of $O(p^2)$ exponentiations. Computing $f(z)$ also involves a cost in $O(p^2)$ while computing each $f_i(z)$ also implies $O(p)$ modular multiplications just like the computation of the product $f(z)(z+a)$. When handling h_2 queries, $\mathcal{B}$ also has to compute $2q_{h_2}$ pairings.

The bound on $\mathcal{B}$'s advantage derives from the fact that it never provides $\mathcal{A}$ with an incoherent view. The simulation only fails at the challenge phase if the selected κ hits the output of a previous h_3-query or if the constructed challenge was previously submitted to the Decrypt/Verify oracle. □

We observe that the reduction is tight in terms of probabilities in that the probability of solving the computational problem is close to the adversary's advantage. Unfortunately, the reduction is not particularly efficient as its cost is bounded by an expression containing a term which is quadratic in the number of adversarial random oracle queries. With the bound $q_{h_1} < 2^{60}$ usually considered in the literature, our reduction does not lead to a strong concrete security.

Nevertheless, we do not believe this lack of efficiency in the reduction to be a serious problem. After all, a similar remark can be made on the security reduction given by Boneh-Boyen [37] for the underlying signature scheme in the random oracle model.

Theorem 8.5 *If an ESUF-SC-CMA adversary $\mathcal{F}$ has a non-negligible advantage ϵ in the game of definition 8.6 when running in time τ, making q_{h_i} queries to oracles h_i ($i = 1, 2, 3$), q_{dv} Decrypt/Verify queries and q_{se} Sign/Encrypt queries, then there exists an algorithm $\mathcal{B}$ that solves the p-strong Diffie-Hellman problem for $p = q_{h_1}$ with probability*

$$\epsilon' > \frac{\epsilon}{2} - \frac{1}{2^k} - \frac{1}{2^{n+\ell}}$$

within time $\tau' < \tau + O(q_{h_1}^2 \tau_m) + 2q_{h_2}\tau_p$ where τ_m is the maximal cost of a scalar multiplication in $\mathbb{G}_1$ and $\mathbb{G}_2$, τ_p being the time for a bilinear map evaluation.

Proof. We build a simulator $\mathcal{B}$ that behaves almost exactly as in the previous proof. The generator $G \in \mathbb{G}_1$, that is given to the forger $\mathcal{F}$ as a part of the output of the common key generation algorithm, is generated as in the proof of theorem 1 so that the simulator $\mathcal{B}$ knows $p-1$ pairs $(w_i, \frac{1}{w_i+x}G)$ (where $x \in \mathbb{Z}_q^*$ is the unknown element that implicitly

defines its input $P, Q, xQ, x^2Q, \ldots, x^pQ$). By doing so, $\mathcal{B}$ is always able to answer signature/encryption queries that are handled, as all other oracle queries, exactly as in the proof of theorem 8.4.

Eventually, the forger $\mathcal{F}$ halts and outputs a forged signature embedded into a ciphertext $C^* = \langle C_1^*, C_2^*, C_3^* \rangle$ and an arbitrary recipient's key pair ($x_R^*, Y_R^* = x_R^* H$) that allows $\mathcal{B}$ recovering the fake message-signature pair ($m^*, \sigma^* = \frac{1}{h_1(b^*||m^*||X)+x} G$) embedded into C^*. With a probability 1/2, b^* differs from the message-dependent bit b_{m^*} (that indicates how a message the message m^* should be signed with the private key corresponding to X in the underlying signature scheme and that is independent from $\mathcal{F}$'s view) and $\mathcal{B}$ can extract a solution to the p-Strong Diffie-Hellman problem as follows: if $\mathcal{F}$ is successful, $\mathcal{B}$ recovers a valid message-signature pair for the sender's public key X by computing $(\gamma^*||b^*) = C_2 \oplus h_2(C_1^*||Y_R^*||x_R^* C_1^*)$, $(m^*||X) = C_3^* \oplus h_3(\gamma^*)$ and $\sigma^* = {\gamma^*}^{-1} C_1^*$. A p-Strong Diffie-Hellman pair $\langle h_1^*, G^* \rangle$ can then be extracted by expanding $f(z)/(z + h_1^*)$ into

$$\frac{\gamma_{-1}}{z + h_1^*} + \sum_{i=0}^{p-2} \gamma_i z^i,$$

where $h_1^* = h_1(b^*, m^*)$, and computing $G^* = \frac{1}{\gamma_{-1}} \left[\sigma^* - \sum_{i=0}^{p-2} \gamma_i \psi(x^i Q)\right]$.

A lower bound on the simulator's probability to obtain it is thus one half of the advantage of the simulator of the previous proof decreased by the (negligible) probability for $\mathcal{F}$ to produce a valid encryption of the fake signature without asking the appropriate h_2 and h_3 values during the simulation (in the latter case, $\mathcal{B}$ is unable to extract a p-Strong Diffie-Hellman pair). □

The next theorem demonstrates the ciphertext anonymity property under the p-DHI assumption. Its proof is somewhat similar to the one of theorem 8.4.

Theorem 8.6 *Assume that an adversary $\mathcal{A}$ has non-negligible advantage ϵ in breaking the SC-INDK-CCA security of the scheme when running in time τ, asking q_{h_i} queries to random oracles h_i (for $i = 1, 2, 3$), q_{se} signature/encryption queries and q_{dv} decryption/verification queries. Then there exists a PPT algorithm $\mathcal{B}$ to solve the p-Diffie-Hellman Inversion*

problem for $p = q_{h_1}$ with advantage

$$\epsilon' > \frac{\epsilon}{2} - \frac{q_{dv}}{2^k} - \frac{q_{h_3}}{2^{n+\ell}}$$

when running in time $\tau' < \tau + O(q_{h_1}^2 \tau_e) + 2q_{h_2}\tau_p$ where τ_e and τ_p denote the same quantity as in theorem 8.4.

PROOF. Algorithm $\mathcal{B}$ takes as input $(P, Q, xQ, x^2Q, \ldots, x^pQ)$ and attempts to compute $x^{p+1}P \in \mathbb{G}_1$. We assume w.l.o.g. that $q_{se} = q_{h_1} - 1$. As in the proof of theorem 8.4, $\mathcal{B}$ prepares generators $H = f(x)P \in \mathbb{G}_2$ (for some polynomial $f(z) = \prod_{i=1}^{p-1}(z + w_i) \in \mathbb{F}_q[z]$) and $G = \psi(H) \in \mathbb{G}_1$ together with a public key $X = xH \in \mathbb{G}_2$ such that it knows $q_{se} = p - 1$ pairs $(w_i, G_i = \frac{1}{w_i + x}G)$ for $w_i \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$. The simulator also chooses a random private key $y \in \mathbb{G}_2$ and sets out a second public key $Y = yH \in \mathbb{G}_2$.

The distinguisher $\mathcal{D}$ is then initialized with generators $H \in \mathbb{G}_2$, $G = \psi(H) \in \mathbb{G}_1$ and public keys $X, Y \in \mathbb{G}_2$. She starts issuing queries in the scenario of definition 8.4. The Sign/Encrypt and Decrypt/Verify oracles related to the public key X are simulated exactly as in the proof of theorem 8.4 whereas queries pertaining to Y are tackled with in accordance with the specification of Sign/Encrypt and Decrypt/Verify using the private key $y \in \mathbb{Z}_q^*$ that is known to $\mathcal{B}$.

At the challenge phase, $\mathcal{A}$ outputs a message $m \in \{0,1\}^n$ and two private keys $x_{S,0}, x_{S,1} \in \mathbb{Z}_q^*$. At this point, $\mathcal{B}$ chooses a random $a \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and computes $C_1^* = (x + a)G = \psi(X) + aG$. It also expands the polynomial $f'(z) = f(z)(z + a) = \sum_{j=0}^{p} f_j z^j \in \mathbb{Z}_q[z]$ and returns the challenge $C^* = \langle C_1^*, C_2^*, C_3^* \rangle$, where $C_2^* \stackrel{R}{\leftarrow} \{0,1\}^{k+1}$ and $C_3^* = (m || x_{S,d}H) \oplus \kappa$ for a random bit $d \stackrel{R}{\leftarrow} \{0,1\}$ and a random $\kappa \stackrel{R}{\leftarrow} \{0,1\}^{n+\ell_2}$. In the extremely unlikely case (its probability is much smaller than $q_{dv}/2^k$) that C^* was submitted in a Decrypt/Verify query before the challenge phase, $\mathcal{B}$ aborts.

Clearly, if κ does not hit the output of an h_3-query (which occurs with probability at most $q_{h_3}/2^{n+\ell_2}$), the ciphertext C^* implicitly defines $(\gamma || b) = C_2^* \oplus \tau$ and $\kappa = h_3(\gamma)$ with either $\tau = h_2(C_1^* || X || \text{co-DH}_H(C_1^*, X))$ or $\tau = h_2(C_1^* || Y || \text{co-DH}_H(C_1^*, Y))$. Let AskH_2 denote the event that an h_2-query is made on one of the triples $(C_1^*, X, \text{co-DH}_H(C_1^*, X))$ and $(C_1^*, Y, \text{co-DH}_H(C_1^*, Y))$. In a real attack, we have

$$\epsilon + \frac{1}{4} = \Pr[\mathcal{A} \text{ wins}] \leq \Pr[\mathcal{A} \text{ wins} | \neg \mathsf{AskH}_2] \Pr[\neg \mathsf{AskH}_2] + \Pr[\mathsf{AskH}_2].$$

Clearly, $\Pr[\mathcal{A} \text{ wins}|\neg\mathsf{AskH}_2] = 1/4$ and it comes that $\Pr[\mathsf{AskH}_2] \geq \epsilon$. This shows that, provided the simulation is consistent, $\mathcal{A}$ issues an h_2-query on either $C_1^*||X||\text{co-DH}_H(C_1^*, X)$ or $C_1^*||Y||\text{co-DH}_H(C_1^*, Y)$ at some point of the game with probability at least ϵ. With probability $\epsilon/2$, an h_2-query involving $\text{co-DH}_H(C_1^*, X) = x(x+a)G$ will be issued and such an event can be detected by computing two pairings for each h_2-query.

When the relevant element

$$Z = \text{co-DH}_H(C_1^*, X) = x(x+a)G = x(x+a)f(x)P = xf'(x)P$$

is obtained, $\mathcal{B}$ can compute

$$(x^{p+1})P = \frac{1}{f_p}\big[Z - \sum_{j=0}^{p-1} f_j \psi(x^{j+1}Q)\big] \in \mathbb{G}_1$$

using the coefficients of the polynomial $f'(z) = (z+a)f(z) = \sum_{j=0}^{p} f_j z^j$.

The bound on $\mathcal{B}$'s advantage derives from the fact that it never provides $\mathcal{A}$ with an incoherent view. □

We were unfortunately unable to formally establish that the scheme satisfies the ciphertext authentication property in the sense of definition 8.7. The reason is that, using the proof technique of theorems 8.4, 8.5 and 8.6, it is difficult to set out a generator and two distinct public keys for which a series of signature/encryption queries must be simulated for both public keys without knowing the matching private keys.

Nevertheless, the scheme does seem to meet that property. Given a message-signature pair, it seems to be infeasible for anyone but the signer to encrypt it into a ciphertext intended for a specific receiver without knowing the latter's private key.

5.6. What if the ciphertext unforgeability is required?

Adapting the scheme to applications that would require the ciphertext unforgeability is straightforward: the first part of the ciphertext must be $C_1 = (1/(h_1(b_m||m||\gamma||Y_S||Y_R) + x_S))P$ where $\gamma \in \mathbb{Z}_q^*$ is encrypted in the component C_2 and Y_R is the receiver's public key.

As a consequence, γ and Y_R become part of the detached signature and the sender of a message is not only committed to the plaintext's

content but he/she is also responsible for having sent it to the owner of the key Y_R. Moreover, the ciphertext unlinkability property is also lost.

6. A scheme built on the Schnorr signature

It is very tempting to combine the Schnorr signature [191] with an ElGamal-like encryption scheme [96] into a signcryption system as both schemes use similar algebraic structures. Indeed, both cryptosystems have public keys $y = g^x$ in cyclic groups of prime order q. Schnorr signatures consist of pairs $(e, r + ex \bmod q)$, where $e = h(m, g^r)$ for a random $r \in_R \mathbb{Z}_q^*$, that satisfy $e = h(m, g^s y^{-e}) \in \mathbb{Z}_q^*$ whereas ElGamal ciphertexts are pairs $(g^r, m \times y^r) \in \mathbb{G} \times \mathbb{G}$.

A recent work [114] unsuccessfully attempted to achieve such a combination and proposed a scheme that was broken in [214]. Another construction described in [213] is not correct either as it suffers from the same weakness as the scheme of Bao and Deng [18]. More recently, Malone-Lee [138] gave a solution named SCNINR as a shorthand for "signcryption with non interactive non-repudiation" and which only provides privacy against outsider attacks. Indeed, it can be verified that anyone who happens to learn some user's private key can recover any message previously signed and encrypted by that user.

In this section, we show how to overcome this limitation and propose a variant of Malone-Lee's scheme which may be regarded as a secure optimized combination of Schnorr and ElGamal. The scheme, called SEG-signcryption, is depicted on the next figure. It makes use of the method suggested by Boyen [45] to first encrypt a part of the signature before using a hash value of it to conceal the plaintext. Unlike Malone-Lee's proposal, SEG-signcryption provably ensures privacy against insider attacks. Besides, it can be implemented with more general groups than our previous schemes which need groups equipped with bilinear maps.

6.1. The SEG signcryption scheme

Similarly to the q-DH Signcryption scheme of section 5, the present one has essentially the same complexity as an ElGamal encryption for the sender. Its decryption/verification operation is much faster than in q-DH Signcryption as it only performs simple arithmetic operations.

As mentioned above, the scheme consists in first computing a Schnorr

Common-Keygen: given a security parameter k, this algorithm outputs a k-bit prime q, a cyclic group $\mathbb{G}$ of order q, a generator $g \in \mathbb{G}$ and hash functions $H : \{0,1\}^n \times \mathbb{G} \to \mathbb{Z}_q^*$, $H' : \mathbb{Z}_q^* \to \{0,1\}^{n+\ell}$ and $G : \mathbb{G}^{\,3} \to \{0,1\}^k$ where n is the length of plaintexts and ℓ is the bitlength of elements from $\mathbb{G}$. The common key is then

$$I = \{q, \mathbb{G}, g, H, H', G, n, \ell\}.$$

Keygen: each user picks $x_u \xleftarrow{R} \mathbb{Z}_q^*$ and computes $y_u = g^{x_u} \in \mathbb{G}$ obtains a public/private key pair $(pk_u, sk_u) = (y_u, x_u)$.

Sign/Encrypt: given a message $m \in \{0,1\}^n$, the recipient's public key y_R and her private key x_S, the sender does the following:

(1) Pick $r \xleftarrow{R} \mathbb{Z}_q^*$, compute $u = g^r \in \mathbb{G}$, $e = H(m, u) \in \mathbb{Z}_q^*$ and $s = r + ex_S \bmod q$.

(2) Compute $v = s \oplus G(u, y_R, y_R^r) \in \{0,1\}^k$ and $w = (m||y_s) \oplus H'(s) \in \{0,1\}^{n+\ell}$.

The ciphertext is $C = \langle u, v, w\rangle \in \mathbb{G} \times \{0,1\}^{k+n+\ell}$

Decrypt/Verify: given $C = \langle u, v, w\rangle$,

(1) Compute $\omega = u^{x_R} \in \mathbb{G}$ and $s = v \oplus G(u, y_R, \omega) \in \{0,1\}^k$. Reject C if $s \notin \mathbb{Z}_q^*$.

(2) Compute $(m||y_S) = w \oplus H'(s) \in \{0,1\}^{n+\ell}$.

(3) Compute $e' = H(m, u) \in \mathbb{Z}_q^*$ and accept the message if

$$u = g^s y_S^{-e'}.$$

If the above relation holds, return $(m, (e', s))$.

FIGURE 8.3. The SEG-signcryption scheme

signature on the plaintext. The randomness of the signature is then reused to encrypt part of it in an ElGamal fashion and a digest of the encrypted part of the signature is in turn employed to scramble the plaintext along with the sender's private key.

SEG-signcryption allows receivers to detach plaintexts and standard Schnorr signatures from ciphertexts. It also satisfies the ciphertext unlinkability property in a slightly weaker sense than the scheme of the previous section: a sender can again deny having created a specific ciphertext (as anyone can turn a valid plaintext-signature pair into a ciphertext addressed to himself by using his private key) but external observers can still establish a link between a ciphertext and a signature

detached from it (since they share a common component u). Interestingly, a similar construction can be applied to fix the security flaw of the Bao-Deng scheme [18].

6.2. Security

The scheme enjoys a very tight reduction from the Gap Diffie-Hellman assumption [168] for the proof of message privacy. We have to mention that Malone-Lee ([138]) gave a proof of message privacy (against outsider attacks only) under the Computational Diffie-Hellman assumption for his scheme but his proof only holds if the scheme is implemented in symmetric bilinear map groups. Our scheme could be proved secure against the CDH assumption as well if it was implemented in those groups.

We nevertheless find it unnatural to use groups that might be inherently weaker whereas no bilinear map is needed by the protocol. We thus prefer leaving the security of the scheme rely on the hardness of the Gap Diffie-Hellman problem which is to compute g^{ab} given (g, g^a, g^b) together with an oracle deciding whether given quadruples (g, g^x, g^y, g^z) satisfy $z \equiv ab \bmod q$.

Theorem 8.7 *Assume that an adversary $\mathcal{A}$ has a non-negligible advantage ϵ in breaking the IND-SC-CCA security of* SEG-signcryption *when running in a time τ, asking respectively q_H, $q_{H'}$ and q_G queries to random oracles H, H' and G, q_{se} signature/encryption queries and q_{dv} decryption/verification queries. Then there exists a PPT algorithm $\mathcal{B}$ to solve the Gap Diffie-Hellman problem with an advantage*

$$\epsilon' > \epsilon - \frac{q_{H'}}{2^{n+\ell-1}}$$

when running in a time $\tau' < \tau + O((q_{se} + q_{dv})\tau_{exp})$ where τ_{exp} is the cost of a group exponentiation.

Proof. Algorithm $\mathcal{B}$ takes as input $(g, g^a, g^b, \mathcal{O}_{DDH})$, where $\mathcal{O}_{DDH}(.)$ denotes an oracle distinguishing Diffie-Hellman tuples (g, g^a, g^b, g^{ab}) from random tuples (g, g^a, g^b, g^c) with probability 1, and attempts to find g^{ab}.

The adversary $\mathcal{A}$ is initialized with $y_u = g^a \in \mathbb{G}$ as a challenge public key. She starts probing the oracles she is given access to and that are simulated by $\mathcal{B}$ as explained below. The queries to the random oracle G need to be simulated using two lists L_G, L'_G that are initially empty.

- H-queries and H'-queries: are answered with random elements respectively sampled from $\mathbb{Z}_q^*$ and $\{0,1\}^{n+\ell}$.

- G-queries on triples $(y_{1,i}, y_{2,i}, y_{3,i}) \in \mathbb{G}^3$: $\mathcal{B}$ checks if the 4-uple $(g, y_{1,i}, y_{2,i}, y_{3,i})$ is a valid Diffie-Hellman tuple (in our notation, we write $y_{3,i} = \mathrm{DH}_g(y_{1,i}, y_{2,i})$) using its DDH oracle $\mathcal{O}_{DDH}(.)$. If it is, $\mathcal{B}$ checks if L'_G contains an entry of the form $(y_{1,i}, y_{2,i}, ., h_i)$ for some $h_i \in \{0,1\}^k$. In this case, h_i is returned and a record $(y_{1,i}, y_{2,i}, y_{3,i}, h_i, 1)$ is added in L_G. If no entry of the form $(y_{1,i}, y_{2,i}, ., h_i)$ is in L'_G, $\mathcal{B}$ returns a string $h_i \xleftarrow{R} \{0,1\}^k$ and inserts $(y_{1,i}, y_{2,i}, y_{3,i}, h_i, 1)$ in L_G. If $(g, y_{1,i}, y_{2,i}, y_{3,i})$ is not a Diffie-Hellman tuple, the record $(y_{1,i}, y_{2,i}, y_{3,i}, h_i, 0)$ is added in L_G. At most q_G calls to oracle $\mathcal{O}_{DDH}(.)$ are needed overall.

- Signature/encryption queries on a message m, for a receiver's key y: $\mathcal{B}$ chooses $e, s \xleftarrow{R} \mathbb{Z}_q^*$, computes $u = g^s y_u^{-e} \in G$ and defines $H(m,u)$ to be e (it aborts if oracle H is already defined at that point). It then checks if L_G contains an entry $(u, y, y', h, 1)$ (indicating that $y' = \mathrm{DH}_g(u, y_u)$). If this entry exists, $\mathcal{B}$ returns $C = \langle u, v, w\rangle$ with $v = s \oplus h \in \{0,1\}^k$ and $w = (m||y_u) \oplus H'(s)$ (where $H'(s)$ is obtained via simulation of H'). Otherwise it returns $C = \langle u, v, w\rangle$ for a random $v \xleftarrow{R} \{0,1\}^k$ and inserts $(u, y, ., s \oplus v)$ in the special list L'_G.

- Decryption/verification queries: when $\mathcal{A}$ submits a ciphertext $C = \langle u, v, w\rangle$, $\mathcal{B}$ checks whether L_G contains the unique entry $(u, y_u, y', h, 1)$ for some $y' \in \mathbb{G}$ and $h \in \{0,1\}^k$ (indicating that $y' = \mathrm{DH}_g(u, y_u)$):
  - if it does, $\mathcal{B}$ obtains $s = v \oplus h \in \{0,1\}^k$, $\kappa = H'(s)$ (via simulation of H') and finally $(m||y_S) = w \oplus \kappa \in \{0,1\}^{n+\ell}$ (C is rejected if $y_S \notin \mathbb{G}$). Finally, $\mathcal{B}$ obtains $e = H(m,u)$ and returns the plaintext $m \in \{0,1\}^n$ and the signature (e,s) together with the sender's public key $y_S \in \mathbb{G}$ if $u = g^s y_S^{-e}$.
  - if it does not, $\mathcal{B}$ picks a random $h \xleftarrow{R} \{0,1\}^k$, inserts $(u, y_u, ., h)$ into the special list L'_G (so that a subsequent G-query on $(u, y_u, \mathrm{DH}_g(u, y_u))$ will receive h as an answer),

before finishing the job as explained above with the randomly chosen h.

Once $\mathcal{A}$ decides that the first stage is over, she outputs messages m_0, m_1 and a sender's private key $x_S \in \mathbb{Z}_q^*$. The simulator $\mathcal{B}$ then responds with a challenge ciphertext $\langle g^b, v^*, w^* \rangle$ for random strings $v^* \stackrel{R}{\leftarrow} \{0,1\}^k$ and $w^* \stackrel{R}{\leftarrow} \{0,1\}^{n+\ell}$.

Clearly, if $(m_0||g^{x_S}) \oplus w^*$ and $(m_1||g^{x_S}) \oplus w^*$ do not hit the output of an H'-query (the probability for this to occur is at most $2q_{H'}/2^{n+\ell}$), the adversary is very likely to ask for the hash value $G(g^b, g^a, g^{ab})$ during the simulation. Such an event, that we call AskG, is easily detected when answering G-queries. With standard arguments, one can show that in a real attack, Pr[AskG] is at least ϵ if $\mathcal{A}$'s advantage in definition 8.2 is ϵ. Besides, as long as $\mathcal{A}$ is provided with a consistent view, Pr[AskG] is the same in the simulation as in the real world.

Post-challenge adversarial queries are handled as above and, as already argued, the G-simulator must detect the AskG event with overwhelming probability. At this moment, $\mathcal{B}$ obtains the group element g^{ab} that it was looking for.

From a computational point of view, $\mathcal{B}$'s running time is dominated by $q_{se} + q_{dv}$ multi-exponentiations with 2 elements.

The bound on $\mathcal{B}$'s advantage derives from the fact that the simulation only fails if the random string w^* chosen in the challenge phase is such that $(m_b||g^{x_S}) \oplus w^*$ collides with the output of a H'-query. □

The proof of signature unforgeability can be obtained using the forking lemma [182, 183] in the same way as for the Schnorr signature. Indeed, definition 8.6 assumes that the adversary's outcome is a ciphertext and a receiver's private key that allows extracting a fake signature on the plaintext. Replaying the adversary according to the forking technique thus enables us to obtain two suitably related forged signatures and solve a discrete logarithm instance which automatically leads to the solution of a Gap Diffie-Hellman problem. All oracles are simulated exactly in the same way as in the proof of theorem 8.7. A decision Diffie-Hellman oracle is necessary to maintain the consistency of the simulation and that is why the signature unforgeability (loosely) relies on the Gap Diffie-Hellman assumption instead of the discrete logarithm problem as in the Schnorr signature.

The ciphertext anonymity property can be proved in the same way as the message confidentiality under the Gap Diffie-Hellman assumption. Unlike what happens with the q-DH-signcryption scheme, it is very easy to formally establish the ciphertext authentication property in the sense of definition 8.7.

Theorem 8.8 *Assume that an adversary $\mathcal{A}$ has a non-negligible advantage ϵ in breaking the AUTH-SC-CMA property of SEG-signcryption when running in a time τ, asking respectively q_H, $q_{H'}$ and q_G queries to random oracles H, H' and G, q_{se} signature/encryption queries and q_{dv} decryption/verification queries. Then there exists a PPT algorithm $\mathcal{B}$ to solve the Gap Diffie-Hellman problem with an advantage*

$$\epsilon' > \epsilon - \frac{1}{2^k}$$

when running in a time $\tau' < \tau + O((q_{se} + q_{dv})\tau_{exp})$ where τ_{exp} is the cost of a group exponentiation.

PROOF. Let $(g, g^a, g^b, \mathcal{O}_{DDH})$ denote algorithm $\mathcal{B}$'s input. The adversary is initialized with the public keys $y_S = g^a$ and $y_R = g^b$ for which the private keys are implicitly set to $x_S = a$ and $x_R = b$. She attempts to produce a non-trivially obtained ciphertext from the sender of public key y_S to the receiver of public key y_R. Throughout the simulation, she is provided with an oracle access to the private key operations (namely signature/encryption and decryption/verification oracles) pertaining to both pk_S and pk_R. All random oracles as well as oracles Sign/Encrypt$(., x_S, .)$, Decrypt/Verify$(., x_S)$ Sign/Encrypt$(., x_R, .)$ and Decrypt/Verify$(., x_R)$ are simulated exactly as in the proof of theorem 8.7. When handling queries to the Sign/Encrypt$(., x_S, .)$ oracle, the simulator $\mathcal{B}$ additionally stores in a dedicated list L_{sig} the generated triples $(u, e, s) \in \mathbb{G} \times \mathbb{Z}_q^* \times \mathbb{Z}_q^*$, where $u = g^s (y_S)^{-e}$, which are simulated Schnorr signatures properly encrypted in returned ciphertexts.

Eventually, $\mathcal{A}$ halts and outputs a ciphertext $\langle u^*, v^*, w^* \rangle$. Under the (unknown) private key $sk_R = b$, the latter is assumed to decrypt into a valid messsage-signature pair for the public key $y_S = g^a$.

Assuming that $\mathcal{A}$ is unable to non-trivially produce a ciphertext containing a valid message-signature pair without knowing the private key $x_S = a$ (the existential signature unforgeability property precludes

this possibility), the ciphertext $\langle u^*, v^*, w^* \rangle$ necessarily results from a re-encryption of a Schnorr signature that $\mathcal{A}$ extracted from a ciphertext returned by the $\mathsf{Sign/Encrypt}(., x_S, .)$ oracle. Hence, the special list L_{sig} contains at least a triple (u^*, s, e) for which $u^* = g^s (y_S)^{-e}$. In other words, the simulator $\mathcal{B}$ necessarily knows at least one of the q distinct representations of u^* in the base (g, y_S).

Moreover, $\mathcal{A}$ has to know the hash value $G(u^*, y_R, u^{*x_R})$ to produce a proper encryption of the signature (u^*, s, e). With probability $1 - 1/2^k$, she thus queried the random oracle $G(.)$ on the triple (u^*, g^b, u^{*b}) at some point of the simulation. For such a query, the simulator $\mathcal{B}$ that must have stored in list L_G (that contains information pertaining to G-queries) an entry $(u^*, g^b, z^*, h^*, 1)$ indicating that $G(u^*, g^b, z^*) = h^*$ and $z^* = u^{*b} = (g^b)^s (g^{ab})^{-e}$. The searched element g^{ab} can finally be extracted as $((g^b)^s / z^*)^{1/e}$.

□

7. Conclusion

This chapter described a security model for signcryption schemes with ciphertext anonymity. We presented three constructions based on discrete logarithm related assumptions, each with their own advantages and disadvantages. Two of them are constructed on pairing-based signature schemes while the third one is obtained from a careful combination of the Schnorr signature with an ElGamal like encryption scheme. The first construction enjoys a near-optimal concrete security but does not offer noticeable computational savings w.r.t. a sequential composition of signature and encryption. The second one features less efficient security reductions under stronger assumptions but it is much more efficient. It additionally enables short signatures to be detached from ciphertexts, which appears to be a unique particularity for a signcryption scheme. The last one can be instantiated with more general groups and is based on the Schnorr signature. It happens to be the most efficient of our proposals upon decryption/verification and its property of ciphertext authentication (which is not relevant for our first scheme) can be formally established. It unfortunately suffers from the disadvantage of a loose reduction under the non-standard Gap Diffie-Hellman assumption in its proof of non-repudiation.

Our three constructions have a great online efficiency for senders. The last two ones ensure the non-repudiation property only for signatures embedded in ciphertexts rather than for entire ciphertexts. We indeed believe that the property of signature-unforgeability is sufficient for all known practical applications. What is more, it is compatible with the definition of ciphertext unlinkability and allows our second scheme to provide short detachable signatures.

Conclusions and open problems

This thesis contributed to various uses of bilinear maps in cryptography. A large part was dedicated to the area of identity based cryptography and the related concept of certificateless cryptography.

The second part of the thesis suggested slight efficiency improvements for the famous identity-based encryption scheme of Boneh and Franklin. Chapter 4 also showed that it was not trivial to build a certificateless encryption scheme by combining an IBE system with a traditional public key encryption scheme. The same chapter described a random oracle-using generic construction to achieve this goal and suggested a new efficient scheme.

The third part describes a new identity based signature where the verification algorithm happens to be the fastest one among all known pairing-based IBS schemes. We showed in chapter 7 that our new IBS construction can be very efficiently turned into a protocol ensuring both privacy and authentication. The security of the new identity-based signcryption scheme rests on stronger computational assumptions than previous ones. However, the new scheme significantly outperforms those previous proposals. Chapter 6 gave an example of an identity-based undeniable signature scheme. This construction is not very efficient but, as far as we know, it is the only known one to be supported by security proofs.

The last part of the thesis showed several constructions of traditional (i.e. non-identity-based) cryptosystems providing both confidentiality and authentication and with security proofs under Diffie-Hellman related assumptions. Our first two constructions rely on bilinear maps. The second one has the unique property that short signatures can be detached from ciphertexts. Our last scheme is constructed on the Schnorr signature and does not use bilinear maps. It can thus be instantiated with more general groups where the discrete logarithm assumption holds.

Although pairings have been used as building blocks in a tremendous number of cryptographic protocols ([19]), we believe that the topic is far from being exhausted. Within the area of identity-based cryptography, several problems have not been really investigated.

One of these issues is the feasibility of a certificateless encryption scheme admitting a security proof in the strongest model ([6]) without random oracles. The strength of the certificateless security model renders really challenging the task of finding a secure CLE scheme without random oracles.

Regarding certificateless cryptography, it is surprising to note that essentially nothing has been carried out about certificateless signatures (CLS). A scheme was suggested by Al-Riyami and Paterson ([6]) without a security proof nor a formal security model. Yum and Lee ([223]) described a generic construction composing identity-based signatures with traditional digital signatures but they used a security model that we deem seriously undermined w.r.t. what it should be. We thus plan to consider a strong model inspired from its analogue ([6]) for the encryption case and we believe that a variant of the pairing-based CLS scheme ([6]) can be proved to fit our model. This would be an interesting theoretical result as the inherent key escrow property is precisely the reason why people may be reluctant to use identity-based signatures.

Other open problems would be to find an IBE scheme in the standard model with shorter public parameters than Waters' proposal ([217]). We also believe that the proof techniques of Waters can give rise to an identity-based signature in the standard model. We finally mention the problem of finding a hierarchical IBE system with security reductions that do not degrade themselves with the depth of the hierarchy. Previous constructions ([99, 38]) indeed feature security reductions that exponentially degrade with this parameter.

Bibliography

[1] M. Abdalla, J.-H. An, M. Bellare, and C. Namprempre. From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In *Eurocrypt'02*, volume 2332 of *LNCS*, pages 418–433. Springer, 2002.

[2] M. Abdalla, M. Bellare, and P. Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In *CT-RSA'01*, volume 2020 of *LNCS*, pages 143–158. Springer, 2001.

[3] M. Abe and T. Okamoto. A signature scheme with message recovery as secure as discrete logarithm. In *Asiacrypt'99*, volume 1716 of *LNCS*, pages 378–389. Springer, 1999.

[4] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *STOC'97*, pages 284–293. ACM Press, 1997.

[5] S. S. Al-Riyami. Cryptographic schemes based on elliptic curve pairings. PhD thesis, University of London, 2004.

[6] S. S. Al-Riyami and K. Paterson. Certificateless public key cryptography. In *Asiacrypt'03*, volume 2894 of *LNCS*, pages 452–473. Springer, 2003.

[7] S. S. Al-Riyami and K. Paterson. CBE from CL-PKE: A generic construction and efficient schemes. In *PKC'05*, volume 3386 of *LNCS*, pages 398–415. Springer, 2005.

[8] J. H An. Authenticated encryption in the public-key setting: Security notions and analyses. Cryptology ePrint Archive, Report 2001/079, 2001. `http://eprint.iacr.org/2001/079`.

[9] J.-H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In *Eurocrypt'02*, volume 2332 of *LNCS*, pages 83–107. Springer, 2002.

[10] R. Anderson. Two remarks on public key cryptology. In 4^{th} *Annual Conference on Computer and Communications Security*, volume 2332. ACM, 1997.

[11] D. Augot and M. Finiasz. A public key encryption scheme based on the polynomial reconstruction problem. In *Eurocrypt'03*, volume 2656 of *LNCS*, pages 229–240. Springer, 2003.

[12] J. Baek, B. Lee, and K. Kim. Secure length-saving elgamal encryption under the computational Diffie-Hellman assumption. In *ACISP'00*, volume 1481 of *LNCS*, pages 49–58. Springer, 1998.

[13] J. Baek, R. Safavi-Naini, and W. Susilo. Certificateless public key encryption without pairing. In *ISC'05*, volume 3650 of *LNCS*, pages 134–148. Springer, 2005.

[14] J. Baek, R. Safavi-Naini, and W. Susilo. Efficient mutli-receiver identity-based encryption and its application to broadcast encryption. In *PKC'05*, volume 3386 of *LNCS*, pages 380–397. Springer, 2005.

[15] J. Baek, R. Steinfeld, and Y. Zheng. Formal proofs for the security of signcryption. In *PKC'02*, volume 2274 of *LNCS*, pages 80–98. Springer, 2002.

[16] J. Baek and Y. Zheng. Simple and efficient threshold cryptosystem from the Gap Diffie-Hellman group. `http://citeseer.nj.nec.com/567030.html`.

[17] J. Baek and Y. Zheng. Zheng and Seberry's public key encryption scheme revisited. *International Journal of Information Security (IJIS)*, 2(1):37–44, 2003.

[18] F. Bao and R. Deng. A signcryption scheme with signature directly verifiable by public key. In *PKC'98*, volume 1988 of *LNCS*, pages 55–59. Springer, 1988.

[19] P. S. L. M. Barreto. The pairing based crypto lounge. `http://planeta.terra.com.br/informatica/paulobarreto/pblounge.html`.

[20] P. S. L. M. Barreto, H. Y. Kim, B. Lynn, and M. Scott. Efficient algorithms for pairing-based cryptosystems. In *Crypto'02*, volume 2442 of *LNCS*, pages 354–368. Springer, 2002.

[21] P. S. L. M. Barreto, B. Libert, N. McCullagh, and J. J. Quisquater. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In *Asiacrypt'05*, 2005. To appear.

[22] P. S. L. M. Barreto, B. Lynn, and M. Scott. On the selection of pairing-friendly groups. In *SAC'03*, volume 3006 of *LNCS*, pages 17–25. Springer, 2003.

[23] P. S. L. M. Barreto and M. Naehrig. Pairing-friendly elliptic curves of prime order. In *SAC'05*, volume 3897 of *LNCS*, pages 319–331. Springer, 2006.

[24] P. S. L. M. Barreto and M. Scott. Compressed pairings. In *Crypto'04*, volume 3152 of *LNCS*, pages 140–156. Springer, 2004.

[25] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *Asiacrypt'01*, volume 2248 of *LNCS*, pages 566–582. Springer, 2004.

[26] M. Bellare, A. Boldyreva, and A. Palacio. An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In *Eurocrypt'04*, volume 3027 of *LNCS*, pages 171–188. Springer, 2004.

[27] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In *Crypto'98*, volume 1468 of *LNCS*, pages 26–45. Springer, 1998.

[28] M. Bellare, C. Namprempre, and G. Neven. Security proofs for identity-based identification and signature schemes. In *Eurocrypt'04*, volume 3027 of *LNCS*, pages 268–286. Springer, 2004.

[29] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *1st ACM Conference on Computer and Communications Security*, pages 62–73, Fairfax, USA, 1993. ACM Press.

[30] M. Bellare and P. Rogaway. Optimal asymmetric encryption - how to encrypt with RSA. In *Eurocrypt'94*, volume 950 of *LNCS*, pages 92–111. Springer, 1995.

[31] M. Bellare and P. Rogaway. The exact security of digital signatures - how to sign with RSA and Rabin. In *Eurocrypt'96*, volume 1070 of *LNCS*, pages 399–416. Springer, 1996.

[32] K. Bentahar, P. Farshim, J. Malone-Lee, and N. P. Smart. Generic construction of identity-based and certificateless KEMs. Cryptology ePrint Archive, Report 2005/058, 2005. `http://eprint.iacr.org/2005/058`.

[33] T. Beth. Efficient zero-knowledged identification scheme for smart cards. In *Eurocrypt'88*, volume 330 of *LNCS*, pages 77–86. Springer, 1988.

[34] A. Boldyreva. Efficient threshold signature, multisignature and blind signature schemes based on the Gap-Diffie-Hellman-group signature scheme. In *PKC'03*, volume 2567 of *LNCS*, pages 31–46. Springer, 2003.

[35] D. Boneh and X. Boyen. Efficient selective-id secure identity based encryption without random oracles. In *Eurocrypt'04*, volume 3027 of *LNCS*, pages 223–238. Springer, 2004.

[36] D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *Crypto'04*, volume 3152 of *LNCS*, pages 443–459. Springer, 2004.

[37] D. Boneh and X. Boyen. Short signatures without random oracles. In *Eurocrypt'04*, volume 3027 of *LNCS*, pages 56–73. Springer, 2004.

[38] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In *Eurocrypt'05*, volume 3494 of *LNCS*, pages 440–456. Springer, 2005.

[39] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Crypto'04*, volume 3152 of *LNCS*, pages 41–55. Springer, 2004.

[40] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In *Crypto'01*, volume 2139 of *LNCS*, pages 213–229. Springer, 2001.

[41] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Eurocrypt'03*, volume 2656 of *LNCS*, pages 416–432. Springer, 2003.

[42] D. Boneh and J. Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In *CT-RSA'05*, volume 3376 of *LNCS*, pages 87–103. Springer, 2005.

[43] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In *Asiacrypt'01*, volume 2248 of *LNCS*, pages 514–532. Springer, 2002.

[44] J. Boyar, D. Chaum, I. Damgård, and T. Pedersen. Convertible undeniable signatures. In *Crypto'90*, volume 537 of *LNCS*, pages 189–208. Springer, 1990.

[45] X. Boyen. Multipurpose identity-based signcryption: A swiss army knife for identity-based cryptography. In *Crypto'03*, volume 2729 of *LNCS*, pages 383–399. Springer, 2003.

[46] D. R. L. Brown, R. Gallant, and S. A. Vanstone. Provably secure implicit certificate schemes. In *Financial Cryptography'01*, volume 2339 of *LNCS*, pages 156–165. Springer, 2002.

[47] J. Camenisch and V. Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *Crypto'03*, volume 2729 of *LNCS*, pages 126–144. Springer, 2003.

[48] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. In *STOC'98*, pages 209–218. ACM Press, 1998.

[49] R. Canetti, S. Halevi, and J. Katz. A forward secure public key encryption scheme. In *Eurocrypt'03*, volume 2656 of *LNCS*, pages 254–271. Springer, 2003.

[50] R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In *Eurocrypt'04*, volume 3027 of *LNCS*, pages 207–222. Springer, 2004.

[51] J. C. Cha and J. H. Cheon. An identity-based signature from gap Diffie-Hellman groups. In *PKC'03*, volume 2567 of *LNCS*, pages 18–30. Springer, 2003.

[52] D. Chaum. Zero-knowledge undeniable signatures. In *Crypto'90*, volume 473 of *LNCS*, pages 458–464. Springer, 1990.

[53] D. Chaum and H. van Antwerpen. Undeniable signatures. In *Crypto'89*, volume 435 of *LNCS*, pages 212–216. Springer, 1989.

[54] D. Chaum, E. van Heijst, and B. Pfitzmann. Cryptographically strong undeniable signatures unconditionally secure for the signer. In *Crypto'91*, volume 576 of *LNCS*, pages 470–484. Springer, 1991.

[55] L. Chen and Z. Cheng. Security proof of Sakai-Kasahara's identity-based encryption scheme. Cryptology ePrint Archive, Report 2005/226, 2005. `http://eprint.iacr.org/2005/226`.

[56] L. Chen, Z. Cheng, J. Malone-Lee, and N. P. Smart. An efficient ID-KEM based on the Sakai–Kasahara key construction. Cryptology ePrint Archive, Report 2005/224, 2005. `http://eprint.iacr.org/2005/224`.

[57] L. Chen, K. Harrisson, N. P. Smart, and D. Soldera. Applications of multiple trust authorities in pairing based cryptosystems. In *Infrasec'02*, volume 2437 of *LNCS*, pages 260–275. Springer, 2002.

[58] L. Chen and J. Malone-Lee. Improved identity-based signcryption. In *PKC'05*, volume 3386 of *LNCS*, pages 362–379. Springer, 2005.

[59] Z. Cheng and R. Comley. Efficient certificateless public key encryption. Cryptology ePrint Archive, Report 2005/012, 2005. `http://eprint.iacr.org/2005/012`.

[60] J. H. Cheon, Y. Kim, and H. J. Yoon. A new ID-based signature with batch verification. Cryptology ePrint Archive, Report 2004/131, 2004. `http://eprint.iacr.org/2004/131`.

[61] B. Chevallier-Mames. An efficient cdh-based signature scheme with a tight security reduction. In *Crypto'05*, LNCS. Springer, 2005. To appear.

[62] B. Chevallier-Mames. New signature schemes with coupons and tight reductions. In *3rd Applied Conference on Cryptography and Network Security*, volume 3531 of *LNCS*, pages 513–528. Springer, 2005.

[63] S. S. M. Chow, L. C. K. Hui, S. M. Yiu, and K. P. Chow. A secure modified ID-based undeniable signature scheme. Cryptology ePrint Archive, Report 2003/262, 2003. `http://eprint.iacr.org/2003/262`.

[64] S. S. M. Chow, S. M. Yiu, L. C. K. Hui, and K. P. Chow. Efficient forward and provably secure ID-based signcryption scheme with public verifiability and public ciphertext authenticity. In *6th International Conference on Information Security and Cryptology – ICISC'2003*, volume 2971 of *LNCS*, pages 352–369. Springer, 2003.

[65] S. S. M. Chow, T. H. Yuen, L. C. K. Hui, and S. M. Yiu. Signcryption in hierarchical identity based cryptosystem. In *20th International Conference on Information Security (SEC'2005)*. IFIP TC11, 2005.

[66] C. Cocks. An identity based encryption scheme based on quadratic residues. In *8th IMA International Conference*, volume 2260 of *LNCS*, pages 360–363. Springer, 2001.

[67] D. Coppersmith. Evaluating logarithms in $gf(2^n)$. In *STOC'84*, pages 201–207. ACM Press, 1984.

[68] J.-S. Coron. On the exact security of Full Domain Hash. In *Crypto'00*, volume 1880 of *LNCS*, pages 229–235. Springer, 2000.

[69] J. S. Coron. Optimal security proofs for PSS and other signature schemes. In *Eurocrypt'02*, volume 2332 of *LNCS*, pages 272–287. Springer, 2002.

[70] J. S. Coron, H. Handschuh, M. Joye, P. Paillier, D. Pointcheval, and C. Tymen. GEM: a generic chosen-ciphertext secure encryption method. In *CT-RSA'02*, volume 2271 of *LNCS*, pages 263–276. Springer, 2002.

[71] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Crypto'98*, volume 1462 of *LNCS*, pages 13–25. Springer, 1998.

[72] R. Cramer and V. Shoup. Signature schemes based on the strong RSA assumption. In 7^{th} *ACM Conference on Computer and Communications Security*, pages 46–51. ACM Press, 1999.

[73] R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *Eurocrypt'02*, volume 2332 of *LNCS*, pages 45–64. Springer, 2002.

[74] R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal of Computing*, 33:167–226, 2003.

[75] A. Dent. A designer's guide to KEMs. In 9^{th} *IMA International Conference*, volume 2898 of *LNCS*, pages 133–151. Springer, 2003.

[76] A. Desai. New paradigms for constructing symmetric encryption schemes secure against chosen-ciphertext attack. In *Crypto'00*, volume 1880 of *LNCS*, pages 394–412. Springer, 2000.

[77] Y. Desmedt and J. J. Quisquater. Public-key systems based on the difficulty of tampering. In *Crypto'86*, volume 263 of *LNCS*, pages 111–117. Springer, 1986.

[78] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[79] Y. Dodis, M.-J. Freedman, S. Jarecki, and S. Walfish. Versatile padding schemes for joint signature and encryption. In *11^{th} ACM Conference on Computer and Communications Security*, pages 344–353, Washington, USA, 2004. ACM Press.

[80] Y. Dodis and J. Katz. Chosen-ciphertext security of multiple encryption. In *TCC'05*, volume 3378 of *LNCS*, pages 188–209. Springer, 2005.

[81] Y. Dodis, J. Katz, S. Xu, and M. Yung. Strong key-insulated signature schemes. In *PKC'03*, volume 2567 of *LNCS*, pages 130–144. Springer, 2003.

[82] Y. Dodis and A. Yampolskiy. A verifiable random function with short proofs and keys. In *PKC'05*, volume 3386 of *LNCS*, pages 416–431. Springer, 2005.

[83] D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. In *STOC'91*, pages 542–552. ACM Press, 1991.

[84] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Crypto'86*, volume 263 of *LNCS*, pages 186–194. Springer, 1986.

[85] M. Fischlin and R. Fischlin. The representation problem based on factoring. In *CT-RSA'02*, volume 2271 of *LNCS*, pages 96–113. Springer, 2002.

[86] G. Frey and H. G. Ruck. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Math. Comp.*, 62 (206):865–874, 1994.

[87] A. Fujioka, T. Okamoto, and K. Ohta. Interactive bi-proof systems and undeniable signature schemes. In *Eurocrypt'91*, volume 547 of *LNCS*, pages 243–256. Springer, 1991.

[88] E. Fujisaki and T. Okamoto. How to enhance the security of public-key encryption at minimum cost. In *PKC'99*, volume 1560 of *LNCS*, pages 53–68. Springer, 1999.

[89] E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Crypto'99*, volume 1666 of *LNCS*, pages 537–554. Springer, 1999.

[90] S. Galbraith and W. Mao. Invisibility and anonymity of undeniable and confirmer signatures. In *CT-RSA'03*, volume 2612 of *LNCS*, pages 80–97. Springer, 2003.

[91] S. Galbraith, W. Mao, and K. G. Paterson. RSA-based undeniable signatures for general moduli. In *CT-RSA'02*, volume 2271 of *LNCS*, pages 200–217. Springer, 2002.

[92] S. Galbraith and V. Rotger. Easy decision-Diffie-Hellman groups. Cryptology ePrint Archive, Report 2004/070, 2004. `http://eprint.iacr.org/2004/070/`.

[93] D. Galindo. Improved identity based encryption. In *ICALP'05*, LNCS.

[94] D. Galindo. The exact security of pairing based encryption and signature schemes. talk at INRIA Workshop on Provable Security, 2004.

[95] C. Gamage, J. Leiwo, and Y. Zheng. Encrypted message authentication by firewalls. In *PKC'99*, volume 1560 of *LNCS*, pages 69–81. Springer, 1999.

[96] T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Crypto'84*, volume 196 of *LNCS*, pages 10–18. Springer, 1985.

[97] R. Gennaro, H. Krawczyk, and T. Rabin. RSA-based undeniable signatures. In *Crypto'97*, volume 1294 of *LNCS*, pages 132–149. Springer, 1997.

[98] C. Gentry. Certificate-based encryption and the certificate revocation problem. In *Eurorypt'03*, volume 2656 of *LNCS*, pages 272–293. Springer, 2003.

[99] C. Gentry and A. Silverberg. Hierarchical ID-based cryptography. In *Asiacrypt'02*, volume 2501 of *LNCS*, pages 548–566. Springer, 2002.

[100] M. Girault. An identity-based identification scheme based on discrete logarithms modulo a composite number. In *Eurocrypt'90*, volume 473 of *LNCS*, pages 481–486. Springer, 1990.

[101] M. Girault. Self-certified public keys. In *Eurocrypt'91*, volume 547 of *LNCS*, pages 490–497. Springer, 1991.

[102] M. Girault and D. Lefranc. Server-aided verification: Theory and practice. In *Asiacrypt'05*, LNCS. Springer, 2005. To appear.

[103] E.-J. Goh and S. Jarecki. A signature scheme as secure as the Diffie-Hellman problem. In *Eurorypt'03*, volume 2656 of *LNCS*, pages 401–415. Springer, 2003.

[104] S. Goldwasser, O. Goldreich, and S. Halevi. Public-key cryptosystems from lattice reduction problems. In *Crypto'97*, volume 1294 of *LNCS*, pages 112–131. Springer, 1997.

[105] S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen message attacks. *SIAM Journal of Computing*, 17(2):281–308, 1988.

[106] L. Guillou and J-J. Quisquater. A "paradoxical" identity-based signature scheme resulting from zero-knowledge. In *Crypto'88*, volume 0403 of *LNCS*, pages 216–231. Springer, 1988.

[107] S. Halevi and P. Rogaway. A tweakable enciphering mode. In *Crypto'03*, volume 2729 of *LNCS*, pages 482–499. Springer, 2003.

[108] S. Halevi and P. Rogaway. A parallelizable enciphering mode. In *CT-RSA'04*, volume 2964 of *LNCS*, pages 292–304. Springer, 2004.

[109] S. Han, K. Y. Yeung, and J. Wang. Identity based confirmer signatures from pairings over elliptic curves. In *ACM conference on Electronic commerce*, pages 262–263. ACM Press, 2003.

[110] W. H. He and T. C. Wu. Cryptanalysis and improvement of petersen-michels signcryption scheme. *IEE Proc. - Computers and Digital Techniques*, 146(2):123–124, 1999.

[111] F. Heß. Efficient identity based signature schemes based on pairings. In *SAC'02*, volume 2595 of *LNCS*, pages 310–324. Springer, 2003.

[112] J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: A ring-based public key cryptosystem. In *ANTS'III*, volume 1423 of *LNCS*, pages 267–288. Springer, 1998.

[113] J. Horwitz and B. Lynn. Toward hierarchical identity-based encryption. In *Eurocrypt'02*, volume 2332 of *LNCS*, pages 466–481. Springer, 2002.

[114] H. F. Huang and C. C. Chang. An efficient convertible authenticated encryption scheme and its variant. In *ICICS'03*, volume 2836 of *LNCS*, pages 382–392. Springer, 2004.

[115] D. Huhnlein, M. Jacobson, and D. Weber. Towards practical non-interactive public key cryptosystems using non-maximal imaginary quadratic orders. In *SAC'00*, volume 2012 of *LNCS*, pages 275–287. Springer, 2000.

[116] M. Jakobsson, K. Sako, and R. Impagliazzo. Designated verifier proofs and their applications. In *Eurocrypt'96*, volume 1992 of *LNCS*, pages 143–154. Springer, 1996.

[117] I. R. Jeong, H. Y. Jeong, H. S. Rhee, D. H. Lee, and I. L. Jong. Provably secure encrypt-then-sign composition in hybrid signcryption. In *ICISC'02*, volume 2587 of *LNCS*, pages 16–34. Springer, 2002.

[118] A. Joux. A one round protocol for tripartite Diffie-Hellman. In *ANTS-IV*, volume 1838 of *LNCS*, pages 385–394. Springer, 2000.

[119] A. Joux and K. Nguyen. Separating decision Diffie-Hellman from Diffie-Hellman in cryptographic groups. *Journal of Cryptology*, 16(4):239–247, 2003.

[120] G. Kang and S. H. Hahn J. H. Park. A certificate-based signature scheme. In *CT-RSA'04*, volume 2964 of *LNCS*, pages 99–111. Springer, 2004.

[121] J. Katz and N. Wang. Efficiency improvements for signature schemes with tight security reductions. In *10^{th} ACM Conference on Computer and Communications Security*, pages 155–164. ACM Press, 2003.

[122] N. Koblitz and A. J. Menezes. Another look at "provable security". Cryptology ePrint Archive, Report 2004/152, 2004. `http://eprint.iacr.org/2004/152`.

[123] K. Kurosawa and Y. Desmedt. A new paradigm of hybrid encryption scheme. In *Crypto'04*, volume 3152 of *LNCS*, pages 426–442. Springer, 2004.

[124] K. Kurosawa and S.-H. Heng. From digital signature to ID-based identification/signature. In *PKC'04*, volume 2947 of *LNCS*, pages 248–261. Springer, 2004.

[125] K. Kurosawa and S.-H. Heng. Identity-based identification without random oracles. In *ISH'05*, volume 3481 of *LNCS*, pages 603–613. Springer, 2005.

[126] K. Kurosawa and T. Matsuo. How to remove MAC from DHIES. In *ACISP'04*, volume 3108 of *LNCS*, pages 236–247. Springer, 2004.

[127] F. Laguillaumie and D. Vergnaud. Designated verifier signatures: Anonymity and efficient construction from *any* bilinear map. In *SCN'04*, volume 3352 of *LNCS*, pages 105–119. Springer, 2004.

[128] F. Laguillaumie and D. Vergnaud. Multi-designated verifiers signatures. In *ICICS'04*, volume 3269 of *LNCS*, pages 495–507. Springer, 2004.

[129] F. Laguillaumie and D. Vergnaud. Time-selective convertible undeniable signatures. In *CT-RSA'05*, volume 3376 of *LNCS*, pages 154–171. Springer, 2005.

[130] Y. R. Lee and H. S. Lee. An authenticated certificateless public key encryption scheme. Cryptology ePrint Archive, Report 2004/150, 2004. `http://eprint.iacr.org/2004/150`.

[131] B. Libert and J. J. Quisquater. New identity based signcryption schemes from pairings. In *IEEE Information Theory Workshop*, Paris, France, 2003. `http://eprint.iacr.org/2003/023`.

[132] B. Libert and J. J. Quisquater. Efficient signcryption with key privacy from Gap Diffie-Hellman groups. In *PKC'04*, volume 2947 of *LNCS*, pages 187–200. Springer, 2004.

[133] B. Libert and J. J. Quisquater. The exact security of an identity based signature and its applications. Cryptology ePrint Archive, Report 2004/102, 2004. `http://eprint.iacr.org/2004/102/`.

[134] B. Libert and J. J. Quisquater. Identity based undeniable signatures. In *CT-RSA'04*, volume 2964 of *LNCS*, pages 112–125. Springer, 2004.

[135] B. Libert and J. J. Quisquater. Improved signcryption from q-Diffie-Hellman problems. In *SCN'04*, volume 3352 of *LNCS*, pages 220–234. Springer, 2004.

[136] B. Libert and J. J. Quisquater. Identity based encryption without redundancy. In *ACNS'05*, volume 3531 of *LNCS*, pages 285–300. Springer, 2005.

[137] B. Lynn. Authenticated identity-based encryption. Cryptology ePrint Archive, Report 2002/072, 2004. `http://eprint.iacr.org/2002/072/`.

[138] J. Malone-Lee. Signcryption with non-interactive non-repudiation. *Designs, Codes and Cryptography*. to appear.

[139] J. Malone-Lee. Identity-based signcryption. Cryptology ePrint Archive, Report 2002/098, 2002. `http://eprint.iacr.org/2002/098`.

[140] J. Malone-Lee and W. Mao. Two birds one stone: Signcryption using RSA. In *CT-RSA'03*, volume 2612 of *LNCS*, pages 211–225. Springer, 2004.

[141] U. Maurer and S. Wolf. Diffie-Hellman, decision Diffie-Hellman, and discrete logarithms. In *ISIT'98*, page 327. IEEE Information Theory Society, 1998.

[142] U. Maurer and S. Wolf. The relationship between breaking the Diffie-Hellman protocol and computing discrete logarithms. *SIAM J. Comput.*, 28(5):1689–1721, 1999.

[143] U. Maurer and Y. Yacobi. Non-interactive public-key cryptography. In *Crypto'91*, volume 547 of *LNCS*, pages 498–507. Springer, 1991.

[144] N. McCullagh and P. S. L. M. Barreto. Efficient and forward-secure identity-based signcryption. Cryptology ePrint Archive, Report 2004/117, 2004. `http://eprint.iacr.org/2004/117`.

[145] R. J. McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN progress report*, pages 114–116, 1978.

[146] A. J. Menezes. *Elliptic Curve Public Key Cryptosystems*. Kluwer Academic Publishers, 1993.

[147] A. J. Menezes, T. Okamoto, and S. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Trans. on Inf. Theory*, 39:1639–1646, 1993.

[148] A. J. Menezes, P. Van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[149] R. C. Merkle and M. E. Hellman. Hiding information and signatures in trapdoor knapsacks. *IEEE Trans. on Inf. Theory*, 24:525–530, 2002.

[150] S. Micali and L. Reyzin. Improving the exact security of digital signature schemes. *Journal of Cryptology*, 15(1), 2002.

[151] M. Michels and M. Stadler. Efficient convertible undeniable signature schemes. In *SAC'97*, 1997.

[152] V. Miller. Short programs for functions on curves. Unpublished manuscript, 1986.

[153] V. Miller. The Weil pairing, and its efficient calculation. *Journal of Cryptology*, 17(4):235–261, 2004.

[154] S. Mitsunari, R. Sakai, and M. Kasahara. A new traitor tracing. *IEICE Transactions on Fundamentals*, E85-A(2):481–484, 2002.

[155] A. Miyaji, M. Nakabayashi, and S. Takano. New explicit conditions of elliptic curve traces for FR-reduction. *IEICE Transactions on Fundamentals*, E84-A(5):1234–1243, 2001.

[156] J. Monnerat and S. Vaudenay. Generic homomorphic undeniable signatures. In *Asiacrypt'04*, volume 3329 of *LNCS*, pages 354–371. Springer, 2004.

[157] J. Monnerat and S. Vaudenay. Undeniable signatures based on characters: How to sign with one bit. In *PKC'04*, volume 2947 of *LNCS*, pages 69–85. Springer, 2004.

[158] D. M'Raihi, D. Naccache, D. Pointcheval, and S. Vaudenay. Computational alternatives to random number generators. In *SAC'98*, volume 1566 of *LNCS*, pages 72–80. Springer, 2004.

[159] D. Naccache and J. Stern. A new public-key cryptosystem. In *Eurocrypt'97*, volume 1233 of *LNCS*, pages 27–36. Springer, 1997.

[160] D. Naccache and J. Stern. A new public key cryptosystem based on higher residues. In *6^{th} ACM Conference on Computer and Communications Security*, pages 59–66. ACM Press, 1998.

[161] D. Nalla and K. C. Reddy. Signcryption scheme for identity-based cryptosystems. Cryptology ePrint Archive, Report 2003/066, 2003. `http://eprint.iacr.org/2003/066`.

[162] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC'90*, pages 427–437. ACM Press, 1990.

[163] V. I. Nechaev. Complexity of a deterministic algorithm for the discrete logarithm. *Mathematical Notes*, 55:165–172, 1994.

[164] NIST/NSA. Fips 180-2: Secure hash standard (shs), August 2002.

[165] K. Ohta and T. Okamoto. On concrete security treatment of signatures derived from identification. In *Crypto'98*, volume 1462 of *LNCS*, pages 354–370. Springer, 1998.

[166] T. Okamoto. Designated confirmer signatures and public key encryption are equivalent. In *Crypto'94*, volume 839 of *LNCS*, pages 61–74. Springer, 1994.

[167] T. Okamoto and D. Pointcheval. REACT: Rapid enhanced-security asymmetric cryptosystem transform. In *CT-RSA'01*, volume 2020 of *LNCS*, pages 159–174. Springer, 1994.

[168] T. Okamoto and D. Pointcheval. The gap-problems: A new class of problems for the security of cryptographic schemes. In *PKC'01*, volume 1992 of *LNCS*, pages 104–118. Springer, 2001.

[169] T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In *Eurocrypt'98*, volume 1403 of *LNCS*, pages 308–318. Springer, 1998.

[170] H. Ong and C. P. Schnorr. Fast signature generation with a fiat-shamir like scheme. In *Eurocrypt'90*, volume 473 of *LNCS*, pages 432–440. Springer, 1990.

[171] D. Page, N. P. Smart, and F. Vercauteren. A comparison of MNT curves and supersingular curves. Cryptology ePrint Archive, Report 2004/165, 2004. `http://eprint.iacr.org/2004/165`.

[172] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Eurocrypt'99*, volume 1592 of *LNCS*, pages 223–238. Springer, 1999.

[173] J. Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms. In *Eurocrypt'96*, volume 1070 of *LNCS*, pages 33–48. Springer, 1996.

[174] K. G. Paterson. ID-based signatures from pairings on elliptic curves. *Electronics Letters*, 38(18):1025–1026, 2002.

[175] H. Petersen and H. Michels. Cryptanalysis and improvement of signcryption schemes. *IEE Proceedings - Computers and Digital Technique*, 145(2):149–151, 1998.

[176] D. H. Phan and D. Pointcheval. Chosen-ciphertext security without redundancy. In *Asiacrypt'03*, volume 2894 of *LNCS*, pages 1–18. Springer, 2003.

[177] D. H. Phan and D. Pointcheval. OAEP 3-round: A generic and secure asymmetric encryption padding. In *Asiacrypt'04*, volume 3329 of *LNCS*, pages 63–78. Springer, 2004.

[178] D. H. Phan and D. Pointcheval. About the security of ciphers (semantic security and pseudo-random permutations). In *SAC'04*, volume 3357 of *LNCS*, pages 185–200. Springer, 2005.

[179] J. Pieprzyk and D. Pointcheval. Parallel authentication and public-key encryption. In *ACISP'03*, volume 2727 of *LNCS*, pages 383–401. Springer, 2003.

[180] D. Pointcheval. Practical security in public-key cryptography. In *ICISC'01*, volume 2288 of *LNCS*, pages 1–17. Springer, 2001.

[181] D. Pointcheval. Self-scrambling anonymizers. In *Financial Cryptography'02*, volume 1962 of *LNCS*, pages 259–275. Springer, 2001.

[182] D. Pointcheval and J. Stern. Security proofs for signature schemes. In *Eurocrypt'96*, volume 1992 of *LNCS*, pages 387–398. Springer, 1996.

[183] D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000.

[184] M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

[185] C. Rackoff and D. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Crypto'91*, volume 576 of *LNCS*, pages 433–444. Springer, 1991.

[186] I. Damgård and T. Pedersen. New convertible undeniable signature schemes. In *Eurocrypt'96*, volume 1070 of *LNCS*, pages 372–386. Springer, 1996.

[187] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public key cryptosystems. *Communications of the ACM*, 21(1):120–126, 1978.

[188] R. L. Rivest. The MD5 message digest algorithm. RFC 1321, 1992.

[189] R. Sakai and M. Kasahara. ID-based cryptosystems with pairing on elliptic curve. In *SCIS'03*, Hamamatsu, Japan, 2003. `http://eprint.iacr.org/2003/054`.

[190] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. In *SCIS'00*, 2000.

[191] C. P. Schnorr. Efficient identification and signatures for smart cards. In *Crypto'89*, volume 435 of *LNCS*, pages 239–252. Springer, 1989.

[192] M. Scott. Computing the Tate pairing. In *CT-RSA'05*, volume 3376 of *LNCS*, pages 293–304. Springer, 2005.

[193] I. A. Semaev. Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p. *Math. Comp.*, 67(221):353–356, 1998.

[194] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[195] A. Shamir. Identity based cryptosystems and signature schemes. In *Crypto'84*, volume 196 of *LNCS*, pages 47–53. Springer, 1984.

[196] J. B. Shin, K. Lee, and K. Shim. New DSA-verifiable signcryption schemes. In *ICISC'02*, volume 2587 of *LNCS*, pages 35–47. Springer, 2002.

[197] V. Shoup. Lower bounds for discrete logarithms and related problems. In *Eurocrypt'97*, volume 1233 of *LNCS*, pages 256–266. Springer, 1997.

[198] V. Shoup. Using hash functions as a hedge against chosen ciphertext attack. In *Eurocrypt'02*, volume 1807 of *LNCS*, pages 275–288. Springer, 2000.

[199] V. Shoup. A proposal for the ISO standard for public-key encryption (version 2.1). manuscript, 2001. `http://shoup.net/`.

[200] V. Shoup and R. Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. In *Eurocrypt'02*, volume 1403 of *LNCS*, pages 1–16. Springer, 1998.

[201] N. P. Smart. The discrete logarithm problem on elliptic curves of trace one. *J. of Cryptology*, 12(3):193–196, 1999.

[202] N. P. Smart. An identity based authenticated key agreement protocol based on the Weil pairing. *Electronics Letters*, 38:630–632, 2002.

[203] N. P. Smart and F. Vercauteren. On computable isomorphisms in efficient pairing based systems. Cryptology ePrint Archive, Report 2005/116, 2005. `http://eprint.iacr.org/2005/116`.

[204] R. Steinfeld and Y. Zheng. A signcryption scheme based on integer factorization. In *ISW'00*, volume 1975 of *LNCS*, pages 308–322. Springer, 2000.

[205] C. H. Tan. Key substitution attacks on provably secure short signature schemes. *IEICE Transactions*, 88-A(2):611–612, 2005.

[206] C. H. Tan. On the security of signcryption scheme with key privacy. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E88-A(4):1093–1095, 2005.

[207] C. H. Tan. Analysis of improved signcryption scheme with key privacy. *Information Processing Letters*, 99:135–138, 2006.

[208] C. H. Tan. Security analysis of signcryption scheme from q-DiffieHellman problems. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E89-A(1):206–208, 2006.

[209] H. Tanaka. A realization scheme for the identity-based cryptosystem. In *Crypto'87*, volume 293 of *LNCS*, pages 341–349. Springer, 1987.

[210] Y. Tsiounis and M. Yung. On the security of ElGamal based encryption. In *PKC'98*, volume 1431 of *LNCS*, pages 117–134. Springer, 1998.

[211] S. Tsuji and T. Itoh. An ID-based cryptosystem based on the discrete logarithm problem. *IEEE Journal on Selected Areas in Communication*, 7(4):467–473, 1989.

[212] E. R. Verheul. Evidence that XTR is more secure than supersingular elliptic curve cryptosystems. In *Eurocrypt'01*, volume 2045 of *LNCS*, pages 195–210. Springer, 2001.

[213] G. Wang, F. Bao, C. Ma, and K. Chen. Efficient authenticated encryption schemes with public verifiability. In *60^{th} IEEE Vehicular Technology Conference*, 2004.

[214] G. Wang, R. Deng, D. J. Kwak, and S. J. Moon. Security analysis of two signcryption schemes. In *ISC'2004*, volume 3225 of *LNCS*, pages 123–133. Springer, 2004.

[215] X. Wang, Y. L. Yin, and H. Yu. Finding collisions in the full SHA-1. In *Crypto'05*, LNCS. Springer, 2005. To appear.

[216] X. Wang and H. Yu. How to break MD5 and other hash functions. In *Eurocrypt'05*, volume 3494 of *LNCS*, pages 19–35. Springer, 2005.

[217] B. Waters. Efficient identity-based encryption without random oracles. In *Eurocrypt'05*, volume 3494 of *LNCS*, pages 114–127. Springer, 2005.

[218] G. Yang, D. S. Wong, and X. Deng. Analysis and improvement of a signcryption scheme with key privacy. In *ISC'05*, volume 3650 of *LNCS*, pages 218–232. Springer, 2005.

[219] X. Yi. An identity-based signature scheme from the Weil pairing. *IEEE Communications Letters*, 7(2):76–78, 2003.

[220] T. H. Yuen and V. K. Wei. Fast and proven secure blind identity-based signcryption from pairings. In *CT-RSA'05*, volume 3376 of *LNCS*, pages 305–322. Springer, 2005.

[221] D. H. Yum and P. J. Lee. New signcryption schemes based on KCDSA. In *ICISC'01*, volume 2288 of *LNCS*, pages 305–317. Springer, 2001.

[222] D. H. Yum and P. J. Lee. Generic construction of certificateless encryption. In *ICCSA'04*, volume 3043 of *LNCS*, pages 802–811. Springer, 2004.

[223] D. H. Yum and P. J. Lee. Generic construction of certificateless signature. In *ACISP'04*, volume 3108 of *LNCS*, pages 200–211. Springer, 2004.

[224] D. H. Yum and P. J. Lee. Identity-based cryptography in public key management. In *EuroPKI'04*, volume 3093 of *LNCS*, pages 71–84. Springer, 2004.

[225] F. Zhang, Y. Mu, and W. Susilo. Identity-based partial message recovery signatures (or how to shorten ID-based signatures). In *Financial Cryptography'05*, LNCS. Springer, 2005. to appear.

[226] F. Zhang, R. Safavi-Naini, and W. Susilo. Attack on han et al.'s ID-based confirmer (undeniable) signature at ACM-EC'03. Cryptology ePrint Archive, Report 2003/129, 2003. `http://eprint.iacr.org/2003/129`.

[227] F. Zhang, R. Safavi-Naini, and W. Susilo. An efficient signature scheme from bilinear pairings and its applications. In *PKC'04*, volume 2947 of *LNCS*, pages 277–290. Springer, 2004.

[228] Y. Zheng. Digital signcryption or how to achieve cost (signature & encryption) << cost(signature) + cost(encryption). In *Crypto'97*, volume 1294 of *LNCS*, pages 165–179. Springer, 1997.

[229] Y. Zheng and J. Seberry. Practical approaches to attaining security against adaptively chosen ciphertext attacks. In *Crypto'92*, volume 740 of *LNCS*, pages 292–304. Springer, 1992.

APPENDIX A

Publications list

Publications in refereed international conferences:

- Benoît Libert, Jean-Jacques Quisquater, *New identity based signcryption schemes from pairings*, IEEE Information Theory Workshop 2003, pages 155-158, 2003.
- Benoît Libert, Jean-Jacques Quisquater, *Efficient Revocation and Threshold Pairing Based Cryptosystems*, 22nd Symposium on Principles of Distributed Computing (PODC 2003), pages 163-171, ACM Press, 2003.
- Benoît Libert, Jean-Jacques Quisquater, *Identity Based Undeniable Signatures*, in Cryptographer's Track – RSA 2004, Vol. 2964 of LNCS, pages 112-125, Springer, 2004.
- Benoît Libert, Jean-Jacques Quisquater, *Efficient Signcryption with Key Privacy from Gap Diffie-Hellman Groups*, in Public Key Cryptography 2004 (PKC'04), Vol. 2947 of LNCS, pages 187-200, Spinger, 2004.
- Julien Cathalo, Benoît Libert, Jean-Jacques Quisquater, *Cryptanalysis of a Verifiably Committed Signature Scheme based on GPS and RSA*, in Information Security Conference (ISC) 2004, Vol. 3225 of LNCS, pages 52-60, Springer, 2004.
- Benoît Libert, Jean-Jacques Quisquater, *What Is Possible with Identity Based Cryptography for PKIs and What Still Must Be Improved*, in EuroPKI'04, Vol. 3093, pages 57-70, Springer, 2004.
- Benoît Libert, Jean-Jacques Quisquater, *Improved Signcryption from q-Diffie-Hellman Problems*, in 4h Conference on Security in Communication Networks '04 (SCN'04), Vol. 3352 of LNCS, pages 220-234, Spinger, 2004.
- Benoît Libert, Jean-Jacques Quisquater, *Identity Based Encryption without Redundancy*, in 3rd Applied Cryptography

and Network Security conference (ACNS'05), Vol. 3531 of LNCS, pages 285-300, Springer, 2005.

- Paulo S. L. M. Barreto, Benoît Libert, Noel McCullagh, Jean-Jacques Quisquater, *Efficient and Provably-Secure Identity-Based Signatures and Signcryption from Bilinear Maps*, in Asiacrypt 2005, LNCS series, Springer, 2005 (to appear).
- Julien Cathalo, Benoît Libert, Jean-Jacques Quisquater, *Efficient and Non-Interactive Timed-Release Encryption*, 7th International Conference on Information and Communications Security (ICICS'05), LNCS series, Springer, 2005.