Cryptology and Information Security Series

CISS

Identity-Based Cryptography

Marc Joye and Gregory Neven (Eds.)

IOS Press

IDENTITY-BASED CRYPTOGRAPHY

Cryptology and Information Security Series

Volume 2

ISSN 1871-6431

Identity-Based Cryptography

Edited by

Marc Joye

Thomson R&D, France

and

Gregory Neven

IBM Zürich Research Laboratory, Switzerland

IOS Press

Amsterdam • Berlin • Oxford • Tokyo • Washington, DC

ISBN 978-1-58603-947-9
Library of Congress Control Number: 2008940895

Publisher
IOS Press
Nieuwe Hemweg 6B
1013 BG Amsterdam
The Netherlands
fax: +31 20 687 0019
e-mail: order@iospress.nl

Distributor in the UK and Ireland
Gazelle Books Services Ltd.
White Cross Mills
Hightown
Lancaster LA1 4XS
United Kingdom
fax: +44 1524 63232
e-mail: sales@gazellebooks.co.uk

Distributor in the USA and Canada
IOS Press, Inc.
4502 Rachael Manor Drive
Fairfax, VA 22032
USA
fax: +1 703 323 3668
e-mail: iosbooks@iospress.com

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009

Foreword

In an active field like that of cryptography, a problem that remains open for seventeen years must be a pretty tough problem. In a practically relevant field like that of cryptography, a solution that inspires hundreds of follow-up papers within a few years' time must be a pretty interesting solution.

Posed as an open problem in 1984, but efficiently instantiated only in 2001, identity-based encryption hasn't left the forefront of cryptographic research since. Praised by fans as the economical alternative to public-key infrastructures, booed by critics for its inherent key escrow — was that 1984 you said? — identity-based cryptography is also the topic of numerous debates in the cryptographic community.

This book looks beyond the controversy and intends to give an overview of the current state-of-the-art in identity-based cryptography. Since research on the topic is still actively continuing, this is necessarily a snapshot of a field in motion, rather than the final word about it. Still, we felt the main concepts have by now sufficiently matured to collect them in a single dedicated volume.

Each of the chapters in this volume is written by international experts on the topic. Our first word of thanks goes to the authors for their top-quality contributions to the book. Our special gratitude is due to Jean-Luc Beuchat, Jérémie Detrey, David Galindo, Kenny Paterson, and Nigel Smart who have looked over various portions of the book and have given comments and suggestions, and to Michel Abdalla for letting us use his extensive bibliographic library. We would also like to thank Juliette Joye for the beautiful illustration on the cover of this book. Finally, we would like to thank the people at IOS Press for the smooth interaction.

September 2008

Marc Joye
Gregory Neven

Contents

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The author and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-1

Chapter I

Introduction to Identity-Based Cryptography

Antoine JOUX
DGA and University of Versailles St-Quentin-en-Yvelines, France

Abstract. Identity-based cryptography is a new development of public-key cryptography. It was first proposed by Adi Shamir at CRYPTO '84. However, it took the cryptographic community a long while to produce effective identity-based cryptosystems. Indeed, this solution only appeared at the beginning of the twenty-first century. Nowadays, identity-based cryptography has become a very active field of research. This introductory chapter presents the basics of identity-based cryptography and briefly surveys its early history.

1. Public-Key Cryptography, Certificates and Identity-Based Cryptography

Identity-based cryptography is an extension of the public-key paradigm, which was initially suggested by Adi Shamir [Sha85] at CRYPTO '84. In order to better understand identity-based cryptography, we start by reviewing how traditional public-key systems are usually put to use in real-life applications. First, to offer reasonable speed, public-key encryption systems are usually used in conjunction with a secret-key encryption scheme. More precisely, the public-key scheme is used in order to produce a shared encryption key for the secret-key scheme, known to the sender and receiver of the communication. Once this is done, they simply use this common secret key for encrypting the rest of the communication. This initial phase is usually called a key exchange protocol. It can be devised in several ways. The simplest approach is simply to let the sender encrypt a random value R with a public-key encryption scheme such as RSA, using the receiver's public key. Since R can be obtained by the receiver after decryption, it is a common value which can be used to key the secret-key encryption. Note that to avoid simple multiplicative attacks against RSA, for example the attacks described in [BJN00], R should preferably be of the length of the RSA modulus. This means that R is usually too long and must be truncated to obtain the secret key. The other classical approach is to use Diffie-Hellman key exchange, either in the multiplicative subgroup of a finite field or on an elliptic curve. There, the common value is no longer chosen by the sender but instead implicitly generated by combining a random value together with the public key of the receiver. Note that once again this common value is usually too long and needs to be truncated before keying the secret-key encryption.

In both cases, the key exchange protocol requires the receiver's public key as input. As a consequence, before sending any message, the sender should take care to go and obtain this public key. Moreover, he should make sure that the public key he obtains is indeed the correct one. This introduces two important steps in real life implementation, fetching the public key and verifying it. At first, it may seem that the second step is the difficult one. Getting data is reasonably easy, verifying that it is genuine seems to be much more difficult. However, there is a simple solution that allows one to verify public keys: public-key certificates. At the most basic level, a public-key certificate is simply a message that asserts: "The public key of user Bob is upk_B." To make sure that the certificate is genuine, it needs to be signed. Assuming that the signature scheme is cryptographically sound, such a certificate cannot be forged. Of course, for certificates to be useful, several conditions need to be satisfied. The end user that wants to send a message to Bob should be able to verify the signature. As a consequence, the first condition is that he needs to know in advance the public verification key of the signer. The second condition is that the end user should trust the signer. Indeed, if the signer is dishonest, he can easily substitute Bob's key in the certificate by another key of his choice. Doing this could clearly allow him to decrypt any message that a user intends to send to Bob. These two conditions are not always easy to satisfy. First, if Alice doesn't know the public key that would allow her to verify Bob's certificate, she needs to obtain another certificate that brings her this key. Of course, if she doesn't know the verification of this second certificate, she needs a third one, ... until she finally reaches a certificate she can verify. This sequence of successive certificates is called a certification path. Of course, a given certification path can only be verified if the user knows a public key that allows him to enter this path. This implies that the certificate approach can only work when all users are initially given at least one public key in a secure manner. This is a relatively minor assumption, since such a key can be pre-installed by a manufacturer and by a system administrator. From this single public key, all certificates can, in principle, be functionally verified if we ensure that a certification path exists between all pair of users that can possibly want to communicate. This can be ensured by creating a certification tree, where a root certification authority certifies delegate authorities, which certifies sub-delegates until the final users are reached. Each user should know the public keys of all the authorities between him and the root. Thus checking the certificate of another user in the tree can be achieved by starting at an authority shared between the two users. To check a very distant user one simple starts verifying the certification path from the root authority.

A much more difficult problem when using such certificates is the management of trust. What guarantees that you can trust the owner of the verification key you initially hold? In fact, the answer to this question is extremely context sensitive. On one hand, within a hierarchical organization, this problem is easily solved. The organization designates a system administrator who is in charge of setting up a certification authority. Since the administrator is trusted by the organization, he can be trusted by the users *for professional communications.* On the other hand, for individual users, there is no easy answer. If the initial certificate verification key is provided by a company or a government, not all users will trust this initial key. To sidestep this difficulty, some systems propose to let users create certificates. In such a system, any user can use a certificate which has been created by a friend he trusts. However, with these systems, another problem quickly arises. Why should anyone trust friends of friends of friends, i.e. persons he never met? This shows that in a direct person-to-person trust model, the quick dilution of trust quickly voids the value of certificates. This trust issue can be limited by obtaining the same key though several independent channels until one is satisfied that the public key is indeed valid, however, this is a expensive process and the resulting trust is hard to quantify.

Where individuals are involved, another issue prevents the use of public-key encryption in electronic communication. Sending encrypted messages is usually a cumbersome process. Each time a user wants to communicate with a new person, he needs to get a public key or a certificate for this person, to check that the key corresponds to an algorithm his encryption software is compatible with and finally to proceed to the encryption step. Moreover, if the person has not already set up a public key, the user needs an initial contact to ask for one. These practical difficulties are real issues which usually forestall or prevent the use of public-key encryption. Indeed, if no one sends them encrypted mail, most people do not feel the need for a public key. Moreover, when someone does not possess a public key, it is much easier to send him unencrypted information.

Identity-based encryption offers a nice solution to a large fraction of these practical problems. Of course, there is no such thing as a free lunch and the associated cost is that in identity-based cryptosystems, the trust issues are even more touchy than with regular public-key systems. The basic foundational remark of identity-based encryption is that even in the case of unencrypted communications, the user already needs to learn some basic information before he can communicate with another person. At the very least, he should obtain his telephone number, his e-mail address or a similar information. As pointed out by Shamir in his seminal paper [Sha85], it would be extremely nice if this basic information could replace the need for a encryption key altogether.

From the point of view of functionality, this would be extremely practical. First, one could send encrypted messages to anyone, without worrying about their public key. Without, in fact, even worrying about their having a public key. In the identity-based framework, encryption is always possible. Of course, if the recipient of a message does not know his private decryption key, he cannot decrypt the messages he receives. However, this gives him a strong motivation to go and get this key. Clearly, the key ingredient to identity-based encryption is going to be the computation of these private keys. Of course, it should not be possible to efficiently compute the private key from the public identity without any additional trapdoor information. Otherwise, anyone could perform the same computation and the system would offer no security. However, being identity-based, the system cannot be based on user specific information other than their public identity. As a consequence, the computation of the private keys needs to be based on a global trapdoor information, common to all or at least many users. This implies that the owner of this trapdoor information is able to compute the private keys of all users. From a functional point of view, this is a nice feature which allows all users to obtain their private key from him. For this reason, this owner is usually called the key generation center of the identity-based cryptosystem. From the security point of view, this is less convenient, since the complete security of the system hinges on a single centralized point. All users need to trust the key generation center, its ability to keep the global trapdoor information secret and its capability to efficiently deliver to any valid user its own private key, after verifying his identity. As a consequence, the level of trust that is required is even stronger than the level of trust required for public-key certificates. A certification authority can indeed generate fake certificates to attack the system, however, this leaves evidence and sooner or later someone is going to notice the discrepancy. With a key generation center, the situation is very different. The key generation center can silently keep copies of all keys he generates, or alternatively recompute these keys at any point in time. With these keys, he can clearly decrypt any communication that he eavesdrops. The eavesdropping may leave some evidence, but from the point of view of the key generation center, listening in the communications is no more difficult than listening in unencrypted communications. In other words, a key generation center may act as a key escrow agent if it so wishes. Key escrow has been heavily discussed in the past and is often considered as a bad idea; see [AAB$^+$97] for a survey of this discussion. However, in identity-based cryptography, key escrow is not introduced for its own sake but as a necessary consequence of added functionalities. Thus, in this context, it might become more acceptable.

However, a major weak point of identity-based cryptosystems is their key generation process. A potential direction for improvement worth considering would be to create identity-based systems involving several key generation authorities, where a single cheating key generation authority cannot compromise the system without help from the other authorities. Such a scheme could mitigate the trust issue, at the cost of making the private key generation step heavier, requiring the users to interact with several key generation authorities. Another approach proposed in [Goy07], is to introduce a cryptographic mechanism that ensures that a cheating key authority has to construct several different decryption keys for the same user and can hopefully be detected.

The key escrow discussion also permits to shed some additional light on what might be acceptable and what certainly is not. In some contexts, it might be useful and legitimate to escrow a decryption key. For example, it might help a company make sure that no vital trade secret will vanish if a key person disappears in a plane crash. On the other hand, there are no legitimate reasons for ever escrowing a signature or authentication key, such a "feature" can only lead to abuse. This basic distinction between encryption and signature is extremely important and should be kept in mind when devising identity-based cryptosystems.

2. Formalizing Identity-Based Cryptography

This section makes the notion of identity-based cryptography more precise and gives a formal definition of this notion.

From the formal definition, it also shows a few elementary facts. In particular, it explains that realizing identity-based signature schemes or identity-based interactive protocols is straightforward (for any existing regular signature scheme or protocol).

An identity-based cryptographic protocol is a family of algorithms, usually consisting of four algorithms, that solves a cryptographic problem. Regardless of the specific problem, there are two algorithms specific to the identity-based paradigm. The first algorithm called Setup, generates the cryptosystem parameters mpk and a master key msk. This algorithm is run by the key generation center who then publishes the system parameters and keeps msk secret. The second algorithm is called KeyGen. Using the system parameters, a user identity id and the master secret msk, it generates the private key usk

of the corresponding user. This algorithm is run by the key generation center on behalf of a user that requests his own private, after checking that the user matches his claimed identity. The other algorithms in an identity-based system depend on the precise cryptographic primitive that the system realizes. For example, a signature scheme requires a Sign algorithm for producing a signature and a Vf algorithm for verifying a signature. Similarly, a key exchange algorithm requires a GenKey algorithm for generating a shared key and auxiliary data on the sender side and a GetKey algorithm to obtain the shared key on the recipient side. Finally, an encryption algorithm has an Enc encryption algorithm and a Dec decryption algorithm. Of course, this can be generalized to a variety of other cryptographic problems, such as identification, voting, ...

With this formalization in mind, we see that there are two very different classes of identity-based cryptosystems. The distinction is based on the natural sequencing of the cryptographic primitive. The class of systems where the first call to a cryptographic primitive is made by the owner of the private key is very easy to realize. The other class, where the first call is made by the other user, is much more difficult. Indeed, when the owner of the private key starts, he can embed information in the message he creates in order to convey all the necessary information about his public key. On the contrary, when the other user starts, he really needs to derive everything from the identity string and the system parameters, without any help from an auxiliary input.

The typical example of the first case is the folklore generic construction of an identity-based signature scheme from any regular signature scheme. The regular signature scheme is a triple of algorithms RegGenerateKeyPair, RegSign and RegVerify. The first algorithm generates a pair containing a public key and the corresponding private key. The second algorithm takes as input a private key and a message and produces a signature. The third algorithm takes as input a public key, a message and a signature and output either `valid` or `invalid`. Using such a signature scheme, we can now generically construct an identity-based signature scheme:

Setup The key generation center runs RegGenerateKeyPair and produces a public key and the corresponding private key. The public key is published as part of the system parameters mpk. The private key becomes the key generation center main key msk.

Key Generation In order to obtain their identity-based private key users perform the following steps:

- The user runs RegGenerateKeyPair and produces a public key upk and the corresponding private key usk
- The user transmits his identity string id and the public key upk to the key generation center (normally using a secure channel of communication). For increased security, the key generation center may require a proof of knowledge of the private key usk.
- After verifying the user's identity, the key generation center forms a message: "The public key upk is the signing key of user id." and signs it with the private key msk using RegSign. The resulting signature is denoted by $ucert$. This amounts to creating a key certificate that attributes the key upk to the user with identity id.
- The certificate $ucert$ is returned to the user.

Signature With this certificate in hand, the user is now ready to sign messages. The identity-based signature algorithm Sign is defined as:

- The signing user runs RegSign on the message M to be signed, using usk as signing key. The resulting signature is denoted by s.
- The user forms the triple

$$\sigma = (upk, ucert, s)$$

and produces it as his identity-based signature.

Verification To verify an identity-based signature, i.e. a triple

$$\sigma = (upk, ucert, s)$$

announced to be a signature of message M by the user with identity id, is done using algorithm Verify:

- The verifying user runs RegVerify with the public key mpk on the signature $ucert$ and the message "The public key upk is the signing key of user id."
- The verifying user runs RegVerify with the public key upk on the signature s and the message M.
- If at least one verification returned `invalid`, then output `invalid`. Otherwise, return `valid`.

Of course, this identity-based signature is nothing more than a certificate based signature scheme, with the certificate being included in every signature, in order to remove the need to obtain it through another channel. A disadvantage of this approach is that the resulting signatures are more than twice as long as the regular signatures they are based on. A noteworthy feature of this generic scheme is that it neatly avoids the trust issue: the key generation authority never accesses the user's signing key and thus cannot escrow it. Thus it can only forge user signatures by producing fake certificates which are hopefully detected in the long run.

3. The Long Road Towards Identity-Based Encryption

After the invention of the identity-based cryptography paradigm by Shamir [Sha85], it was clear that identity-based signature schemes were feasible (cf. previous section). However, identity-based encryption was a much more challenging task. Neither the RSA nor the Diffie-Hellman cryptosystems can easily be transformed into identity-based encryption systems. With RSA, a tempting idea would be to use a shared encryption modulus $N = pq$ for all users and to derive the individual encryption exponents from the users' identities. The factorization of N would then be a common secret used to produce decryption exponents. However, it is well-known that given an RSA modulus N, an encryption exponent e and the corresponding description exponent d, it is easy to recover the factorization of N. A probabilistic method was already described in the paper of Rivest, Shamir and Adleman [RSA78] which proposed the RSA cryptosystem. More recently, Coron and May [CM07,May04] described a deterministic algorithm that solves the same problem. As a consequence, any individual user of the system could recover the factorization of N from the knowledge of his own decryption key. This common modulus attack is described in more details in [MvV97, p. 289]. As a consequence, such an identity-based system would be very insecure. Thus, any attempt to use RSA as an identity-based system seems to require the ability to produce from the user identity I an hard to factor number N_I, with in addition a global trapdoor to allow the factorization by the key generation center. Producing such numbers is currently an open problem.

A different approach proposed in [Tan88] makes use of RSA numbers and yields an identity-based key exchange protocol. However, this protocol is very weak against users' collusion. More precisely, the system contains a parameter t which plays the role of a threshold. The size of the private keys and the running time of the key exchange algorithm are linear in t; the private key of the key generation center is secure against collusions of up to t users.

3.1. The Trapdoor Discrete Logarithm Approach

Deriving an identity-based variation of Diffie-Hellman schemes seems easier. Indeed, it suffices to construct a group with a trapdoor that allows to efficiently compute discrete logarithms. Since in a regular Diffie-Hellman key pair, the public key is obtained by raising a fixed group element to the value of the chosen private key, such a trapdoor discrete logarithm group would allow to reverse the process: choosing the public key from the identity and obtaining the private key after a discrete logarithm computation. This approach to identity-based encryption was proposed independently by Murakami and Kasahara in [MK90] and by Maurer and Yacobi in [MY91]. The basic idea is to use discrete logarithms in a multiplicative group defined modulo N, where N is a composite number. As special case of interest is to consider $N = pq$ is a RSA number. With this special case, the rationale is that since N has twice as many bits as p or q, factoring N is going to be much harder than taking discrete logarithms modulo p or q. Moreover, using the Chinese remainder theorem, it should be possible to paste together a logarithm modulo p or a logarithm modulo q into a logarithm modulo N. However, this is more subtle than it initially seems. Discrete logarithms modulo p are numbers defined modulo $p - 1$ and discrete logarithm modulo q are defined modulo $q - 1$. Since $p - 1$ and $q - 1$ are not coprime, the two discrete logarithms cannot necessarily be pasted together by the Chinese remainder theorem. If p and q are chosen to ensure that the greatest common factor of $p-1$ and $q-1$ is 2, then this compatibility issue can be expressed in a reasonably simple form. Assume that we are given g which generates the full multiplicative group modulo p and modulo q. Then any number x coprime to N has a discrete logarithm l_p modulo p, i.e. $x \equiv g^{l_p} \pmod{p}$ and a discrete logarithm l_q modulo q. If l_p and l_q are either both odd or both even then we can put them together and construct a number l such that:

$$l \equiv l_p \pmod{p-1} \quad \text{and} \quad l \equiv l_q \pmod{q-1} \ .$$

In that case, $g^l \equiv x \pmod{N}$ and l is the logarithm of x we wanted to extract. The parity condition on l_p and l_q means that x has a discrete logarithm modulo N if and only if x is either a quadratic residue modulo both p and q or a quadratic non-residue modulo both p and q. As a consequence, only half of the invertible numbers modulo N have a discrete logarithm. In the general case, with more than two factors, x needs to have the same type of quadratic residuosity with respect to all factors.

The main difficulty to construct an identity-based encryption with this kind of trapdoor for the discrete logarithm problem is to make sure that all users have access to a public function that associate to any identity a number that has a discrete logarithm. One easy option is to first map the identity to a value modulo N, without any special restriction, and then to square this value. This clearly guarantees that the final chosen value has a valid discrete logarithm. However, as we will see in the sequel, this leads to an insecure system.

In order to avoid squaring, we need to introduce a different idea, which only works with two factors. The good point when we have only two factors is that given x, it is possible to test whether it satisfies the quadratic residuosity condition with a simple computation that does not require knowledge of the factorization of N. Without going into the technical details, let us say that x has a discrete logarithm modulo $N = pq$ when the Legendre-Jacobi-Kronecker symbol (often called the Jacobi symbol) of x and N is equal to 1. With more than two factors, having a Jacobi symbol of 1 only means that x may be a quadratic non-residue modulo an even number of factors only. Thus in the general case, the Jacobi symbol is not enough to test for the existence of a discrete logarithm. Thanks to this efficient test, given any public process, for example based on a hash function, that transforms the identity of a user into a number x modulo N, this number can directly be used as the user's public key if its Jacobi symbol is 1. Otherwise, two options are possible, either increment or similarly modify the number x until a number with Jacobi symbol is reached, or remark that the Jacobi symbol is multiplicative with values 1 or -1 and multiply x by another number x_0 with Jacobi symbol -1. This second solution is very elegant and simply requires x_0 to be fixed as a parameter of the system at setup.

Security aspects. Before studying the applicability of such systems, it is important to discuss their security. One important aspect is that enforcing the existence of a discrete logarithm through squaring does not work. Indeed, if the public key x is created by squaring, then x can be written as z^2. Moreover, the user corresponding to the key x, learns a number l such that $g^l \equiv x \pmod{N}$. Among the values l obtained by users, about half are even. In that case, the corresponding user can write:

$$z^2 = (g^{l/2})^2 \pmod{N} .$$

Of course, such an equality of squares often yields the factorization of N. It suffices to compute the greatest common divisor of N and $z - g^{l/2}$. As a consequence, using squares leads to a completely insecure system, as was noticed in [MY92]. These security issues are further discussed in [MK05].

Without squares, this attack in its basic form is avoided. Nevertheless, we should pay special attention to the public-key generation and, in particular, make sure that it is based on a cryptographic hash function. To understand why, let's assume that a corrupt user can submit a public key of his choice x and obtain the corresponding discrete logarithm l if it exists. In this scenario, the user can choose for x a value g^m, where he chooses for m a number larger than N. When he obtains l, this value is smaller than $\phi(N)/2 = (p-1) \cdot (q-1)/2$ and thus than N. As a consequence, $m - l$ is a multiple of $\phi(N)/2$. This situation is similar to having both a RSA encryption key and the corresponding decryption key and allows the corrupt user to factor N and break the cryptosystem. If the public-key generation is based on a strong hash function, this second attack is no longer possible.

Practicality issue. The main drawback of the identity-based system of Maurer and Yacobi or of Murakami and Kasahara is that in order to make sure that N is hard to factor, the factors p and q should be large. This in turn implies that discrete logarithm modulo p and q are going to be hard to extract. Since the best known algorithms for factoring and discrete logarithms are essentially the same and since they are subexponential, the system cannot be secure unless the discrete logarithm computations are incredibly expensive. For this reason, despite the elegant construction, this identity-based cryptosystem is not very practical. Note that any attempt to make the discrete logarithm computations easier by choosing special values of p and q usually leads to disaster by also giving a shortcut for factorization.

In order to improve the practicality of this approach, the only hope is to increase the number of factors of N. However, since we cannot use a squaring to ensure that public keys have a discrete logarithm, we need another technique to work around the lack of an easy test based on the Jacobi symbol. In fact, if we are ready to accept longer ciphertexts, this can be done reasonably easily. As before, let us assume that x denotes a public key resulting from a hash process. The first step is to transform x into a number with Jacobi symbol 1 modulo N, by potentially multiplying by a system parameter x_0 chosen as before. Remember that this does not suffice to ensure that x has a discrete logarithm. However, when N has t factors, the key generation center can add some additional parameters $x_1, x_2, \ldots, x_{t-2}$ and know for sure that by multiplying x by a subset of these x_i values it is possible to reach a value with a valid discrete logarithm. Moreover, when after extracting discrete logarithm of x modulo each factor of N, it is easy to determine the exact subset that needs to be used. Thus, the key generation center can give away such a subset, together with a discrete logarithm of x multiplied by all the subset elements. From this, a Diffie-Hellman like key exchange can be derived, it suffices for the sender to determine the public key x assigned to the intended recipient and to transmit $(g^r, x_1^r, x_2^r, \ldots, x_{t-2}^r)$. From this information and from his private key, the recipient can easily recover x^r which can thus serve as a common key between the sender and the receiver. Note that with multiple recipients, the overhead cost associated with sending x_i^r can be amortized by reusing the same r for all recipients.

If we are willing to pay this cost, then the gap between the cost of computing private key and the cost of attacking the scheme becomes much larger and can be potentially be considered as practical. From a asymptotic complexity point of view, this gap is mostly due to the fact that the current most efficient general discrete logarithm algorithm, the Number Field Sieve [LL93], is much more efficient than the current best algorithm for factoring numbers with small factor, the Elliptic Curve Method (ECM) from [Len87]. Indeed, using the traditional notation:

$$L_p(\alpha, c) = \exp((c + o(1)) \log(p)^{\alpha} \log\log(p)^{1-\alpha}),$$

and the shorthand $L_p(\alpha)$ when c is left unspecified, the complexity of discrete logarithms modulo p is $L_p(1/3)$, while the complexity of finding the small factor p is $L_p(1/2)$. Since, in addition, there is a Number Field Sieve algorithm to factor N in time $L_N(1/3)$, we achieve a nice balance by choosing prime factors of size $L_N(2/3)$. With this choice, the cost of the discrete logarithm computations, expressed as a function of N is $L_N(2/9)$, while the cost of the factoring algorithms is $L_N(1/3)$. Note that this balance is achieved with:

$$t \approx \left(\frac{\log(N)}{\log\log(N)} \right)^{1/3} \quad \text{factors} \enspace .$$

From a practical point of view, the record of the largest factor found by ECM is currently 67 digits (it was found by B. Dodson in 2006, see [Zim]), while the record of discrete logarithms in prime fields is 160 digits by T. Kleinjung (see [Kle07]). From these records, we see that there is some margin to set up a practical system, where the private keys can be computed is reasonable time and factoring remains unreachable. For example, in order to stay above the 1024 bits limit, three or four factors of 120 digits each could be considered. However, this overall approach of using discrete logarithms with a trapdoor at best yields a partial solution. Because of the extremely high computational requirements on the key generation center, it is going to be unacceptable for most practical applications.

3.2. Two Solutions of the Twenty-First Century

In fact, no efficient solution was found for the problem of identity-based encryption during the twentieth century. Then suddenly, in 2001, two nice solutions were discovered. One of these two solutions discovered by Cocks [Coc01] and later improved by Boneh, Gentry and Hamburg in [BGH07a], makes use of the properties of quadratic residues modulo RSA composite. The other solution relies on the existence of efficiently computable bilinear pairing on some groups, mostly based on elliptic curves. It was proposed by Boneh and Franklin [BF01] at CRYPTO 2001. Compared to the solution presented in Section 3.1, the advantage of these new solutions is that their key generation is efficient, more precisely, knowing the main secret it can be performed in polynomial time.

Identity-based encryption from pairings. This solution [BF01] to the identity-based encryption problem is one of the major applications of pairing-based cryptography. At its core, it is based on the fact that some groups, for example groups given by well-chosen elliptic curves, admits efficiently computable [Mil04] pairings or bilinear maps to other groups of the same cardinality, often groups of roots of unity. More precisely, a pairing $\hat{e}$, from a group $\mathbb{G}$ to another group $\mathbb{G}_\mathrm{T}$ is a map:

$$\hat{e}\colon \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_\mathrm{T}$$

such that, for all elements P and Q in $\mathbb{G}$ and for all integers a and b, we have $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$. Note that traditionally, $\mathbb{G}$ is written additively and $\mathbb{G}_\mathrm{T}$ is written multiplicatively. Of course, such a pairing could be trivially achieved by choosing for $\hat{e}$ the constant map equal to the unit element of $\mathbb{G}_\mathrm{T}$. To exclude this case, we say that the pairing should be non-degenerate.

To create the identity-based encryption, we also require a hash function that maps any given identity id to a point P_{id} in $\mathbb{G}$ in a collision resistant manner. The key generation center in this system chooses a random secret s and publishes a triple of points $(P, Q, \mathcal{Q})$ as the system parameters. The last two points are related by $\mathcal{Q} = sQ$. When a user whose identity maps to P_{id} asks for his private key, he receives the point

$\mathcal{P}_{id} = sP_{id}$. After this setup, anyone can perform a simple key exchange with this user as follows:

1. Choose a random value r.
2. Send to user id the point rQ.
3. Compute the shared secret as $\hat{e}(P_{id}, r\mathcal{Q})$.
4. Upon reception, id computes the shared secret as $\hat{e}(\mathcal{P}_{id}, rQ)$.

It is easy to check that both values used as the shared secret are equal. Indeed, thanks to bilinearity, they can both be identified with $\hat{e}(P_{id}, Q)^{rs}$.

Of course, there is much more to this solution than in this nutshell description, the reader can refer to Chapters II, IV and XII.

Identity-based encryption from quadratic residuosity. In its first version [Coc01], the solution based on quadratic residuosity modulo RSA numbers is not very efficient in terms of bandwidth. Sending a single encrypted bit is achieved by transmitting a value modulo the RSA number, which essentially has the same size as the RSA number itself. This problem is fixed by the improvement presented in [BGH07a]. However, this improved solution is slower than Cocks' original approach. Here, we briefly describe Cocks' solution and refer to the original paper ([Coc01]) for a detailed discussion. All computations are done modulo an RSA number $N = pq$, whose decomposition into primes is the key generation master secret. As in the pairing based case, we need a specific hash function. Here, the hash function needs to maps an identity I to a value x_I, whose Jacobi symbol modulo N is 1. Thus, either x_I is a quadratic residue modulo both p and q or a quadratic non-residue modulo both. When x_I is a quadratic residue, the corresponding private key is a square root r_I of x_I modulo N. Note that to avoid trivial attacks on the scheme, the key generation center should make sure that for a given identity x_I, it always outputs the same root r_I. When x_I is not a square, the private key r_I is a square root of ux_I where u is an additional system parameter chosen by the key generation center. By construction, u is a quadratic non-residue modulo both p and q and thus multiplying by u turns non squares into squares. With this setting, sending a single bit b to user I is done as follows:

1. Choose random values t and t', whose Jacobi symbol with N both are $(-1)^b$.
2. Send to user I the values $d = (t^2 + x_I)/t$ and $d' = (t'^2 + ux_I)/t'$.
3. Upon reception, I computes either $d + 2r_I$ if $r_I^2 = x_I$ or $d' + 2r_I$ if $r_I^2 = ux_I$. He finally recovers b by computing the Jacobi symbol of this value with N.

To check that the decryption is correct, we remark that $d + 2r_I = (t + r_I)^2/t$ in the first case and $d' + 2r_I = (t' + r_I)^2/t'$ in the second case. Since neither inverting nor multiplying by a square change the value of the Jacobi symbol, the receiver computes the Jacobi symbol of either t or t' with N. Since both are equal to $(-1)^b$, this ensures decryption.

4. Perspectives

One important research path in the field of identity-based cryptography is to discover new cryptosystems, based on a variety of hard problems, to complete the two solutions based on quadratic residuosity and pairings which are becoming pretty standard. Some alternative approaches are already discussed in this book. For example, hard problems from coding theory are used in Chapter VIII.

Discovering new applications of identity-based encryption, especially applications which cannot be achieved without using the identity-based paradigm is also a very interesting field of research. Current applications are heavily discussed in Chapters IV, VII, IX and X.

Of course, making identity-based cryptography practical and deploying it securely in practice is also essential. This implies several implementation issues which are discussed in Chapters XII, XIII and XIV.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The author and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-13

Chapter II

Pairings on Elliptic Curves

Frederik VERCAUTEREN [1]
Katholieke Universiteit Leuven, Belgium

Abstract. This chapter presents an overview of the various pairings on elliptic curves used in pairing based cryptography and explains the necessary mathematical background. Several constructions of pairing friendly elliptic curves are also reviewed.

Keywords. Elliptic curve, Weil pairing, Tate pairing, eta pairing, ate pairing, pairing friendly curve

Fast algorithms to compute bilinear pairings are of central importance to pairing based cryptography. All these algorithms are based on Miller's algorithm [Mil86,Mil04] to compute the Weil and Tate pairings on elliptic curves. In recent years, a large number of papers [BKLS02,GHS02,DL03,Sco05b,ZZH06,BGhS07,HSV06,GHO$^+$07, MKHO07,ZZH07,LLP08,ZZH08,Ver08b,Hes08,ZZ08] have improved the efficiency of Miller's algorithm. This chapter will mainly focus on those papers that introduced new pairings such as the eta pairing [DL03,BGhS07] and all variants of the ate pairing [HSV06,GHO$^+$07,MKHO07,ZZH07,LLP08,Ver08b,Hes08]. All these pairings can be viewed as the restriction of a fixed power of the Tate pairing to a cleverly chosen domain.

1. Background on Elliptic Curves

1.1. Elliptic Curves, Torsion Subgroups and Frobenius

Let $\mathbb{F}_q$ denote a finite field with $q = p^m$ elements where p is prime. An elliptic curve $E/\mathbb{F}_q$ can be given by an equation of the form

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \qquad a_i \in \mathbb{F}_q\,,$$

together with the condition that the curve has no singular points. For every field K containing $\mathbb{F}_q$, $E(K)$ denotes the set of K-rational points on E and is defined as

$$E(K) = \{(x, y) \in K^2 \mid y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\} \cup \{\mathcal{O}\}\,,$$

[1]Postdoctoral Fellow of the Research Foundation - Flanders (FWO).

where $\mathcal{O}$ has projective coordinates $(0 : 1 : 0)$ and is called the point at infinity. The notation E is shorthand for $E(\overline{\mathbb{F}}_q)$ where $\overline{\mathbb{F}}_q$ is the algebraic closure of $\mathbb{F}_q$. One of the reasons why elliptic curves are useful in cryptography is the following fact: $E(K)$ is an abelian group using the "chord and tangent" addition law. The neutral element is the point at infinity $\mathcal{O}$, the opposite of $P_1 = (x_1, y_1)$ is the point $-P_1 = (x_1, -y_1 - a_1x_1 - a_3)$ and the sum of two non-opposite points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ is given by $P_3 = (x_3, y_3)$ with

$$x_3 = \lambda^2 + a_1\lambda - a_2 - x_1 - x_2 \quad \text{and} \quad y_3 = -(\lambda + a_1)x_3 - \nu - a_3$$

where $y = \lambda x + \nu$ is the line through P_1 and P_2 (or the tangent line if $P_1 = P_2$) with

$$\lambda = \begin{cases} (y_2 - y_1)/(x_2 - x_1) & \text{if } x_1 \neq x_2 \\ (3x_1^2 + 2a_2x_1 + a_4 - a_1y_1)/(2y_1 + a_1x_1 + a_3) & \text{if } x_1 = x_2 \end{cases}$$
$$\nu = \begin{cases} (y_1x_2 - y_2x_1)/(x_2 - x_1) & \text{if } x_1 \neq x_2 \\ (-x_1^3 + a_4x_1 + 2a_6 - a_3y_1)/(2y_1 + a_1x_1 + a_3) & \text{if } x_1 = x_2 \end{cases} .$$

Since $\mathbb{F}_{q^k}$ is finite, all groups $E(\mathbb{F}_{q^k})$ are finite and their orders satisfy the Hasse-Weil theorem.

Theorem II.1 (Hasse-Weil) *Let $E/\mathbb{F}_q$ be an elliptic curve defined over $\mathbb{F}_q$, then the order of the group $E(\mathbb{F}_{q^k})$ with $k \in \mathbb{N}_0$ satisfies*

$$\#E(\mathbb{F}_{q^k}) = q^k + 1 - t_k\,, \quad \textit{with} \quad |t_k| \leq 2\sqrt{q^k}\ .$$

Furthermore, if $X^2 - t_1X + q = (X - \alpha)(X - \beta)$ with $\alpha, \beta \in \mathbb{C}$, then $t_k = \alpha^k + \beta^k$.

For every $n \in \mathbb{N}$, let $[n]$ denote scalar multiplication by n, i.e. $[n] : P \mapsto [n]P := P + \cdots + P$ where there are n terms in the sum and define $[-n]P = -[n]P$. For any $n \in \mathbb{Z}_0$, the n-torsion subgroup of E, denoted $E[n]$ is the set of points of order dividing n in E, i.e. $E[n] = \{P \in E \mid [n]P = \mathcal{O}\}$. The structure of $E[n]$ for $p \nmid n$ is as follows:

$$E[n] \cong (\mathbb{Z}/n\mathbb{Z}) \times (\mathbb{Z}/n\mathbb{Z})\ .$$

Furthermore, for $n = p^e$ with $e \in \mathbb{N}_0$ we have either $E[p^e] \cong (\mathbb{Z}/p^e\mathbb{Z})$ or $E[p^e] = \{\mathcal{O}\}$. In the former case, the curve is called *ordinary*, and in the latter case *supersingular*.

Recall that $\alpha \in \overline{\mathbb{F}}_q$ belongs to $\mathbb{F}_q$ if and only if $\alpha^q = \alpha$. Furthermore, q-th powering is an automorphism on $\overline{\mathbb{F}}_q$, so if E is defined over $\mathbb{F}_q$, we obtain a well defined map

$$\pi_q : E \to E : (x, y) \mapsto (x^q, y^q) \quad \text{and} \quad \pi_q(\mathcal{O}) = \mathcal{O}\,,$$

called the Frobenius endomorphism. Since the addition law on the elliptic curve is algebraic, it is easy to verify that π_q is indeed an endomorphism. Furthermore, π_q satisfies $\pi_q^2 - [t] \circ \pi_q + [q] = 0$, with t the trace of Frobenius defined in Theorem II.1. For future reference, note that $E(\mathbb{F}_q)$ is precisely $\ker(\pi_q - [1])$.

Let r be a large prime with $r \mid \#E(\mathbb{F}_q)$ and $\gcd(r, q) = 1$, then we can easily analyze the action of Frobenius on $E[r]$: by assumption, at least one factor $\mathbb{Z}/r\mathbb{Z}$ is contained in $E(\mathbb{F}_q)$, which is an eigenspace of π_q with eigenvalue 1. By restricting the characteristic polynomial of π_q to $E[r]$, we conclude that the other eigenvalue is given by $q \bmod r$. First, we deal with the special case $r \mid (q-1)$ in which π_q has eigenvalue 1 with algebraic multiplicity 2. If π_q has 2 linearly independent eigenvectors, then π_q acts as the identity on $E[r]$ and thus $E[r] \subset E(\mathbb{F}_q)$. If π_q has only one eigenvector, then we can choose $V_2 \in E[r]$ linearly independent of the eigenvector and set $V_1 = (\pi_q - [1])V_2$ (which is an eigenvector). Note that $\pi_q^n(V_2) = [n]V_1 + V_2$ and thus the smallest $n \in \mathbb{N}_0$ with $\pi_q^n(V_2) = V_2$ is r. Since $E[r] = \langle V_1, V_2 \rangle$, we conclude that $E[r] \subset E(\mathbb{F}_{q^r})$. Assume now that $r \nmid (q-1)$, then π_q has two linearly independent eigenvectors V_1 and V_q corresponding to the eigenvalues 1 and q. Since $E[r] = \langle V_1, V_q \rangle$ and $\pi_q^n(V_q) = q^n V_q$, we have $E[r] \subset E(\mathbb{F}_{q^k})$ with k the smallest positive integer such that $q^k \equiv 1 \pmod{r}$.

Definition II.2 (Embedding Degree) Let E be an elliptic curve over $\mathbb{F}_q$ and $r \mid \#E(\mathbb{F}_q)$ with $\gcd(r, q) = 1$, then the embedding degree is the smallest positive integer k such that $r \mid (q^k - 1)$.

Note that k depends on both q and r and equals the order of q in $(\mathbb{Z}/r\mathbb{Z})^*$ and is therefore expected to be of size $\simeq r$. Only for specially constructed curves, called pairing friendly curves, will k be small. For prime r, the very definition of the embedding degree implies that $\Phi_k(q) \equiv 0 \pmod{r}$ with Φ_k the k-th cyclotomic polynomial. This follows directly from the factorization $x^k - 1 = \prod_{d|k} \Phi_d(x)$. The converse of this remark will be central in the construction of ordinary pairing friendly curves (see Proposition II.30).

In the remainder of the chapter we will use the following notation for the eigenspaces of Frobenius.

Notation II.3 Let E be an elliptic curve over $\mathbb{F}_q$ and r a large prime with $r \mid \#E(\mathbb{F}_q)$ and $\gcd(r, q) = 1$. Then

$$\mathbb{G}_1 = E[r] \cap \ker(\pi_q - [1]) \quad \text{and} \quad \mathbb{G}_2 = E[r] \cap \ker(\pi_q - [q]) \ .$$

1.2. Divisors and Miller Functions

Let C be a non-singular curve over $\mathbb{F}_q$, then a divisor D on C is a formal sum of points over the algebraic closure $\overline{\mathbb{F}}_q$

$$D = \sum_{P \in C(\overline{\mathbb{F}}_q)} c_P(P)$$

with only finitely many non-zero coefficients $c_P \in \mathbb{Z}$. The set of all divisors on C is denoted Div_C and clearly forms a group under formal addition. The degree of D is defined as $\deg(D) = \sum_{P \in C(\overline{\mathbb{F}}_q)} c_P$ and the subgroup of degree 0 divisors is denoted by Div_C^0. The support $\mathrm{Supp}(D)$ of a divisor D is the set of points P with $c_P \neq 0$ and we define $\mathrm{ord}_P(D) = c_P$.

Let π_q be the Frobenius morphism $\pi_q : C \to C$ given by $\pi_q(x, y) = (x^q, y^q)$ and define

$$\pi_q(D) = \sum_{P \in C(\mathbb{F}_q)} c_P(\pi_q(P)) \, ,$$

then D is called $\mathbb{F}_{q^k}$-rational if and only if $\pi_q^k(D) = D$. The set of $\mathbb{F}_{q^k}$-rational divisors is denoted by $\mathrm{Div}_C(\mathbb{F}_{q^k})$ and similarly for the degree 0 divisors.

To each non-constant rational function $f \in \overline{\mathbb{F}}_q(C)^*$, we can associate the divisor $\mathrm{div}(f) = \sum_{P \in C(\overline{\mathbb{F}}_q)} \mathrm{ord}_P(f)(P)$, where $\mathrm{ord}_P(f)$ denotes the order of vanishing of f at P. One can prove that only finitely many $\mathrm{ord}_P(f)$ are non-zero and furthermore, that $\deg(\mathrm{div}(f)) = 0$. Any divisor of the form $\mathrm{div}(f)$ with $f \in \overline{\mathbb{F}}_q(C)^*$ is called a principal divisor. The only functions $f \in \overline{\mathbb{F}}_q(C)^*$ with $\mathrm{div}(f) = 0$ are the constant functions, which shows that $\mathrm{div}(f)$ determines f up to a non-zero constant. Two divisors D and D' are called linearly equivalent, denoted $D \sim D'$ if $D' = D + \mathrm{div}(f)$ for some function f. Note that $\sim$ defines an equivalence relation on Div_C^0 and equivalence classes are called divisor classes. The quotient group $J_C := \mathrm{Div}_C^0 / \sim$ is called the Jacobian of C.

If C is an elliptic curve E, then $E \cong J_E$ by the following isomorphism $\phi : E \to \mathrm{Div}_C^0 : P \mapsto [(P) - (\mathcal{O})]$, where $[(P) - (\mathcal{O})]$ denotes the divisor class of $(P) - (\mathcal{O})$. This follows immediately from the definition of the group law on the elliptic curve: let $P_3 = P_1 + P_2$ with $P_i \in E$, then we have to show that there exists a function g with $(P_1) + (P_2) - 2(\mathcal{O}) = (P_3) - (\mathcal{O}) + \mathrm{div}(g)$. Let $l_{P_1,P_2} = y - \lambda x - \nu$ denote the line through P_1 and P_2 and $v_{P_3} = x - x_3$ the line through P_3 and $\mathcal{O}$, then $\mathrm{div}(l_{P_1,P_2}) = (P_1) + (P_2) + (-P_3) - 2(\mathcal{O})$ and $\mathrm{div}(v_{P_3}) = (P_3) + (-P_3) - 2(\mathcal{O})$, so we can take $g = l_{P_1,P_2}/v_{P_3}$. More general: if $D = \sum_{P \in E} c_P(P)$ is a degree zero divisor on E then $D \sim 0$ if and only if $\sum_{P \in E}[c_P]P = \mathcal{O}$ as a sum on E. As a consequence, we obtain a function f with $\mathrm{div}(f) = D$. A particular type of these functions plays a central role in the definition and computation of all pairings.

Definition II.4 Let $E/\mathbb{F}_q$ be an elliptic curve over $\mathbb{F}_q$, then any function $f \in \overline{\mathbb{F}}_q(E)$ with divisor $\mathrm{div}(f) = n(P) - ([n]P) - (n-1)(\mathcal{O})$ for some $n \in \mathbb{Z}$ and $P \in E$ is called a Miller function (also called Weil function) and will be denoted as $f_{n,P}$.

Let f be a function on C and D a divisor on C with $\mathrm{Supp}(D) \cap \mathrm{Supp}(\mathrm{div}(f)) = \emptyset$, then by definition we set

$$f(D) = \prod_{P \in C} f(P)^{\mathrm{ord}_P(D)} \ .$$

Note that if D has degree zero, then $f(D)$ only depends on $\mathrm{div}(f)$ and not on f itself, since $\mathrm{div}(f)$ determines f up to a non-zero constant. Furthermore, if we take $D = \mathrm{div}(g)$ for some function g on C, we have the following theorem, which will be used frequently to prove the main properties of the Weil and Tate pairings.

Theorem II.5 (Weil Reciprocity) *Let C be a curve and f, g non-zero functions on C with* $\mathrm{Supp}(\mathrm{div}(f)) \cap \mathrm{Supp}(\mathrm{div}(g)) = \emptyset$, *then* $f(\mathrm{div}(g)) = g(\mathrm{div}(f))$.

Miller [Mil86,Mil04] described an efficient algorithm to compute the functions $f_{n,P}$ and their evaluations at a given divisor D. The crucial ingredient is the following observation that is easily proved by taking divisors of both sides: let i and j be positive integers, then

$$f_{i+j,P} = f_{i,P} \cdot f_{j,P} \cdot \frac{l_{[i]P,[j]P}}{v_{[i+j]P}} \ , \tag{1}$$

where $l_{[i]P,[j]P}$ is the equation of the line through $[i]P$ and $[j]P$ (or the tangent line when $[i]P = [j]P$) and $v_{[i+j]P}$ the equation of the vertical line trough $[i+j]P$. For $n < 0$ it suffices to remark that $\mathrm{div}(f_{n,P}) = -\mathrm{div}(f_{-n,P}) - \mathrm{div}(v_{[n]P})$ with $v_{[n]P}$ the vertical line through $[n]P$, so we can take $f_{n,P} = 1/(f_{-n,P} v_{[n]P})$.

The equality (1) then immediately leads to Algorithm II.1 by a simple double and add procedure.

Algorithm II.1 Miller's algorithm for elliptic curves

```
Input: $n \in \mathbb{N}$, $P \in E[r]$ and divisor $D$ with $\mathrm{Supp}(D) \cap \{P, \mathcal{O}\} = \emptyset$.
Output: $f_{n,P}(D)$.
1: Write $n = \sum_{j=0}^{L} n_j 2^j$, with $n_j \in \{0,1\}$ and $n_L = 1$
2: $T \leftarrow P$ ; $f \leftarrow 1$
3: for $j$ from $L - 1$ downto 0 do
4:     $f \leftarrow c^2 \cdot l_{T,T}(D)/v_{[2]T}(D)$
5:     $T \leftarrow [2]T$
6:     if $n_j = 1$ then
7:         $f \leftarrow f \cdot l_{T,P}(D)/v_{T \oplus P}(D)$
8:         $T \leftarrow T \oplus P$
9: return $f$.
```

1.3. Supersingular Elliptic Curves and Distortion Maps

Recall that an elliptic curve E over a finite field $\mathbb{F}_q$ of characteristic p is called supersingular if and only if $E[p] = \{\mathcal{O}\}$. In fact, we have the following equivalent characterizations due to Deuring [Deu41].

Proposition II.6 *Let E be an elliptic curve over $\mathbb{F}_q$ with $q = p^m$ and p prime, then E is supersingular if and only if any of the following equivalent conditions hold:*

1. $E[p] = \{\mathcal{O}\}$.
2. $p \mid t$ *with* $\#E(\mathbb{F}_q) = q + 1 - t$.
3. $\mathrm{End}(E)$ *is an order in a quaternion algebra, and thus non-commutative.*
4. *There exists an integer k such that* $\pi_q^k = [\pm q^{k/2}]$.

Since the group orders of supersingular elliptic curves are so restricted, it is possible to list all of them, together with the corresponding embedding degree and group structure. Table II.1 taken from [Gal05a, Theorem IX.20] summarizes all possibilities.

Table II.1. Group orders, embedding degree and group structure of supersingular elliptic curves.

q	k	$\#E(\mathbb{F}_q)$	Group structure of $E(\mathbb{F}_{q^k})$
p^{2m}	1	$q \pm 2\sqrt{q} + 1$	$(\mathbb{Z}/(\sqrt{q} \pm 1)\mathbb{Z})^2$
p^{2m+1} or p^{2m} and $p \not\equiv 1 \pmod 4$	2	$q + 1$	$(\mathbb{Z}/(q+1)\mathbb{Z})^2$
p^{2m} and $p \not\equiv 1 \pmod 3$	3	$q \pm \sqrt{q} + 1$	$(\mathbb{Z}/(q^{3/2} \mp 1)\mathbb{Z})^2$
2^{2m+1}	4	$q \pm \sqrt{2q} + 1$	$(\mathbb{Z}/(q^2 + 1)\mathbb{Z})^2$
3^{2m+1}	6	$q \pm \sqrt{3q} + 1$	$(\mathbb{Z}/(q^3 + 1)\mathbb{Z})^2$

The two main advantages of supersingular elliptic curves are that they are easy to construct and admit non-rational endomorphisms, i.e. which are not defined over the field of definition of the curve. Such non-rational endomorphisms or *distortion maps* are useful in that for $k > 1$ they map $\mathbb{G}_1$ onto a subgroup of $E[r]$ which is not $\mathbb{F}_q$-rational, e.g. $\mathbb{G}_2$ and thus linearly independent of $\mathbb{G}_1$. This will enable to define a non-degenerate pairing on $\mathbb{G}_1 \times \mathbb{G}_1$. It is easy to see that such distortion maps can only exist for supersingular curves: if E is an ordinary elliptic curve defined over $\mathbb{F}_q$, then any endomorphism $\phi \in \mathrm{End}(E)$ will be $\mathbb{F}_q$-rational since $\phi \circ \pi_q = \pi_q \circ \phi$ by the commutativity of $\mathrm{End}(E)$. The most frequently used supersingular elliptic curves are as follows.

1.3.1. Embedding degree $k = 2$

Constructing supersingular elliptic curves with $k = 2$ is easy: given a prime r, choose a cofactor h such that $p = hr - 1$ is prime.

- If $p \equiv 2 \pmod 3$, then the curve $E : y^2 = x^3 + a$ with $a \in \mathbb{F}_p^*$ is supersingular and has $p+1 = hr$ points. Furthermore, the endomorphism $\phi : (x, y) \mapsto (\xi_3 x, y)$ with $\xi_3^3 = 1$ is a distortion map.
- If $p \equiv 3 \pmod 4$, then $E : y^2 = x^3 + ax$ is supersingular with $p + 1 = hr$ points. The endomorphism $\phi : (x, y) \mapsto (-x, iy)$ with $i^2 = -1$ can be used as a distortion map.

1.3.2. Embedding degree $k = 4$

From Table II.1, embedding degree $k = 4$ only occurs over finite fields of characteristic two with odd extension degree. Let $q = 2^m$ with m odd, then the curves

$$E_0 : y^2 + y = x^3 + x \quad \text{and} \quad E_1 : y^2 + y = x^3 + x + 1\,,$$

are supersingular and have $t = \pm\sqrt{2q}$. Let $\alpha \in \mathbb{F}_{2^2}$ and $\beta \in \mathbb{F}_{2^4}$ satisfy $\alpha^2 + \alpha + 1 = 0$ and $\beta^2 + (\alpha + 1)\beta + 1 = 0$, then the endomorphism

$$\phi : (x, y) \mapsto (\alpha^2 x + \beta^2, y + \alpha^2 \beta x + \beta)$$

is a distortion map.

1.3.3. Embedding degree $k = 6$

From Table II.1, embedding degree $k = 6$ only occurs over finite fields of characteristic three with odd extension degree. Let $q = 3^m$ with m odd, then the curves

$$E_1 : y^2 = x^3 - x + 1 \quad \text{and} \quad E_{-1} : y^2 = x^3 - x - 1\,,$$

are supersingular and have $t = \pm\sqrt{3q}$. Let $i \in \mathbb{F}_{3^2}$ and $\alpha \in \mathbb{F}_{3^3}$ satisfy $i^2 = -1$ and $\alpha^3 - \alpha - (\pm 1) = 0$ depending on the equation of the curve, then the endomorphism

$$\phi : (x, y) \mapsto (\alpha - x, iy)$$

is a distortion map.

1.4. Twists of Elliptic Curves

Twists play a central role in efficient implementations of pairings on ordinary elliptic curves. In this section we present the necessary background on twists of elliptic curves over finite fields of characteristic $p \geq 5$. The proofs of the theorems contained in this section can be found in [Sil92,HSV06].

Definition II.7 Let E and E' be two elliptic curves over $\mathbb{F}_q$, then E' is called a twist of degree d of E if there exists an isomorphism $\phi_d : E' \to E$ defined over $\mathbb{F}_{q^d}$ and d is minimal.

Although not immediate from the above definition, there are only a very limited number of possible degrees d. To see why, let σ be the q-th power Frobenius automorphism of $\overline{\mathbb{F}}_q$, then $\phi_d^\sigma \circ \phi_d^{-1} \in \operatorname{Aut}(E)$, with ϕ_d^σ the isomorphism obtained by applying σ to the coefficients of ϕ_d. Furthermore, since d was chosen minimal, the order of this automorphism is d. So, if E' is a degree d twist of E, then $\operatorname{Aut}(E)$ must contain an element of order d. However, $\operatorname{Aut}(E)$ always is a finite group of order dividing 24 [Sil92, Theorem III.10.1] and if $p \geq 5$, we have $\# \operatorname{Aut}(E) \in \{2, 4, 6\}$. So for $p \geq 5$, only $d = 2, 3, 4, 6$ are possible. Furthermore, all twists can be described explicitly as in [Sil92, Proposition X.5.4].

Proposition II.8 *Assume that $p \geq 5$, then the set of twists of E is canonically isomorphic with $\mathbb{F}_q^*/(\mathbb{F}_q^*)^d$ with $d = 2$ if $j(E) \neq 0, 1728$, $d = 4$ if $j(E) = 1728$ and $d = 6$ if $j(E) = 0$.*

Note that in the above cases we have $\operatorname{Aut}(E) \cong \mu_d$, with μ_d the d-th roots of unity. If we assume that E is given by a short Weierstrass equation $E : y^2 = x^3 + ax + b$ with $a, b \in \mathbb{F}_q$, then an isomorphism is given by

$$[\cdot] : \mu_d \to \operatorname{Aut}(E) : \xi \mapsto [\xi] \quad \text{with} \quad [\xi](x, y) = (\xi^2 x, \xi^3 y) \ .$$

Also it is rather easy to find an equation for the twists given an equation for E. Let $D \in \mathbb{F}_q^*$, then the twists corresponding to $D \bmod (\mathbb{F}_q^*)^d$ are given by

$$\begin{array}{ll} d = 2 \quad : y^2 = x^3 + a/D^2 x + b/D^3, & \phi_d : E' \to E : (x, y) \mapsto (Dx, D^{3/2} y) \\ d = 4 \quad : y^2 = x^3 + a/Dx, & \phi_d : E' \to E : (x, y) \mapsto (D^{1/2} x, D^{3/4} y) \\ d = 3, 6 : y^2 = x^3 + b/D, & \phi_d : E' \to E : (x, y) \mapsto (D^{1/3} x, D^{1/2} y) \end{array}$$

An alternative representation of the quadratic twists, which may be more convenient from an implementation perspective, is given by

$$d = 2 : \ Dy^2 = x^3 + ax + b, \quad \phi_d : E' \to E : (x, y) \mapsto (x, D^{1/2} y) \ .$$

Note that if we want to construct a twist of degree d, then $\mathbb{F}_q^*/(\mathbb{F}_q^*)^d$ has to have d elements or equivalently, $q \equiv 1 \pmod d$. The group orders of twists are also easily computed as shown in the following proposition.

Proposition II.9 *Let E be an ordinary elliptic curve over $\mathbb{F}_q$ with $\#E(\mathbb{F}_q) = q + 1 - t$, admitting a twist E' of degree d, then the possible group orders of $E'(\mathbb{F}_q)$ are given by the following:*

$$\begin{array}{lll}
\underline{d=2}: & \#E'(\mathbb{F}_q) = q + 1 + t & \\
d=3: & \#E'(\mathbb{F}_q) = q + 1 - (3f - t)/2 & \textit{with } t^2 - 4q = -3f^2 \\
 & \#E'(\mathbb{F}_q) = q + 1 - (-3f - t)/2 & \textit{with } t^2 - 4q = -3f^2 \\
d=4: & \#E'(\mathbb{F}_q) = q + 1 + f & \textit{with } t^2 - 4q = -f^2 \\
 & \#E'(\mathbb{F}_q) = q + 1 - f & \textit{with } t^2 - 4q = -f^2 \\
d=6: & \#E'(\mathbb{F}_q) = q + 1 - (-3f + t)/2 & \textit{with } t^2 - 4q = -3f^2 \\
 & \#E'(\mathbb{F}_q) = q + 1 - (3f + t)/2 & \textit{with } t^2 - 4q = -3f^2 .
\end{array}$$

Twists are useful since they allow an efficient representation of the group $\mathbb{G}_2$ for $k > 1$, which follows from the following theorem. The notation $a \| b$ means that $a \mid b$ and $a^2 \nmid b$.

Theorem II.10 *Let E be an ordinary elliptic curve over $\mathbb{F}_q$ admitting a twist of degree d and let E_i for $i = 0, \ldots, d-1$ be the twists of E of degree dividing d. Assume that $r > 6$ satisfies $r \| \#E(\mathbb{F}_q)$ and $r^2 \| \#E(\mathbb{F}_{q^d})$, then there exists a unique twist E_i of degree d such that $r \| \#E_i(\mathbb{F}_q)$.*

Assume that E admits a twist of degree d and let $z = \gcd(k, d)$ and $e = k/z$. Since $k > 1$ is the minimal value such that r divides $q^k - 1$, we know that the group $E(\mathbb{F}_{q^e})$ has order divisible by r, but not r^2. Furthermore, since $r > 6$, we know that there is a unique degree z twist E' of E over $\mathbb{F}_{q^e}$ such that $r \mid \#E'(\mathbb{F}_{q^e})$. The following lemma provides an efficient representation for $\mathbb{G}_2$.

Lemma II.11 *Let $\mathbb{G}_2'$ be the unique subgroup of order r of $E'(\mathbb{F}_{q^e})$ and denote $\phi_z : E' \to E$ the twisting isomorphism, then*

$$\mathbb{G}_2 = \phi_z(\mathbb{G}_2') \ .$$

2. The Weil Pairing

The Weil pairing was introduced by Weil [Wei40] in 1940 in his proof of the Riemann hypothesis for curves. Although there exist two equivalent definitions of the Weil pairing (see [How96] for a proof of equivalence), we follow the original definition, which is also more amenable to efficient computation.

Definition II.12 (Weil Pairing) Let E be an elliptic curve over $\mathbb{F}_q$, $r \in \mathbb{N}_0$, and take $m \in \mathbb{N}_0$ such that $E[r] \subset E(\mathbb{F}_{q^m})$, then the Weil pairing w_r is a map

$$w_r : E[r] \times E[r] \to \mu_r \subset \mathbb{F}_{q^m} : (P, Q) \mapsto w_r(P, Q) = \frac{f(D_Q)}{g(D_P)}$$

where $D_P \sim (P) - (\mathcal{O})$, $D_Q \sim (Q) - (\mathcal{O})$ have disjoint support and $\operatorname{div}(f) = rD_P$, $\operatorname{div}(g) = rD_Q$.

We first show that the above definition is well defined by proving independence of the choices made in the divisors D_P and D_Q. If D'_P is another divisor linearly equivalent with $(P)-(\mathcal{O})$, then by definition there exists a function h such that $D'_P = D_P + \mathrm{div}(h)$. If we set $f' = fh^r$, then $\mathrm{div}(f') = rD'_P$ and thus

$$\frac{f'(D_Q)}{g(D'_P)} = \frac{f(D_Q)h^r(D_Q)}{g(D_P)g(\mathrm{div}(h))} = \frac{f(D_Q)h^r(D_Q)}{g(D_p)h(\mathrm{div}(g))} = \frac{f(D_Q)}{g(D_P)},$$

where we used Weil reciprocity in the second equality. A similar reasoning shows independence of the choices made in D_Q. Proving that w_r maps into μ_r is equally easy:

$$w_r(P,Q)^r = \frac{f(rD_Q)}{g(rD_P)} = \frac{f(\mathrm{div}(g))}{g(\mathrm{div}(f))} = 1,$$

by Weil reciprocity. The following theorem summarizes other useful properties.

Theorem II.13 *The Weil pairing $w_r : E[r] \times E[r] \to \mu_r$ satisfies the following properties:*

– Bilinearity: *If $P, Q, R \in E[r]$, then*

$$w_r(P+R,Q) = w_r(P,Q)w_r(R,Q)$$
$$w_r(P,Q+R) = w_r(P,Q)w_r(P,R)$$

– Alternating: *If $P \in E[r]$, then $w_r(P,P) = 1$, which implies for $Q \in E[r]$ $w_r(P,Q) = w_r(Q,P)^{-1}$ by linearity.*
– Non-degeneracy: *If $P \in E[r] \setminus \mathcal{O}$, then there exists $Q \in E[r]$ such that $w_r(P,Q) \neq 1$.*
– Compatibility: *If $P \in E[rs]$ and $Q \in E[r]$, then $w_{rs}(P,Q) = w_r([s]P,Q)$.*
– Galois invariance: *For all $\sigma \in \mathrm{Gal}(\overline{\mathbb{F}}_q/\mathbb{F}_q)$ and $P, Q \in E[r]$*

$$w_r(\sigma(P),\sigma(Q)) = \sigma(w_r(P,Q)) .$$

Most properties in the above theorem follow easily from the definition and Weil reciprocity. Only non-degeneracy is non-trivial and a proof can be found in [Sil92] and [Hes04].

In the definition of the Weil pairing we need an extension of degree m such that $E[n] \subset \mathbb{F}_{q^m}$. From the analysis given in Section 1.1 follows that if r is a large prime with $\gcd(r,q) = 1$ and $r \mid \#E(\mathbb{F}_q)$, then m can be taken as follows: if $k > 1$, then $m = k$ (the embedding degree), if $k = 1$ and $E[r] \subset E(\mathbb{F}_q)$ then $m = 1$, else $m = r$.

An immediate consequence of Theorem II.13 is that the Weil pairing can be used to determine linear dependence of two points on an elliptic curve. Let $P, Q \in E[r]$, then Q lies in the subgroup generated by P if and only if $w_r(P,Q) = 1$. This property can be used to determine the group structure of an elliptic curve as described in [Mil04, Section 6].

In practice, working with Definition II.12 is not very efficient since we need to evaluate the functions f and g at two points each. Recall the Miller functions $f_{n,P}$ with divisor $\mathrm{div}(f_{n,P}) = n(P) - ([n]P) - (n-1)(\mathcal{O})$, then we have the following proposition, a proof of which is given in [Mil04].

Proposition II.14 *Let E be an elliptic curve over $\mathbb{F}_q$ and let $P, Q \in E[r]$ with $P \neq Q$, then*

$$w_r(P, Q) = (-1)^r \frac{f_{r,P}(Q)}{f_{r,Q}(P)} \ .$$

3. The Tate Pairing

The Tate pairing was introduced by Tate [Tat95] as a very general pairing on abelian varieties over local fields relating the Mordell-Weil group, the first cohomology group and the Brauer group. Lichtenbaum [Lic69] applied this to the special case of Jacobians of curves (still over local fields) and Frey and Rück [FR94,FMR99] defined the pairing over finite fields. For elliptic curves the definition is as follows.

Definition II.15 (Tate Pairing) Let E be an elliptic curve over $\mathbb{F}_q$ and $r \mid \#E(\mathbb{F}_q)$ with $\gcd(r, q) = 1$, and let k be the embedding degree, then the Tate pairing $\hat{t}_r$ is the map

$$\begin{aligned}\hat{t}_r : E(\mathbb{F}_{q^k})[r] \times E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k}) \rightarrow \mathbb{F}_{q^k}^*/(\mathbb{F}_{q^k}^*)^r :& \\ (P, Q) \mapsto \hat{t}_r(P, Q) = f_{r,P}(D_Q) \,,&\end{aligned}$$

where $\operatorname{div}(f_r) = r(P) - r(\mathcal{O})$ and $D_Q \sim (Q) - (\mathcal{O})$ coprime with $\operatorname{div}(f_r)$.

In the above definition the point Q should be considered as a representative of the whole equivalence class $Q + rE(\mathbb{F}_{q^k})$ and similarly, the value of the Tate pairing really is the class $f_{r,P}(D_Q)(\mathbb{F}_{q^k}^*)^r$. As for the Weil pairing, we need to show that the definition makes sense, i.e. is independent of the choices made for the representative of the class $Q + rE(\mathbb{F}_{q^k})$ and the divisor D_Q. Both follow easily from Weil reciprocity as follows: for any divisor $D_Q' = D_Q + \operatorname{div}(h)$ coprime with $\operatorname{div}(f_{r,P})$ we have

$$f_{r,P}(D_Q') = f_{r,P}(D_Q) f_{r,P}(\operatorname{div}(h)) \,,$$

and by Weil reciprocity $f_{r,P}(\operatorname{div}(h)) = h(r(P) - r(\mathcal{O})) = (h(P)/h(\mathcal{O}))^r \in (\mathbb{F}_{q^k}^*)^r$. Similarly, if we replace Q by $Q + rR$ for $R \in E(\mathbb{F}_{q^k})$, then

$$D_{Q+rR} \sim (Q + rR) - (\mathcal{O}) \sim (Q) + r(R) - (r-1)(\mathcal{O}) \sim D_Q + r(R) - r(\mathcal{O}) \,,$$

where the second equivalence follows from the definition of the elliptic curve group law. Finally, this implies $f_{r,P}(D_{Q+rR}) = f_{r,P}(D_Q) f((R) - (\mathcal{O}))^r$, which proves independence of the choice of representative.

If r is prime, then by definition of the embedding degree we have $\gcd(r, q^d - 1) = 1$ for all positive integers $d \mid k$ and $d < k$. This shows that all elements of the fields $\mathbb{F}_{q^d}$ are r-th powers. In particular, if $k > 1$ and if both P and Q are chosen to have coordinates in a strict subfield of $\mathbb{F}_{q^k}$ then $\hat{t}_r(P, Q) = 1$, which shows that at least one of the input points has to be defined over $\mathbb{F}_{q^k}$. In practice, one will often choose $P \in E(\mathbb{F}_q)[r]$ and Q necessarily in $E(\mathbb{F}_{q^k}) \setminus \cup_{d|k, d<k} E(\mathbb{F}_{q^d})$.

The Tate pairing satisfies similar properties as the Weil pairing.

Theorem II.16 *The Tate pairing* $\hat{t}_r : E(\mathbb{F}_{q^k})[r] \times E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k}) \to \mathbb{F}_{q^k}^*/(\mathbb{F}_{q^k}^*)^r$ *satisfies the following properties:*

– Bilinearity: *For all* $P, R \in E(\mathbb{F}_{q^k})[r]$ *and* $Q, S \in E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k})$

$$\begin{aligned} \hat{t}_r(P+R,Q) &= \hat{t}_r(P,Q)\hat{t}_r(R,Q) \\ \hat{t}_r(P,Q+S) &= \hat{t}_r(P,Q)\hat{t}_r(P,S) \end{aligned} .$$

– Non-degeneracy: *If* $P \in E(\mathbb{F}_{q^k})[r] \setminus \{\mathcal{O}\}$*, then there exists a* $Q \in E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k})$ *such that* $\hat{t}_r(P,Q) \neq 1$*. Similarly, if* $Q \in E(\mathbb{F}_{q^k}) \setminus rE(\mathbb{F}_{q^k})$*, then there exists a* $P \in E(\mathbb{F}_{q^k})[r]$ *with* $\hat{t}_r(P,Q) \neq 1$.

– Galois invariance: *For all* $\sigma \in \mathrm{Gal}(\overline{\mathbb{F}}_q/\mathbb{F}_q)$*,* $P \in E(\mathbb{F}_{q^k})[r]$*,* $Q \in E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k})$*:*

$$\hat{t}_r(\sigma(P),\sigma(Q)) = \sigma(\hat{t}_r(P,Q)) \ .$$

As for the Weil pairing, we can avoid working with the divisor D_Q and simply evaluate at Q. For this simplification to hold, we need that the function $f_{r,P}$ is properly normalized. To define what this means, let $u_{\mathcal{O}}$ be a fixed $\mathbb{F}_q$-rational uniformizer at $\mathcal{O}$, then for any function $f \in \overline{\mathbb{F}}_q(E)^*$ we define $\mathrm{lc}_{\mathcal{O}}(f)$ to be the leading coefficient of f as a Laurent series in $u_{\mathcal{O}}$. Note that when f is defined at $\mathcal{O}$ we simply have $f(\mathcal{O}) = \mathrm{lc}_{\mathcal{O}}(f)$ independent of the uniformizer chosen. The following lemma (see [GHO$^+$07] for a proof) shows that when $f_{r,P}$ is properly normalized, it suffices to evaluate at Q.

Lemma II.17 *Let* $P \in E(\mathbb{F}_{q^k})[r]$ *and* $Q \in E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k})$ *with* $P \neq Q$*, then*

$$\hat{t}_r(P,Q) = f_{r,P}(Q) \cdot (\mathbb{F}_{q^k}^*)^r ,$$

if and only if $\mathrm{lc}_{\mathcal{O}}(f_{r,P}) \in (\mathbb{F}_{q^k}^*)^r$*. Furthermore,* $\mathrm{lc}_{\mathcal{O}}(f_{r,P})$ *being an* r*-th power is independent of the uniformizer chosen.*

Note that for $k > 1$ and $P \in E(\mathbb{F}_q)[r]$, any $\mathbb{F}_q$-rational Miller function $f_{r,P}$ will be automatically normalized since all elements in $\mathbb{F}_q$ are r-th powers, so also $\mathrm{lc}_{\mathcal{O}}(f_{r,P})$.

In practice, we often need a unique representative for the value of the Tate pairing, and not a whole equivalence class, which justifies the following definition.

Definition II.18 (The Reduced Tate Pairing) Let E be an elliptic curve over $\mathbb{F}_q$ with $r \mid \#E(\mathbb{F}_q)$ and $\gcd(r,q) = 1$, and let k be the embedding degree, then the reduced Tate pairing t_r is the map

$$\begin{aligned} t_r : E(\mathbb{F}_{q^k})[r] \times E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k}) &\to \mu_r \subset \mathbb{F}_{q^k}^* : \\ (P,Q) &\mapsto t_r(P,Q) = \hat{t}_r(P,Q)^{(q^k-1)/r} \ . \end{aligned}$$

An interesting compatibility property of the reduced Tate pairing is the following: let $r \mid N \mid (q^k - 1)$, then for $P \in E(\mathbb{F}_{q^k})[r]$ and $Q \in E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k})$ we have the equality $t_r(P,Q) = t_N(P,Q)$.

Finally, in many cases $E(\mathbb{F}_{q^k})[r]$ and $E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k})$ are isomorphic. Indeed, as long as $E(\mathbb{F}_{q^k})[r] \cap rE(\mathbb{F}_{q^k}) = \{\mathcal{O}\}$, which is satisfied if $E(\mathbb{F}_{q^k})$ does not contain points of order r^2, we have the following isomorphism

$$\iota : E(\mathbb{F}_{q^k})[r] \to E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k}) : Q \mapsto Q + rE(\mathbb{F}_{q^k}) \ .$$

Remark II.19 To speed up the computation of the Tate pairing, one will often restrict the first argument P to $E(\mathbb{F}_q)$. Miller's algorithm corresponds to a scalar multiplication of the point P, which is obviously faster if P is defined over $\mathbb{F}_q$ than over some extension field $\mathbb{F}_{q^k}$.

4. The Eta and Ate Pairings

The eta and ate pairings are best viewed as efficient methods to compute a power of the reduced Tate pairing when restricted to the eigenspaces of Frobenius: $\mathbb{G}_1 = E[r] \cap \ker(\pi_q - [1])$ and $\mathbb{G}_2 = E[r] \cap \ker(\pi_q - [q])$.

The basic idea of all these pairings is the following lemma.

Lemma II.20 *Let E be an elliptic curve over $\mathbb{F}_q$ and let $r \mid \#E(\mathbb{F}_q)$, with* $\gcd(r, q) = 1$ *and embedding degree k. Let $\lambda = Cr$ be a multiple of r, then the following map*

$$a_\lambda : E(\mathbb{F}_{q^k})[r] \times E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k}) \to \mu_r \subset \mathbb{F}_{q^k}^* :$$
$$(P, Q) \mapsto a_\lambda(P, Q) = f_{\lambda,P}(Q)^{(q^k-1)/r} ,$$

with $f_{\lambda,P}$ normalized, defines a bilinear pairing which is non-degenerate if and only if $\gcd(r, C) = 1$.

PROOF: By taking divisors of both sides, it is easy to verify that $f_{\lambda,P}$ can be taken as

$$f_{\lambda,P} = f_{Cr,P} = f_{r,P}^C \cdot f_{C,[r]P} \ .$$

Since $[r]P = \mathcal{O}$, we have $f_{C,[r]P} = 1$. By taking the C-th power of the reduced Tate pairing we thus obtain

$$t_r(P, Q)^C = f_{r,P}(P)^{C(q^k-1)/r} = a_\lambda(P, Q) \ .$$

Furthermore, since t_r has order r and is non-degenerate, we conclude that a_λ is non-degenerate if and only if $\gcd(r, C) = 1$. □

Remark II.21 The above lemma can be modified as follows: let $N = \gcd(q^k - 1, \lambda)$ and $C = \lambda/N$, then $f_{\lambda,Q}^{(q^k-1)/N}$ defines a bilinear pairing which is non-degenerate if and only if $\gcd(r, C) = 1$. The advantage of this formulation is that for particular choices of N, the final exponentiation $(q^k - 1)/N$ has low Hamming weight in base q, which can be used to speed up the computation.

The eta and ate pairings are then simply obtained by choosing special λ and exploiting the action of an efficiently computable endomorphism, such as Frobenius, on $\mathbb{G}_1$ and $\mathbb{G}_2$. Although the eta pairing [BGhS07] was introduced before the ate pairing [HSV06], we will first derive the ate pairing and obtain the eta pairing as a special case.

4.1. Ate Pairings on Ordinary Elliptic Curves

All ate pairings on ordinary elliptic curves are derived from the Tate pairing by restricting to $\mathbb{G}_2 \times \mathbb{G}_1$ and exploiting $\pi_q(Q) = [q]Q$ for $Q \in \mathbb{G}_2$ and $\pi_q(P) = P$ for $P \in \mathbb{G}_1$. By definition of the embedding degree we have $r \mid (q^k - 1)$ and thus $r \mid (S^k - 1)$ for any integer $S \equiv q \pmod{r}$. This leads to the following theorem.

Theorem II.22 (Ate Pairing I) *Let S be any integer with $S \equiv q \pmod{r}$. Let $\lambda = S^k - 1$, and $C = \lambda/r$, then*

$$a_S : \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r \subset \mathbb{F}_{q^k}^* : (Q, P) \mapsto a_S(Q, P) = f_{S,Q}(P)^{(q^k-1)/r} ,$$

with $f_{S,Q}$ normalized defines a bilinear pairing which is non-degenerate if and only if $\gcd(r, C) = 1$.

PROOF: By Lemma II.20 we already know that a_λ defines a bilinear pairing which is non-degenerate if and only if $\gcd(r, C) = 1$. Since $r \mid \lambda$ we have $f_{\lambda,Q} = f_{S^k,Q}$ and a repeated application of the formula $f_{ab,Q} = f_{b,Q}^a f_{a,[b]Q}$ shows that

$$f_{S^k,Q} = f_{S,Q}^{S^{k-1}} \cdot f_{S,[S]Q}^{S^{k-2}} \cdots f_{S,[S^{k-1}]Q} \; .$$

Now $S \equiv q \pmod{r}$ and thus $[S^i]Q = [q^i]Q = \pi_q^i(Q)$ since $Q \in \mathbb{G}_2$. Furthermore, since π_q is purely inseparable of degree q, we have the following equality (see [HSV06, Lemma 1])

$$f_{S,\pi_q^i(Q)} \circ \pi_q^i = f_{S,Q}^{q^i} \; .$$

Since $\pi_q(P) = P$, we thus have $f_{S,\pi_q^i(Q)}(P) = f_{S,Q}(P)^{q^i}$, which gives the expression

$$f_{\lambda,Q}(P) = f_{S,Q}(P)^{\sum_{i=0}^{k-1} S^{k-1-i} q^i} \; .$$

Raising both sides to the power $(q^k - 1)/r$ leads to a non-degenerate bilinear pairing if and only if $\gcd(r, C) = 1$. Finally, note that $\sum_{i=0}^{k-1} S^{k-1-i} q^i \equiv kq^{k-1} \pmod{r}$ and $\gcd(kq^{k-1}, r) = 1$ since $k \mid (r - 1)$ and $\gcd(r, q) = 1$, so we can invert the powering by kq^{k-1} since $f_{S,Q}(P)^{(q^k-1)/r}$ is an r-th root of unity. $\square$

Remark II.23 Similarly to Lemma II.20, Theorem II.22 can be adapted using a different final exponentiation as follows: let $N = \gcd(q^k - 1, \lambda)$ and $C = \lambda/N$, then $f_{S,Q}^{c_S(q^k-1)/N}$ with $c_S = \gcd(N, \sum_{i=0}^{k-1} S^{k-1-i} q^i)$ defines a bilinear pairing which is non-degenerate if and only if $\gcd(r, C) = 1$.

Theorem II.22 can be extended immediately by noting that $r \mid ((q^i)^k - 1)$ for all positive integers i, so the condition on S in the theorem can be relaxed to $S \equiv q^i \pmod{r}$. The main advantage of the ate pairing over the Tate pairing is that S can be much smaller than r and thus leads to a much shorter loop in Miller's algorithm. However, this gain is offset completely by the fact that Q is defined over $\mathbb{F}_{q^k}$ and not over $\mathbb{F}_q$. Luckily, it is possible to represent $\mathbb{G}_2$ by a subgroup of a twist as in Lemma II.11, which is defined over a small extension of $\mathbb{F}_q$. The net effect is that the ate pairing for small S and using twists will be faster than the corresponding Tate pairing.

The above automatically leads to the question of how small S can be. Note that

$$r \mid \Phi_{k/d}(q^i) \quad \text{where } d = \gcd(i,k)\,,$$

which implies that the minimal value for $q^i \bmod r$ is $r^{1/\varphi(k/d)}$. For $\gcd(i,k) = 1$ we therefore obtain the smallest lower bound of roughly $r^{1/\varphi(k)}$. This optimal bound is attained for some families of pairing friendly curves, but not in general.

Theorem II.22 is a rather limited application of Lemma II.20 in that we only considered obvious multiples of r of the form $S^k - 1$ for well chosen S. By considering all possible multiples of r and exploiting the fact that the product and thus also division of two pairings (since the pairing has order r) is again a pairing, we obtain the following construction [Ver08b,Hes08]. For $h = \sum_{i=0}^{d} h_i z^i \in \mathbb{Z}[z]$ with $h(S) \equiv 0 \bmod r$, let $f_{S,h,Q} \in \mathbb{F}_{q^k}(E)$ denote the normalised function with divisor

$$\operatorname{div}(f_{S,h,Q}) = \sum_{i=0}^{d} h_i((S^iQ) - (\mathcal{O}))$$

and let $||h||_1 = \sum_{i=0}^{d} |h_i|$. The following theorem which generalises Theorem II.22 was proven in [Hes08].

Theorem II.24 (Ate Pairing II) *Assume that S is a primitive k-th root of unity modulo r^2, then*

$$a_{S,h} : \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r \subset \mathbb{F}_{q^k}^* : (Q,P) \mapsto (f_{S,h,Q}(P))^{(q^k-1)/r}$$

defines a bilinear pairing which is non-degenerate if and only if $h(S) \not\equiv 0 \bmod r^2$. The relation with the reduced Tate pairing is $a_{S,h}(Q,P) = t_r(Q,P)^{h(S)/r}$. Any $h \in \mathbb{Z}[z]$ such that the above pairing is non-degenerate satisfies $||h||_1 \geq r^{1/\varphi(k)}$.

To use Theorem II.24 in practice, we need to find a h with $||h||_1$ small, but larger than $r^{1/\varphi(k)}$, else the pairing is degenerate. Such a polynomial can be found by finding short vectors in the following m-dimensional lattice (with $\varphi(k) \leq m \leq n$) (spanned by the rows)

$$L := \begin{pmatrix} r & 0 & 0 & \cdots & 0 \\ -S & 1 & 0 & \cdots & 0 \\ -S^2 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \ddots & \\ -S^{m-1} & 0 & \dots & 0 & 1 \end{pmatrix}.$$

Any short vector $(w_0, w_1, \ldots, w_{m-1})$ such that $h = \sum_{i=0}^{m-1} w_i z^i$ satisfies $||h||_1 \geq r^{1/\varphi(k)}$ and $h(S) \not\equiv 0 \bmod r^2$ gives rise to a non-degenerate pairing.

For families of pairing friendly curves (see Section 5), r and q are obtained as the evaluation of polynomials $r(x)$ and $q(x)$. As such, the shortest vectors in L can also be parameterized by polynomials in x, which shows that coefficients h_i of h in Theorem II.24 are related and this provides a further simplification of the pairing computation. To illustrate this approach we give the following example taken from [Ver08b].

Example II.25 The family of BN-curves [BN05] (see Section 5.2.2) has $k = 12$ and is given by the following parameterizations:

$$p(x) = 36x^4 + 36x^3 + 24x^2 + 6x + 1 \qquad r(x) = 36x^4 + 36x^3 + 18x^2 + 6x + 1 \ .$$

The shortest vectors in the lattice L for the Euclidean norm are given by

$$V_1(x) = [x + 1, x, x, -2x] \qquad V_2(x) = [2x, x + 1, -x, x] \ .$$

Since there is such an easy relation between the $h_i(x)$, the computation of the function $f_{S,h,Q}$ in Theorem II.24 follows immediately from $f_{x,Q}$. Alternatively, we can look for short vectors with minimal number of coefficients of size x and obtain

$$W(x) = [6x + 2, 1, -1, 1] \ .$$

Since $f_{1,Q} = 1$ and $f_{-1,Q} = 1/f_{1,Q}v_Q$ (which disappears after final exponentiation), the pairing $a_{S,h}$ can be computed as

$$a_{S,h} = \left(f_{6x+2,Q}(P) \cdot l_{Q_3,-Q_2}(P) \cdot l_{-Q_2+Q_3,Q_1}(P) \cdot l_{Q_1-Q_2+Q_3,[6x+2]Q}\right)^{(q^k-1)/r} ,$$

where $Q_i = Q^{q^i}$ for $i = 1, 2, 3$.

Theorem II.22 and II.24 define ate pairings on $\mathbb{G}_2 \times \mathbb{G}_1$ and as remarked before, we really require twists to speed up the computations in $\mathbb{G}_2$. This complication could be avoided completely if a similar construction would be possible on $\mathbb{G}_1 \times \mathbb{G}_2$. As will be shown in the next section, this is possible for supersingular elliptic curves. However, for ordinary elliptic curves it is only possible in a restricted sense. The following theorem should be compared with Theorem II.22. The proof is exactly the same as the proof of Theorem II.22 by replacing π_q by a different purely inseparable endomorphism $\psi \circ \pi_q^e$, with $\psi \circ \pi_q^e(Q) = Q$ for $Q \in \mathbb{G}_2$ and $\psi \circ \pi_q(P) = [q]P$ for $P \in \mathbb{G}_1$.

Theorem II.26 (Twisted Ate Pairing) *Assume E admits a twist of degree d and let $z = \gcd(d, k)$ and $e = k/z$. Let S be any integer with $S \equiv q^e \pmod r$ and set $\lambda = S^z - 1$ and $C = \lambda/r$, then*

$$a_S^t : \mathbb{G}_1 \times \mathbb{G}_2 \to \mu_r \subset \mathbb{F}_{q^k}^* : (P, Q) \mapsto a_S^t(P, Q) = f_{S,P}(Q)^{(q^k-1)/r} ,$$

defines a bilinear pairing which is non-degenerate if and only if $\gcd(r, C) = 1$.

Theorem II.24 can equally easy be adapted by replacing all q's by q^e's. The main difference with Theorem II.22 is that $S \equiv q^e \pmod{r}$, with $e = k/z$. The parameter S can thus only be as small as $r^{\varphi(z)}$ and in most cases we have $z < k$. For $k = d$, we would obtain a very nice pairing faster than the ate pairing. Unfortunately, the following proposition [FST06, Proposition 7.1] shows that in this case the curve is either supersingular or has a very large cofactor, which offsets the speed up.

Proposition II.27 *Let E be an elliptic curve over $\mathbb{F}_q$ with subgroup of order r and embedding degree $k > 1$. If E has a twist E' of degree k and $r > 4\sqrt{q}$, then E is supersingular.*

4.2. Eta Pairing on Supersingular Elliptic Curves

The eta pairing was introduced by Barreto et al. [BGhS07] building on earlier work of Duursma-Lee [DL03] and predates the ate pairing introduced in the previous section. However, it can also be viewed as an instance of the twisted ate pairing on supersingular elliptic curves, which was not excluded by Proposition II.27.

Theorem II.28 (Eta Pairing) *Let $E/\mathbb{F}_q$ be supersingular with $r \mid \#E(\mathbb{F}_q)$ and $\gcd(r, q) = 1$. Let S be any integer with $S \equiv q \pmod{r}$ and set $\lambda = S^k - 1$ and $C = \lambda/r$, then*

$$\eta_S : \mathbb{G}_1 \times \mathbb{G}_2 \to \mu_r \subset \mathbb{F}_{q^k}^* : (P, Q) \mapsto \eta_S(P, Q) = f_{S,P}(Q)^{(q^k-1)/r},$$

defines a bilinear pairing which is non-degenerate if and only if $\gcd(r, C) = 1$.

The proof is again very similar to the proof of Theorem II.22, the only difference being that π_q needs to be replaced by its dual $\hat{\pi}_q$, called Verschiebung.

Remark II.29 The final exponentiation can be changed as follows: let $N = \gcd(q^k - 1, \lambda)$ and $C = \lambda/N$, then $f_{S,Q}^{c_S(q^k-1)/N}$ with $c_S = \gcd(N, \sum_{i=0}^{k-1} S^{k-1-i}q^i)$ defines a bilinear pairing which is non-degenerate if and only if $\gcd(r, C) = 1$.

5. Ordinary Pairing Friendly Elliptic Curves

This section gives a brief overview of the most important algorithms to construct ordinary pairing friendly elliptic curves. For the supersingular case we refer to Section 1.3. The exposition will be limited, since any reader interested in implementing pairing based cryptography should really consult the excellent taxonomy of pairing friendly elliptic curves [FST06] by Freeman, Scott and Teske.

The need for specialized constructions was already apparent from Section 1.1: for an elliptic curve E over $\mathbb{F}_q$ and r a prime divisor of $\#E(\mathbb{F}_q)$ with $\gcd(r, q) = 1$, the embedding degree k was defined as the order of q in $(\mathbb{Z}/r\mathbb{Z})^*$. So in general, k will be as large as r which makes computing in the finite field $\mathbb{F}_{q^k}$ impossible. Furthermore, q, r and k should be chosen such that the effort taken by the best attacks on the discrete logarithm problem in an order r subgroup of $E(\mathbb{F}_q)$, e.g. using Pollard rho [Pol78], is balanced with the effort taken by the best attack on the discrete logarithm problem in

$\mathbb{F}_{q^k}^*$. For example, to attain an 128-bit security level one should choose $r \simeq 2^{256}$ and $q^k \simeq 2^{3072}$.

All algorithms for ordinary pairing friendly curves are based on the complex multiplication (CM) method (see [AM93]). The CM method constructs an elliptic curve E over $\mathbb{F}_q$ with a given number of points $\#E(\mathbb{F}_q) = q + 1 - t$. By taking the discriminant of the characteristic equation of Frobenius, we obtain the CM norm equation

$$4q - t^2 = DV^2 \,, \tag{2}$$

where D is called the discriminant and is positive and squarefree for odd t or of the form $4d$ with d squarefree for even t. Given q and t, the CM method will efficiently construct an elliptic curve E over $\mathbb{F}_q$ with $q + 1 - t$ points only when D is sufficiently small, e.g. $D < 10^{10}$.

The following proposition provides an easy characterization of the embedding degree in terms of the trace of Frobenius and lies at the heart of all constructions.

Proposition II.30 *Let E be an elliptic curve over $\mathbb{F}_q$ with $\#E(\mathbb{F}_q) = q + 1 - t = hr$, with r prime. Let k be a positive integer with $r \nmid k$, then k is the embedding degree with respect to r if and only if $\Phi_k(q) \equiv 0 \pmod{r}$ or equivalently $\Phi_k(t - 1) \equiv 0 \pmod{r}$.*

5.1. Cocks-Pinch Method

The Cocks-Pinch method [CP01] is one of the most flexible methods to construct pairing friendly elliptic curves of arbitrary embedding degree k. The idea of the method is to first choose the embedding degree k, a prime r and a CM discriminant D small enough for the CM method to be efficient. The other parameters then follow from the CM equation (2) and Proposition II.30.

Lemma II.31 *Let k be a positive integer and fix a positive square-free integer D. Then compute the following:*

1. *Let r be a prime with $k \mid (r - 1)$ and $\left(\frac{-D}{r}\right) = 1$.*
2. *Let z be a k-th root of unity in $(\mathbb{Z}/r\mathbb{Z})^*$ and set $t \equiv z + 1 \pmod{r}$.*
3. *Let $V = \pm(t - 2)/\sqrt{-D} \bmod r$.*
4. *Let j be an integer such that $q = (t^2 + D(V + jr)^2)/4$ is prime.*

Then there exists an elliptic curve E over $\mathbb{F}_q$ with $r \mid \#E(\mathbb{F}_q)$ and embedding degree k.

Although the Cocks-Pinch method is very flexible and efficient, it has one major drawback: since the candidates for t computed in step 2 and for V in step 3 behave as random integers modulo r, step 4 implies that the prime $q \simeq r^2$, so the cofactor h will in general be of size r. A related method was devised by Dupont, Enge and Morain [DEM05], but is less flexible in the choice of r.

5.2. Families of Curves

A very powerful method to construct pairing friendly elliptic curves is to parameterize t, r, q by polynomials $t(x), r(x), q(x)$ and require that these satisfy the corresponding properties: $r(x) \mid q(x) + 1 - t(x)$, $r(x) \mid \Phi_k(t(x) - 1)$ and $DV^2 = 4q(x) - t(x)^2$ has infinitely many integer solutions. To illustrate this approach we review two famous examples.

5.2.1. MNT-curves

Miyaji, Nakabayashi and Takano [MNT01] obtained a complete classification of ordinary elliptic curves of *prime order* and embedding degree $k = 3, 4, 6$.

Theorem II.32 (MNT-curves) *Let $q > 64$ be prime and E an ordinary elliptic curve over $\mathbb{F}_q$ such that $r = \#E(\mathbb{F}_q)$ is prime. Let $t = q+1-r$, then E has embedding degree k if and only if q and t are of the form*

k	q	t
3	$12x^2 - 1$	$-1 \pm 6x$
4	$x^2 + x + 1$	$-x$ *or* $x + 1$
6	$4x^2 + 1$	$1 \pm 2x$

Although it is easy to find $\bar{x}$'s such that both $q(\bar{x})$ and $r(\bar{x})$ are prime in the above theorem, in general it will be impossible to construct the corresponding elliptic curve using the CM method since the discriminant D will be far too large. Furthermore, Luca and Shparlinski [LS06] argue then there are only a finite number of MNT elliptic curves with bounded discriminant D. Despite these results, it is possible to obtain a rather limited number of MNT curves in the cryptographic range [PSV06,SB04].

5.2.2. BN-curves

The MNT theorem shows that to obtain more pairing friendly elliptic curves we either have to allow non-trivial cofactors or have to take $k > 6$. Barreto and Naehrig [BN05] discovered a family of ordinary elliptic curves of prime order and embedding degree $k = 12$. In stark contrast with the MNT curves, the BN curves are available in abundance and easy to construct, since they have fixed discriminant $D = 3$. The main observation uses results by Galbraith, McKee and Valença [GMV07] in that $\Phi_{12}(6x^2)$ factors as a product on two degree four polynomials. Barreto and Naehrig noted that if one takes $r(x) = 36x^4+36x^3+18x^2+6x+1$, which is one of these factors, and sets $t(x) = 6x^2+1$ and thus $q(x) = r(x) + t(x) - 1$, then the CM norm equation becomes

$$t(x)^2 - 4q(x) = -3(1 + 4x + 6x^2)^2 ,$$

which shows that BN curves have discriminant $D = 3$. For every $\bar{x}$ with $q(\bar{x})$ and $r(\bar{x})$ prime, the corresponding elliptic curve can be easily generated by considering elliptic curves with equation $E_b : y^2 = x^3 + b$ for $b \in \mathbb{F}_q$. Since there are only 6 isomorphism classes, the correct curve E can be found quickly by choosing random $b \in \mathbb{F}_q$ and testing if $r(\bar{x})P = \mathcal{O}$ for an $\mathbb{F}_q$-rational point P on the curve E_b.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-31

Chapter III

Identity-Based Signatures

Eike KILTZ [a] and Gregory NEVEN [b]
[a] *CWI Amsterdam, The Netherlands*
[b] *IBM Zürich Research Laboratory, Switzerland, and Katholieke Universiteit Leuven, Belgium*

Abstract. This chapter gives an overview of the literature on identity-based signature (IBS) schemes, from Shamir's seminal scheme to the current state-of-the-art. Rather than presenting all schemes separately, we present three generic transformations that together cover the majority of known IBS schemes as special cases. The first transformation follows a certification approach based on standard signatures; the second is a transformation in the random oracle model from "convertible" identification schemes; and the third is based on hierarchical identity-based encryption. We also discuss a number of direct schemes that escape being covered by any of the generic transformations. Finally, we show how the principles of the first transformation can be extended to a hierarchical setting and to IBS schemes with special properties.

Keywords. Identity-based signatures, digital signatures, identification schemes

Digital signatures are among the most basic primitives in cryptography, providing authenticity, integrity, and non-repudiation in an asymmetric setting. In their most basic form, each user in the system generates his own key pair consisting of a public key and a corresponding secret key, and the user is assumed to be uniquely identified by his public key. In the real world however, users are generally not identified by randomly generated keys, but by more meaningful identities like their names or email addresses. To map public keys to real-world identities, a so-called public-key infrastructure (PKI) needs to be set up, for example involving a hierarchy of trusted certification authorities (CAs) that can certify public keys as belonging to a certain user.

In the identity-based setting, as proposed by Shamir in 1984 [Sha85], the public key of a user simply *is* his identity, simplifying the PKI requirements. The corresponding secret key is issued by a trusted key generation center (KGC), who derives it from a master secret that only the KGC knows, and who is assumed to have an out-of-band way to verify the identity of the user. This eliminates some of the costs associated to PKIs and certificates, and opens the way to more efficient schemes.

From a security point of view, the major drawback of identity-based cryptography is the inherent key escrow property: the KGC can derive the secret keys of all users in the system, and must therefore be trusted not to abuse this power. This is unlike a traditional PKI, where the CA only issues certificates on user-generated public keys, but does not know the corresponding secret keys. While most people find it a discomforting thought that a malafide KGC can sign any message on their behalf, one should be aware that the same type of fraud is possible in the public-key setting as well. Namely, since the certificate is usually sent along with the signature, a cheating CA can always generate a fake certificate for a public key of which it knows the corresponding secret key, and thereby create valid signatures. The victim could try to prove his innocence by showing his real certificate to a judge, but nothing prevents the CA from claiming that the user registered two different public keys. The escrow property is therefore not so much an issue for signatures as it is for encryption, where a malafide KGC can actually decrypt ciphertexts intended for any of its users. So even though there is no legitimate use for escrow of signing keys, as already mentioned in Chapter 1, a limited form of key escrow is inherently present in *both* PKI-based and ID-based signature schemes.

Identity-based signatures (IBS) also seem to be much "easier" to achieve than identity-based encryption (IBE), of which only few instantiations are known. In contrast, many practical instantiations of IBS schemes have been known for decades, including the scheme in Shamir's seminal paper from 1984 [Sha85]. In this chapter, we give an overview of the state-of-the-art in IBS schemes. We present three generic transformations to build IBS schemes from standard signature schemes, from a special class of identification schemes, and from hierarchical identity-based encryption schemes, respectively. With each transformation, we discuss some of its most interesting concrete instantiations, and compare the efficiency and security properties of all these schemes. Finally, we show how the first transformation can also be applied to obtain identity-based variants of signature schemes with special properties, including blind, threshold, and verifiably encrypted signatures.

1. Definition of Identity-Based Signatures

We first introduce some notation. If $x_1, \ldots, x_n$ are bit strings, then we denote by $x_1 \| \ldots \| x_n$ a string encoding of $x_1, \ldots, x_n$ from which the constituent objects are uniquely recoverable. If x is a string, then $|x|$ denotes its length, and if S is a set, then $|S|$ is its cardinality. If A is a randomized algorithm, then $y \overset{\$}{\leftarrow} \mathsf{A}(x_1, x_2, \ldots)$ means that A has inputs $x_1, x_2, \ldots$ and that y is assigned the output of A when run on a fresh random tape.

We measure the resources of an adversary, such as its running time and its number of oracle queries, asymptotically in terms of an underlying security parameter k. A function $\nu(k)$ is said to be negligible (in k) if for all $c \in \mathbb{N}$ there exists $k_c \in \mathbb{N}$ such that $\nu(k) < k^{-c}$ for all $k > k_c$.

An *identity-based signature (IBS) scheme* is a tuple of algorithms $\mathcal{IBS} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Sign}, \mathsf{Vf})$ with running time polynomial in the security parameter k. The first three may be randomized but the last is not. The trusted key distribution center runs the setup algorithm Setup on input 1^k to obtain a master public and secret key pair (mpk, msk). (Here, 1^k is the unary notation of the security parameter k.) To generate the secret signing key usk for the user with identity $id \in \{0,1\}^*$, it runs the key derivation algorithm KeyDer on input msk and id. The signing key is assumed to be securely communicated to the user in question. On input usk and a message M, the signing algorithm Sign returns a signature σ of M. On input mpk, id, M, and a signature σ, the verification algorithm Vf returns 1 if σ is valid for id and M, and returns 0 otherwise. Correctness requires that $\mathsf{Vf}(mpk, id, M, \mathsf{Sign}(usk, M)) = 1$ with probability one for all $k \in \mathbb{N}$ and $id, M \in \{0,1\}^*$ whenever the keys mpk, M, usk are generated as indicated above.

For security we consider the notion of existential unforgeability under chosen-message and chosen-identity attack (uf-cma) [GS02]. Security is defined through an experiment with a forger F and parameterized with the security parameter k. The experiment begins with the generation of a fresh master key pair $(mpk, msk) \xleftarrow{\$} \mathsf{Setup}(1^k)$. The forger F is run on input the master public key mpk, and has access to the following oracles:

- $\mathsf{KeyDer}(\cdot)$: On input identity $id \in \{0,1\}^*$, this oracle returns a secret signing key $usk \xleftarrow{\$} \mathsf{KeyDer}(msk, id)$.
- $\mathsf{Sign}(\cdot,\cdot)$: On input identity $id \in \{0,1\}^*$ and message $M \in \{0,1\}^*$, this oracle returns a signature $\sigma \xleftarrow{\$} \mathsf{Sign}(usk, M)$ where $usk \xleftarrow{\$} \mathsf{KeyDer}(msk, id)$.

At the end of its execution, the forger outputs identity id^*, message M^* and a forged signature σ^*. The forger is said to win the game if $\mathsf{Vf}(mpk, id^*, M^*, \sigma^*) = 1$ and F never queried $\mathsf{KeyDer}(id^*)$ or $\mathsf{Sign}(id^*, M^*)$. The advantage $\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{IBS},\mathsf{F}}(k)$ is defined as the probability that F wins the game, and $\mathcal{IBS}$ is said to be uf-cma secure if $\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{IBS},\mathsf{F}}(k)$ is negligible in k for all polynomial-time forgers F.

2. The Certification Approach

Unlike identity-based encryption, there is a very natural way to build identity-based signatures from more basic cryptographic tools by using certificates, as already pointed out in Chapter 1. This may sound paradoxical since one of the primary purposes of identity-based cryptography is to avoid certificates, but certification here refers to a technique, not to a PKI. The idea is simply that the a user's secret key includes a secret key of a standard signature scheme and a certificate for the corresponding public key, i.e., a standard signature from the authority that links the user's identity to that public key. To accomplish IBS, the user signs messages using the secret signing key, and appends to the signature his public key and certificate.

This straightforward construction has long been folklore in the research community, and was explicitly mentioned in [GS02,DKXY03,BNN04]. The construction is important however as a benchmark, relative to which the efficiency of direct IBS schemes must be measured. Moreover, from a foundational point of view, the certificate-based construction shows that IBS schemes can be built in the standard model from one-way functions [Rom90]. This is in stark contrast with identity-based encryption, for which the answer to such fundamental questions is still unknown.

2.1. Standard Signature Schemes

Before giving more details about the construction, we recall the syntax and security definitions of standard signature (SS) schemes [GMR88]. It is defined as a tuple of polynomial-time algorithms $\mathcal{SS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vf})$. The randomized key generation algorithm KeyGen on input 1^k generates a key pair (pk, sk). The signer creates a signature on a message M via $\sigma \xleftarrow{\$} \mathsf{Sign}(sk, M)$, and the verifier can check the validity of a signature by testing whether $\mathsf{Vf}(pk, M, \sigma) = 1$. Correctness requires that $\mathsf{Vf}(pk, M, \mathsf{Sign}(sk, M)) = 1$ with probability one for all $M \in \{0,1\}^*$.

Security is defined through the notion of existential unforgeability under chosen-message attack (uf-cma), described by the following game with a forger F. The forger is run with a fresh public key pk as input, and is given access to a signing oracle for the corresponding secret key sk. It is said to win the game if it can output a pair (M^*, σ^*) such that $\mathsf{Vf}(pk, M^*, \sigma^*) = 1$ and it never queried M^* from the signing oracle. The advantage $\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{SS},\mathsf{F}}(k)$ is defined as the probability that F wins this game, and $\mathcal{SS}$ is said to be uf-cma secure if this is a negligible function in k for all polynomial-time forgers F.

2.2. *The* SS-2-IBS *Transformation*

Given a standard signature scheme $\mathcal{SS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vf})$, one can build a certificate-based IBS scheme $\mathcal{C}ert\text{-}\mathcal{IBS} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Sign}', \mathsf{Vf}') = \text{SS-2-IBS}(\mathcal{SS})$ as follows. One can easily prove that if $\mathcal{SS}$ is uf-cma secure, then $\mathcal{C}ert\text{-}\mathcal{IBS}$ is uf-cma secure as well.

Scheme III.1 The certificate-based IBS scheme $\mathcal{C}ert\text{-}\mathcal{IBS}$

```
Algorithm Setup($1^k$):
    $(mpk, msk) \xleftarrow{\$} \mathsf{KeyGen}(1^k)$
    return $(mpk, msk)$.
Algorithm KeyDer($msk, id$):
    $(pk, sk) \xleftarrow{\$} \mathsf{KeyGen}(1^k)$ ; $cert \xleftarrow{\$} \mathsf{Sign}(msk, pk\|id)$
    return $usk \leftarrow (sk, pk, cert)$.
Algorithm Sign'($usk, M$):
    Parse $usk$ as $(sk, pk, cert)$ ; $\sigma \xleftarrow{\$} \mathsf{Sign}(sk, M)$
    return $\sigma' \leftarrow (\sigma, pk, cert)$.
Algorithm Vf'($mpk, id, M, \sigma'$):
    Parse $\sigma'$ as $(\sigma, pk, cert)$
    if $\mathsf{Vf}(pk, M, \sigma) = 1$ and $\mathsf{Vf}(mpk, pk\|id, cert) = 1$ then $d \leftarrow 0$ else $d \leftarrow 1$
    return $d$.
```

Since there are numerous constructions of uf-cma secure SS schemes without random oracles [GHR99,CS99,BB04c], we obtain from the above IBS schemes without random oracles. Given that the existence of uf-cma secure SS schemes is equivalent to the existence of one-way functions [Rom90], an easy corollary says that uf-cma secure IBS schemes exist if and only if one-way functions exist.

When instantiated with an efficient SS scheme, the SS-2-IBS transformation yields fairly efficient IBS schemes. Signing costs the same as for the underlying SS scheme, and verification comes at twice the cost of the SS scheme. The size of the signature increases due to inclusion of the certificate. The sole goal of the direct IBS schemes presented in this chapter is therefore to reduce costs below that of even the best instantiations of the SS-2-IBS transform.

2.3. Instantiations

The main advantage of the certificate-based approach from is its generality. The efficiency and security properties depend highly on the underlying SS scheme being used. For example, to obtain IBS schemes with security in the standard model, one can use any of the standard-model SS schemes of [GHR99,CS99,BB04c]. To save on signature size, one can instantiate the scheme with short BLS signatures [BLS01]. The IBS schemes thus obtained are not the most efficient, but they may be an interesting alternative in case for example fast SS implementations are readily available.

3. Constructions from Identification Schemes

A second generic transformation yielding more efficient IBS schemes was proposed by Bellare, Namprempre, and Neven [BNN04]. The transform works for a particular class of three-move identification schemes called *convertible* identification schemes. Instances of such schemes can be found based on various cryptographic assumptions such as RSA, factoring, and pairings. The transformation itself bears a lot in common with the Fiat-Shamir transform [FS87] that turns a three-move identification scheme into a standard (i.e., not identity-based) signature scheme. Bellare et al. proved the security of the transformation in the random oracle model, and gave an extensive overview of instantiations that either appeared in the literature as identification schemes, or that appeared as IBS schemes directly but that in retrospect can be seen as being derived from a convertible identification scheme.

3.1. Canonical Convertible Identification Schemes

We first recall the definition of standard identification (SI) schemes, and then define a particular class of schemes to which the transform applies. A SI scheme is a tuple of polynomial-time algorithms $\mathcal{SI} = (\mathsf{KeyGen}, \mathsf{P}, \mathsf{V})$. The key generation algorithm KeyGen, on input the security parameter 1^k, outputs a fresh key pair (pk, sk). The prover and verifier algorithms P and V are interactive algorithms that together form the identification protocol. The prover P is run with the secret key sk as initial input, and interacts with the verifier V that gets the public key pk as initial input. At the end of the interaction, V outputs 0 or 1 indicating whether the prover was successfully identified. Correctness requires that V outputs 1 for all honest provers.

For security, we focus on the relatively weak notion of resistance against impersonation under passive attacks (imp-pa), since this notion suffices for our purposes of building IBS schemes. The adversary A, also called an impersonator, gets as input a fresh public key pk, and has access to a transcript oracle that on each invocation returns the transcript of an interaction between the $\mathsf{P}(sk)$ and $\mathsf{V}(pk)$ algorithms. (It is easy to see that the protocol has to be randomized for the scheme to be secure, so the generated transcripts will be different at each invocation.) The impersonator A can then interact with an instance of $\mathsf{V}(pk)$, and wins the game if the latter outputs 1. The advantage $\mathbf{Adv}^{\text{imp-pa}}_{\mathcal{SI},\mathsf{A}}(k)$ is the probability that A wins, and $\mathcal{SI}$ is said to be imp-pa secure if the advantage is negligible for all polynomial-time adversaries A.

Before defining convertibility of SI schemes, we need to introduce the concept of trapdoor-samplable relations.

Definition III.1 A family of *trapdoor-samplable relations* $\mathcal{F}$ is a triplet of polynomial-time algorithms $(\mathsf{TDG}, \mathsf{Smp}, \mathsf{Inv})$ such that the following properties hold:

- *Efficient generation:* On input 1^k, TDG outputs the description of a relation $R \subseteq \mathrm{Dom} \times \mathrm{Rng}$ together with its trapdoor information t;
- *Samplability:* The algorithm Smp, on input the description of a relation R, returns a uniformly random couple from R;
- *Inversion:* On input the description of a relation R, the corresponding trapdoor t, and an element $y \in \mathrm{Rng}$, the randomized algorithm Inv outputs a random element of $R^{-1}(y)$;
- *Regularity:* For every relation R in the family, there is an integer d such that $|R^{-1}(y)| = d$ for all $y \in \mathrm{Rng}$.

A SI scheme $\mathcal{SI} = (\mathsf{KeyGen}, \mathsf{P}, \mathsf{V})$ is said to be *convertible* (or a cSI scheme) if its key-generation process is underlain by a family of trapdoor-samplable relations. More specifically, there must exist a family $\mathcal{F} = (\mathsf{TDG}, \mathsf{Smp}, \mathsf{Inv})$ such that the keys generated by KeyGen are of the form $pk = (R, y)$ and $sk = (R, x)$ distributed according to

$$(R, t) \overset{\$}{\leftarrow} \mathsf{TDG}(1^k) \;\; ; \;\; (x, y) \overset{\$}{\leftarrow} \mathsf{Smp}(R) \ .$$

A cSI scheme $\mathcal{SI} = (\mathsf{KeyGen}, \mathsf{P}, \mathsf{V})$ is said to be *canonical* if it follows a three-move structure where the prover initiates the communication with a "commitment" *cmt* distributed uniformly over a set $\mathrm{CmtSet}(R)$ possibly depending on the relation embedded in the public and secret keys; the verifier sends back a "challenge" *ch* chosen uniformly from a set $\mathrm{ChSet}(R)$; the prover replies with a "response" *rsp*; and the verifier's decision to accept or reject is a deterministic function $\mathrm{dec}(pk, cmt \,\|\, ch \,\|\, rsp) \in \{0, 1\}$ of the public key and the communication transcript. We require that $1/|\mathrm{CmtSet}(R)|$ is negligible.

3.2. *The* cSI-2-IBS *Transformation*

A canonical cSI scheme directly yields a SS scheme through the well-known Fiat-Shamir transform [FS87]; the resulting SS scheme is uf-cma secure in the random oracle model if the underlying SI scheme is imp-pa secure [AABN02]. To obtain an IBS scheme from a canonical cSI scheme $\mathcal{SI} = (\mathsf{KeyGen}, \mathsf{P}, \mathsf{V})$, consider the scheme $\mathcal{IBS} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Sign}, \mathsf{Vf}) = \text{cSI-2-IBS}(\mathcal{SI})$ as given below, where $\mathrm{H} : \{0,1\}^* \to \mathrm{Rng}$ and $\mathrm{G} : \{0,1\}^* \to \mathrm{ChSet}(R)$ are hash functions, modeled as random oracles, whose range depends on the relation R. Bellare et al. [BNN04] showed that if $\mathcal{SI}$ is imp-pa secure, then $\text{cSI-2-IBS}(\mathcal{SI})$ is uf-cma secure in the random oracle model.

3.3. *Instantiations*

As many as twelve different suitable cSI schemes were surfaced in [BNN04] based on factoring, RSA, pairings, and discrete logarithms. These give rise to twelve different IBS schemes through the cSI-2-IBS transform. As an example, we highlight here the original IBS scheme proposed by Shamir in 1984 [Sha85].

Let $\mathsf{K}_{\mathrm{rsa}}$ be an *RSA key generator* that on input 1^k outputs a modulus N that is the product of two distinct odd primes, and exponents e, d such that $ed = 1 \bmod (p -$

Scheme III.2 The scheme $\mathcal{IBS} = \text{cSI-2-IBS}(\mathcal{SI})$

```
Algorithm Setup(1^k):
    $(R,t) \xleftarrow{\$} \mathsf{TDG}(1^k)$ ; $mpk \leftarrow R$ ; $msk \leftarrow (R,t)$
    return $(mpk, msk)$.
Algorithm KeyDer(msk, id):
    $(R,t) \leftarrow msk$ ; $x \xleftarrow{\$} \mathsf{Inv}(R,t,\mathrm{H}(id)$
    return $usk \leftarrow (R,x)$.
Algorithm Sign(usk, M):
    $(R,x) \leftarrow usk$ ; $cmt \xleftarrow{\$} \mathsf{P}(usk)$ ; $ch \leftarrow \mathrm{G}(cmt\|M)$ ; $rsp \leftarrow \mathsf{P}(ch)$
    return $\sigma \leftarrow (cmt, rsp)$.
Algorithm Vf(mpk, id, M, σ):
    $R \leftarrow mpk$ ; $(cmt, rsp) \leftarrow \sigma$ ; $pk \leftarrow (R, \mathrm{H}(id))$ ; $ch \leftarrow \mathrm{G}(cmt\|M)$
    return $\mathrm{dec}(pk, cmt\|ch\|rsp)$.
```

$1)(q-1)$ and such that $e > 2^{l(k)}$ for some function $l(\cdot)$. We say that the RSA function associated to $\mathsf{K}_{\mathrm{rsa}}$ is one-way if

$$\mathbf{Adv}^{\mathrm{rsa}}_{\mathsf{K}_{\mathrm{rsa}},\mathsf{A}}(k) = \Pr\left[x^e = y \bmod N \ : \ \begin{array}{c} (N,e,d) \xleftarrow{\$} \mathsf{K}_{\mathrm{rsa}}(1^k) \ ; \ y \xleftarrow{\$} \mathbb{Z}_N^* \ ; \\ x \leftarrow \mathsf{A}(1^k, N, e, y) \end{array} \right]$$

is negligible in k for all polynomial-time algorithms A. First consider the identification scheme $\mathcal{Sh}$-$\mathcal{SI}$ given in Scheme III.3.

Scheme III.3 The SI scheme underlying Shamir's IBS scheme

```
Algorithm KeyGen(1^k):
    $(N,e,d) \xleftarrow{\$} \mathsf{K}_{\mathrm{rsa}}(1^k)$ ; $x \xleftarrow{\$} \mathbb{Z}_N^*$ ; $y \leftarrow x^e \bmod N$
    $pk \leftarrow (N,e,y)$ ; $sk \leftarrow (N,e,x)$
    return $(pk, sk)$.
Algorithm P(sk):                                     Algorithm V(pk):
    Parse $sk$ as $(N,e,x)$                              Parse $pk$ as $(N,e,y)$
    $t \xleftarrow{\$} \mathbb{Z}_N^*$ ; $T \leftarrow t^e \bmod N$     $T$ ►
                                        ◄  $c$           $c \xleftarrow{\$} \{0,1\}^{l(k)}$
    $s \leftarrow x\, t^c \bmod N$             $s$ ►        if $s^e \equiv y T^c \pmod N$
                                                             then $d \leftarrow 1$ else $d \leftarrow 0$
                                                         return $d$.
```

To see why this SI scheme is convertible, observe the family of trapdoor-samplable relations described by pairs (N,e) and corresponding trapdoor d such that $R = \{(x,y) \in {\mathbb{Z}_N^*}^2 \ : \ y = x^e \bmod N\}$. It is also canonical with $\mathrm{CmtSet}(N,e) = \mathbb{Z}_N^*$ and $\mathrm{ChSet}(N,e) = \{0,1\}^{l(k)}$. Applying the cSI-2-IBS transformation yields the $\mathcal{Sh}$-$\mathcal{IBS}$ scheme given in Scheme III.4, which is exactly the scheme from [Sha85].

Scheme III.4 Shamir's IBS scheme $\mathcal{Sh\text{-}IBS}$

```
Algorithm Setup(1^k):
  $(N, e, d) \xleftarrow{\$} \mathsf{K}_{\mathrm{rsa}}(1^k)$ ; $mpk \leftarrow (N, e)$ ; $msk \leftarrow (N, e, d)$
  return $(mpk, msk)$.

Algorithm KeyDer(msk, id):
  Parse $msk$ as $(N, e, d)$ ; $x \leftarrow \mathrm{H}(id)^d \bmod N$
  return $usk \leftarrow (N, e, x)$.

Algorithm Sign(usk, id, M):
  $(N, e, x) \leftarrow usk$ ; $t \xleftarrow{\$} \mathbb{Z}_N^*$ ; $T \leftarrow t^e \bmod N$ ; $c \leftarrow \mathrm{G}(T \| M)$ ; $s \leftarrow x\, t^c \bmod N$
  return $\sigma \leftarrow (T, s)$.

Algorithm Vf(mpk, id, M, σ):
  $(N, e) \leftarrow mpk$ ; $(T, s) \leftarrow \sigma$
  if $s^e \equiv \mathrm{H}(id)\, T^{\mathrm{G}(T\|M)} \pmod N$ then $d \leftarrow 1$ else $d \leftarrow 0$
  return $d$.
```

The $\mathcal{Sh\text{-}SI}$ scheme was shown [BNN04] to be imp-pa secure if the RSA function associated to $\mathsf{K}_{\mathrm{rsa}}$ is one-way. The $\mathcal{Sh\text{-}IBS}$ scheme is therefore uf-cma secure under the same assumption.

Other instantiations of the cSI-2-IBS transform sketched in [BNN04] include the RSA-based Guillou-Quisquater ($\mathcal{GQ\text{-}IBS}$) scheme [GQ90], the factoring-based iterated-root ($\mathcal{ItR\text{-}IBS}$) scheme [FS87,FFS88,OS90], and the pairing-based Cha-Cheon ($\mathcal{ChCh\text{-}IBS}$) scheme [CC03,Yi03]. We refer to [BNN04] for a more complete overview. The $\mathcal{BNN\text{-}IBS}$ [BNN04] and $\mathcal{BLMQ\text{-}IBS}$ [BLMQ05] schemes bear some similarity to schemes derived via the cSI-2-IBS transform, but fail to be captured by it. They were proved secure in the random oracle model under the discrete logarithm assumption and the q-strong Diffie-Hellman assumption, respectively.

4. Constructions from Hierarchical Identity-Based Encryption

As noted by Naor [BF01, Section 6] and formalized in [CFH+07], the key derivation of an identity-based encryption (IBE) scheme immediately gives rise to a standard signature scheme. Similarly, Gentry and Silverberg [GS02] observed that any two-level hierarchical identity-based encryption (HIBE) scheme (which is a natural extension of an IBE allowing for hierarchical key delegation) can be transformed into an IBS scheme. We revisit their transformation in this section.

4.1. Hierarchical Identity-Based Encryption.

A hierarchical identity $\vec{id}$ of depth d is a tuple $\vec{id} = (id_1, \ldots, id_d)$, where $id_i \in \{0,1\}^*$. We say that $\vec{id}$ of depth d is an ancestor of $\vec{id}'$ of depth d' if $\vec{id}$ is a proper prefix of $\vec{id}'$, i.e., if $d \le d'$ and $id_i = id'_i$, for all $1 \le i \le d$. If $\vec{id}$ has depth 0 then it is the empty string ϵ. Note that ϵ is a ancestor of any hierarchical identity.

A *hierarchical identity-based encryption* (HIBE) scheme of depth D is a tuple of polynomial-time algorithms $\mathcal{HIBE} = (\mathsf{Setup}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$. The first three may be randomized but the last is not. The trusted key distribution center runs the setup algorithm Setup on input 1^k to obtain a master public and secret key pair (mpk, msk). As a convention, the user secret key of $\vec{id} = \epsilon$ is the master secret key msk. When a user with hierarchical identity $\vec{id}$ wants to generate the secret key for a descendant $\vec{id}'$, it runs the key derivation algorithm KeyDer on input its own user secret key $usk_{\vec{id}}$ and the identity $\vec{id}'$. The resulting user secret key is assumed to be securely communicated to the user in question. On input $mpk, \vec{id}, M$, the encryption algorithm Enc returns a ciphertext C of M for hierarchical identity $\vec{id}$. On input $usk_{\vec{id}}$ and a ciphertext C, the decryption algorithm Dec returns a message M, or $\perp$ when the ciphertext is invalid. Correctness requires that $\mathsf{Dec}(usk_{\vec{id}}, \mathsf{Enc}(mpk, \vec{id}, M)) = M$ with probability one for all $k \in \mathbb{N}$ and $\vec{id}, M$ whenever the keys $mpk, M, usk_{\vec{id}}$ are generated as indicated above. As a special case, an IBE scheme is a HIBE of depth $D = 1$.

The common security notion of HIBE schemes is indistinguishability against chosen-plaintext attacks [GS02] (ind-id-cpa). Here we only require the HIBE to be one-way against chosen-plaintext attacks (ow-id-cpa), a slightly weaker notion that only requires it to be hard to decrypt encryptions of random messages (as opposed to adversarially-chosen ones).

4.2. *The* HIBE-2-IBS *Transformation*

The idea of this transformation is to use the user secret key of $\vec{id} = (id, M)$ as the identity-based signature of M under identity id. Given the user secret key usk_{id} of id, the hierarchical key derivation algorithm can be used for signing. Verification is done by checking whether the encryption of a random message under identity (id, M) decrypts correctly when using the signature as decryption key. More formally, given $\mathcal{HIBE} =$ (Setup, KeyDer, Enc, Dec) of depth $D = 2$ with message space $MsgSp$, we build $\mathcal{IBS} =$ (Setup, KeyDer, Sign, Vf) = HIBE-2-IBS($\mathcal{HIBE}$) as given in Scheme III.5.

Scheme III.5 The scheme $\mathcal{IBS}$ = HIBE-2-IBS($\mathcal{HIBE}$)

```
Algorithm Sign(usk_id, M):
    $\vec{id} \leftarrow (id, M)$ ; $\sigma \overset{\$}{\leftarrow} \mathsf{KeyDer}(usk_{id}, \vec{id})$
    return $\sigma$.
Algorithm Vf(mpk, id, M, σ):
    $\vec{id} \leftarrow (id, M)$ ; $M' \overset{\$}{\leftarrow} MsgSp$ ; $C \overset{\$}{\leftarrow} \mathsf{Enc}(mpk, \vec{id}, M')$
    if $\mathsf{Dec}(usk_{\vec{id}} = \sigma, C) = M'$ then $d \leftarrow 1$ else $d \leftarrow 0$
    return $d$.
```

One can prove that if $\mathcal{HIBE}$ is ow-id-cpa secure, then HIBE-2-IBS($\mathcal{HIBE}$) is uf-cma secure. In contrast to the cSI-2-IBS transformation from Section 3, this transformation does not rely on the random oracle model, so when instantiated with a HIBE scheme that is ow-id-cpa secure in the standard model, one obtains an IBS scheme that is uf-cma secure in the standard model as well.

The verification algorithm above works generically for any HIBE scheme, but for most concrete instantiations a more efficient deterministic test exists. Also, while the generic transformation yields uf-cma security of the IBS scheme under the same assumption as the HIBE scheme, the IBS scheme can usually be proved secure under a weaker assumption via a direct proof.

4.3. Instantiations

The first practical HIBE construction was the $\mathcal{GS}$-$\mathcal{HIBE}$ due to Gentry and Silverberg [GS02], which through the HIBE-2-IBS transformation leads to the $\mathcal{GS}$-$\mathcal{IBS}$ scheme that was also mentioned in [GS02]. Its security is based on the CDH assumption in groups equipped with bilinear maps in the random oracle model. We briefly describe the scheme here.

For simplicity we restrict our attention to symmetric pairings $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_{\mathrm{T}}$ generated by a polynomial-time *pairing generator* $\mathsf{K}_{\mathrm{pair}}$. On input 1^k this algorithm outputs $pars = (\mathbb{G}, \mathbb{G}_{\mathrm{T}}, \hat{e}, p, g)$ where $\mathbb{G}, \mathbb{G}_{\mathrm{T}}$ are descriptions of an additive group $\mathbb{G}$ and a multiplicative group $\mathbb{G}_{\mathrm{T}}$ of the same prime order p, P is a generator of $\mathbb{G}$, and $\hat{e}$ is the description of a non-degenerate computable bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_{\mathrm{T}}$. The computational Diffie-Hellman (CDH) problem in $\mathbb{G}$ associated to $\mathsf{K}_{\mathrm{pair}}$ is said to be hard if

$$\mathbf{Adv}^{\mathrm{cdh}}_{\mathsf{K}_{\mathrm{pair}},\mathsf{A}}(k) \;=\; \Pr[\mathsf{A}(pars, aP, bP) = abP \;:\; pars \xleftarrow{\$} \mathsf{K}_{\mathrm{pair}}(1^k)\;;\, a, b \xleftarrow{\$} \mathbb{Z}_p]$$

is negligible in k for any polynomial-time algorithm A. The assumption that CDH is hard is a weaker assumption than the Bilinear CDH assumption used by Boneh and Franklin [BF03] which states that, given (aP, bP, cP), computing $\hat{e}(P, P)^{abc}$ is hard.

The Setup and KeyDer algorithms of the $\mathcal{GS}$-$\mathcal{HIBE}$ scheme are as follows. The master secret key msk is a random exponent $x \xleftarrow{\$} \mathbb{Z}_p$, the master public key mpk contains a pairing description $pars$ generated by $\mathsf{K}_{\mathrm{pair}}(1^k)$ and the element $X \leftarrow xP$. The user secret key of an identity id_1 at the first level is $usk_{id_1} \leftarrow x \cdot \mathrm{H}(id_1)$, where $\mathrm{H} : \{0,1\}^* \rightarrow \mathbb{G}$ is a public hash function, modeled as a random oracle. The user secret key of an second-level identity $\vec{id} = (id_1, id_2)$ is a pair (R, S) where $R \leftarrow rP$ for a random $r \xleftarrow{\$} \mathbb{Z}_p$ and $S \leftarrow usk_{id_1} + r \cdot \mathrm{H}(id_1, id_2)$. Applying the HIBE-2-IBS transform and using a more efficient verification test yields the $\mathcal{GS}$-$\mathcal{IBS}$ scheme given in Scheme III.6.

The $\mathcal{GS}$-$\mathcal{HIBE}$ scheme is ow-id-cpa secure in the random oracle model if the Bilinear CDH problem in $pars = (\mathbb{G}, \mathbb{G}_{\mathrm{T}}, \hat{e}, p, g)$ associated to $\mathsf{K}_{\mathrm{pair}}$ is hard. With a direct proof the $\mathcal{GS}$-$\mathcal{IBS}$ scheme can be proved to be uf-cma secure under the weaker CDH assumption.

Another IBS scheme, $\mathcal{BB}$-$\mathcal{IBS}$, with the same security properties can be obtained by transforming the random-oracle variant of the HIBE proposed by Boneh and Boyen [BB04a]. Waters [Wat05] proposed a HIBE scheme that is ow-id-cpa secure under the Bilinear CDH assumption in the standard model. The HIBE-2-IBS transformation yields the $\mathcal{W}aters$-$\mathcal{IBS}$ scheme that was directly proposed in [PS06]. The HIBE scheme due to Boneh, Boyen and Goh [BBG05] can be combined with Waters' techniques to obtain a HIBE scheme [CS06c,KV08] that is ow-id-cpa secure under some variant of the Bilinear CDH assumption in the standard model. Again, applying the HIBE-2-IBS transform and using a more efficient verification test yields the $\mathcal{BBG}$-$\mathcal{IBS}$ scheme given in Scheme III.7.

Scheme III.6 The $\mathcal{GS}$-$\mathcal{IBS}$ scheme

```
Algorithm Setup($1^k$):
    $pars \xleftarrow{\$} \mathsf{K}_{\text{pair}}(1^k)$ ; $x \xleftarrow{\$} \mathbb{Z}_p$ ; $X \leftarrow xP$ ; $mpk \leftarrow (pars, X)$ ; $msk \leftarrow (pars, x)$
    return $(mpk, msk)$.

Algorithm KeyDer($msk, id$):
    $(pars, x) \leftarrow msk$
    return $usk_{id} \leftarrow x \cdot \mathrm{H}(id)$.

Algorithm Sign($usk_{id}, M$):
    $r \xleftarrow{\$} \mathbb{Z}_p$ ; $R \leftarrow rP$ ; $S \leftarrow usk_{id} + r \cdot \mathrm{H}(id, M)$
    return $\sigma \leftarrow (R, S)$.

Algorithm Vf($mpk, id, M, \sigma$):
    Parse $mpk$ as $(pars, X)$ and $\sigma$ as $(R, S)$
    if $\hat{e}(g, S) = \hat{e}(\mathrm{H}(id), X) \cdot \hat{e}(R, \mathrm{H}(id, M))$ then $d \leftarrow 1$ else $d \leftarrow 0$
    return $d$.
```

Scheme III.7 The $\mathcal{BBG}$-$\mathcal{IBS}$ scheme

```
Algorithm Setup($1^k$):
    $pars \xleftarrow{\$} \mathsf{K}_{\text{pair}}(1^k)$ ; $X \xleftarrow{\$} \mathbb{G}$ ; $Y \leftarrow \hat{e}(X, P)$
    $\mathbf{U} = (U_0, \ldots, U_n) \xleftarrow{\$} \mathbb{G}^{n+1}$ ; $\mathbf{V} = (V_1, \ldots, V_n) \xleftarrow{\$} \mathbb{G}^n$
    $mpk \leftarrow (pars, Y, \mathbf{U}, \mathbf{V})$ ; $msk \leftarrow (pars, X, \mathbf{U}, \mathbf{V})$
    return $(mpk, msk)$.

Algorithm KeyDer($msk, id$):
    Parse $msk$ as $(pars, X, \mathbf{U}, \mathbf{V})$ ; $r \xleftarrow{\$} \mathbb{Z}_p$
    $R_1 \leftarrow rP$ ; $R_2 \leftarrow X + r \cdot (U_0 + \sum_{i=1}^{n} id_i \cdot U_i)$
    for $i$ from 1 to $n$ do $W_i \leftarrow rV_i$
    $usk \leftarrow (pars, \mathbf{U}, \mathbf{V}, R_1, R_2, W_1, \ldots, W_n)$
    return $usk$.

Algorithm Sign($usk, id, M$):
    Parse $usk$ as $(pars, \mathbf{U}, \mathbf{V}, R_1, R_2, W_1, \ldots, W_n)$
    $s \xleftarrow{\$} \mathbb{Z}_p$ ; $S_1 \leftarrow R_1 + sP$ ; $S_2 \leftarrow R_2 + s \cdot (U_0 + \sum_{i=1}^{n} id_i \cdot U_i) + \sum_{i=1}^{n} M_i \cdot (W_i + sV_i)$
    return $\sigma \leftarrow (S_1, S_2) \in \mathbb{G}^2$.

Algorithm Vf($mpk, id, M, \sigma$):
    Parse $mpk$ as $(pars, Y, \mathbf{U}, \mathbf{V})$ and $\sigma$ as $(S_1, S_2)$
    if $\hat{e}(S_2, P) = Y \cdot \hat{e}(S_1\ ,\ U_0 + \sum_{i=1}^{n} id_i \cdot U_i + M_i \cdot V_i)$ then $d \leftarrow 1$ else $d \leftarrow 0$
    return $d$.
```

Here we assume that identities and messages are bit-stings from $\{0,1\}^n$. For arbitrary identities and messages one can use a collision-resistant hash function with image $\{0,1\}^n$, where $n = 2k$ (due to the birthday paradox). The $\mathcal{BBG}$-$\mathcal{IBS}$ scheme has a particularly short signature size of only two elements of $\mathbb{G}$, matching the signature size of the (random-oracle) $\mathcal{GS}$-$\mathcal{IBS}$ scheme. Using a direct proof its $\mathrm{uf\text{-}cma}$ security can proved under the mCDH assumption which states that given (g^a, g^b, g^{b^2}), computing g^{ab} is hard. The drawback of the $\mathcal{BBG}$-$\mathcal{IBS}$ scheme are its relatively large public parameters and (user) secret keys.

5. Efficiency and Security Comparison

Table III.1 gives an overview of the efficiency and security properties of the IBS schemes covered in this chapter. For each scheme it displays the transform through which the IBS scheme was obtained (if any), the signature size, the dominating computational overhead of signing and verification, the security assumption under which the IBS scheme has been proved secure, and whether this proof is in the random-oracle model (ROM) or the standard model (SM).

For the certificate-based scheme $\mathcal{C}ert$-$\mathcal{IBS}$ all properties are denoted in terms of the underlying standard signature scheme. The numbers for the RSA- and factoring-based schemes $\mathcal{S}h$-$\mathcal{IBS}$, $\mathcal{GQ}$-$\mathcal{IBS}$, and $\mathcal{I}t\mathcal{R}$-$\mathcal{IBS}$ are stated in terms of elements (el.), multiplications (mult.), exponentiations (exp.), and multi-exponentiations (mexp.) in the group $\mathbb{Z}_N^*$ where N is the product of two large primes. For the pairing-based schemes $\mathcal{C}h\mathcal{C}h$-$\mathcal{IBS}$, $\mathcal{GS}$-$\mathcal{IBS}$, $\mathcal{W}aters$-$\mathcal{IBS}$, $\mathcal{BBG}$-$\mathcal{IBS}$, and $\mathcal{BLMQ}$-$\mathcal{IBS}$, we consider symmetric pairings $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ over groups of prime order p. Signature sizes and computational overhead are given in terms of elements (el.) of $\mathbb{G}$ or $\mathbb{Z}_p$; exponentiations (exp.) in $\mathbb{G}$ or $\mathbb{G}_T$; and pairing evaluations (pairings). (Note that we follow the popular convention to refer to multiplication in the additive group $\mathbb{G}$ as exponentiations.) Computing a sum $U_0 + \sum_{i=1}^{n} id_i \cdot U_i$ takes $n+1$ multiplications (i.e., additions in the group) in $\mathbb{G}$, which for $n = 2k = \log_2 |\mathbb{G}|$ takes about half the time of an exponentiation in $\mathbb{G}$ using the "square-and-multiply" method. For the discrete-logarithm based $\mathcal{BNN}$-$\mathcal{IBS}$ scheme, we consider multiplicative groups $\mathbb{G}$ of prime order p. Values are in terms of elements (el.) of $\mathbb{G}$ and $\mathbb{Z}_p$, exponentiations (exp.) and multi-exponentiations (mexp.) in $\mathbb{G}$.

6. Extensions

6.1. Hierarchical Identity-Based Signatures

Similar to the concept of hierarchical identity-based encryption (HIBE) one can also consider the concept of hierarchical identity-based signatures (HIBS) [GS02]. Here hierarchical identities are tuples of identities $\vec{id} = (id_1, \ldots, id_d)$, and user secret keys for hierarchical identity $\vec{id}'$ can be derived by the owner of the user secret key from some ancestor $\vec{id}$ of $\vec{id}'$. In analogy with the certification approach for IBS schemes, Kiltz et al. [KMPR05] showed how to construct HIBS schemes from SS schemes using certificate chains. Also, analogously to the HIBE-2-IBS transform, more efficient d-level HIBS schemes can be constructed from $(d+1)$-level HIBE schemes.

6.2. Identity-Based Signatures with Special Properties

In order to satisfy the needs of some specific scenarios such as electronic commerce, cash, voting, or auctions, the original concept of a digital signature has been extended

Table III.1. Efficiency and security comparison of all treated IBS schemes.

Scheme	Transform	Sig. size	Signing time	Verif. time	Sec. assumption	ROM/SM
$\mathcal{C}ert$-$\mathcal{IBS}$	SS-2-IBS	2 sig. of $\mathcal{SS}$ 1 pk of $\mathcal{SS}$	1 Sign of $\mathcal{SS}$	2 Vf of $\mathcal{SS}$	$\mathcal{SS}$ is uf-cma	SM
$\mathcal{S}h$-$\mathcal{IBS}$	cSI-2-IBS	2 el. of $\mathbb{Z}_N^*$	2 exp. in $\mathbb{Z}_N^*$	1 mexp in $\mathbb{Z}_N^*$	RSA	ROM
$\mathcal{GQ}$-$\mathcal{IBS}$	cSI-2-IBS	2 el. of $\mathbb{Z}_N^*$	2 exp. in $\mathbb{Z}_N^*$	1 mexp in $\mathbb{Z}_N^*$	RSA	ROM
$\mathcal{I}t\mathcal{R}$-$\mathcal{IBS}$	cSI-2-IBS	2 el. of $\mathbb{Z}_N^*$	$O(k)$ mult. in $\mathbb{Z}_N^*$	$O(k)$ mult. in $\mathbb{Z}_N^*$	factoring	ROM
$\mathcal{C}h\mathcal{C}h$-$\mathcal{IBS}$	cSI-2-IBS	2 el. of $\mathbb{G}$	2 exp. in $\mathbb{G}$	2 pairings	CDH in $\mathbb{G}$	ROM
$\mathcal{BNN}$-$\mathcal{IBS}$	direct	3 el. of $\mathbb{G}$ 1 el. of $\mathbb{Z}_p$	1 exp. in $\mathbb{G}$	1 mexp in $\mathbb{G}$ 1 exp. in $\mathbb{G}$	discrete log	ROM
$\mathcal{BLMQ}$-$\mathcal{IBS}$	direct	1 el. of $\mathbb{G}$ 1 el. of $\mathbb{Z}_p$	2 exp. in $\mathbb{G}$	1 pairing 1 exp. in $\mathbb{G}_{\mathrm{T}}$	ℓ-SDH in $\mathbb{G}$	ROM
$\mathcal{GS}$-$\mathcal{IBS}$	HIBE-2-IBS	2 el. of $\mathbb{G}$	2 exp. in $\mathbb{G}$	3 pairings	CDH in $\mathbb{G}$	ROM
$\mathcal{W}aters$-$\mathcal{IBS}$	HIBE-2-IBS	3 el. of $\mathbb{G}$	1.5 exp. in $\mathbb{G}$	3 pairings 0.5 exp. in $\mathbb{G}$	CDH in $\mathbb{G}$	SM
$\mathcal{BBG}$-$\mathcal{IBS}$	HIBE-2-IBS	2 el. of $\mathbb{G}$	2 exp. in $\mathbb{G}$	2 pairings 0.5 exp. in $\mathbb{G}$	mCDH in $\mathbb{G}$	SM

and modified in multiple ways, giving rise to many kinds of digital signatures with "special properties", including blind signatures [Cha83], threshold signatures [Des88], and aggregated signatures [BGLS03]. Originally, these extensions were introduced for the (certificate-based) public-key setting, but nothing prohibits extending them to the identity-based setting.

6.2.1. The certification approach

The easiest way to construct IBS schemes with special properties is to again follow the certification approach from Section 2. A signature then consists of two parts: a standard signature with special properties on the message using a standard secret key, and a certificate that links the corresponding public key to the identity of the user. The latter can be implemented using a standard signature scheme. As shown by Galindo et al. [GHK06], this construction works for many types of identity-based signatures with special properties such as proxy signatures, blind signatures, verifiably encrypted signatures, undeniable signatures, forward-secure signatures, strong key insulated signatures, online/offline signatures, threshold signatures, and aggregate signatures. However, the same approach does not seem to work in settings when additional public keys have to be used in the protocol, different from that of the signer. This includes ring, designated verifier, confirmer, nominative, and chameleon signatures. For these kinds of signatures, therefore, it makes more sense to consider specific constructions in the identity-based framework.

6.2.2. Verifiably encrypted signatures

As an example of IBS schemes with special properties, let us consider ID-based verifiably encrypted signatures. Verifiably encrypted signature (VES) schemes can be seen as a special extension of the standard signature primitive. VES schemes enable the signer to create a signature that is encrypted using an adjudicator's public key (the VES signature), but in such a way that public verification of the signature remains possible. The adjudicator is a trusted third party, who can reveal the plaintext signature when needed. VES schemes provide an efficient way to enable fairness in many practical applications such as contract signing. Compared to a standard signature a VES scheme has three additional algorithms: VES signing/verification (with respect to an adjudicator's public key), and adjudication. Here the adjudication algorithm inputs an adjudicator's secret key and transforms a VES into a standard signature.

Identity-based verifiably encrypted signature (IB-VES) schemes were introduced in [GZ05] where also a concrete instantiation based on bilinear maps was proposed. For the generic certificate-based construction, VES signing and verification can be lifted to the identity-based case using certificates following the techniques from Section 2. IB-VES signing replaces σ with its VES counterpart by running the VES signing algorithm on sk, M, and the adjudicator's public key. IB-VES verification checks the certificate and the VES using the standard VES verification algorithm.

An efficient VES scheme in the random oracle model based on pairings was given in [BGLS03], one in the standard model in [LOS$^+$06]. It was further noted in [LOS$^+$06] that VES schemes can be constructed on general assumptions such as trapdoor one-way permutations. Therefore the generic construction yields an IB-VES scheme based on any trapdoor one-way function [LOS$^+$06], and a more efficient one using [BGLS03].

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-45

Chapter IV

Identity-Based Encryption and Hierarchical Identity-Based Encryption

Sanjit CHATTERJEE [a] and Palash SARKAR [b]
[a] *University of Waterloo, Canada*
[b] *Indian Statistical Institute, India*

Abstract. We take a bird's eye view of different identity-based encryption (IBE) and hierarchical identity-based encryption (HIBE) protocols that are available in the literature and use bilinear pairing as the principal building block. We start with the seminal work on IBE by Boneh-Franklin. The concept of IBE has been generalized to HIBE and we illustrate this with Gentry-Silverberg HIBE. The salient features of the security reductions are also discussed along with the protocols. These initial protocols were proven secure in the adaptive setting using random oracle. A non-adaptive security model called the selective-ID model was developed later and protocols secure in this restricted security model are discussed along with an insider's view on their proof technique. Finally, protocols secure in the adaptive setting without the use of random oracle are introduced and their security discussed. The (H)IBE protocols available in the literature are generally shown to be secure against chosen plaintext attack. Generic as well as endogenous techniques from the existing literature are briefly discussed on how to achieve chosen ciphertext security for these protocols.

In this chapter, we take a look at some of the (hierarchical) identity-based encryption schemes, based on bilinear pairing. Our aim is to trace the conceptual evolution in the field. This being a relatively new area, there are quite a large number of works studying the different aspects of identity-based encryption, including security notions and applications of this primitive to other areas in cryptography. We concentrate on the works that help to place the concepts of IBE and HIBE in a proper context – rather than trying to be encyclopedic.

1. Identity-Based Encryption

The research effort that started with Shamir [Sha85] posing an open problem to construct an IBE came to its fruition in 2001 when Boneh and Franklin [BF03] proposed a practical identity-based encryption scheme using bilinear pairing with a proper security model and a security reduction. Sakai, Ohgishi and Kashahara had also independently proposed an IBE [SOK00] using pairing. The work of Boneh and Franklin brought the immediate attention of the crypto community to IBE.

1.1. Definition and Security Model

An identity-based encryption scheme is specified by four probabilistic polynomial time algorithms:

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

Key Derivation This randomized algorithm takes as input an identity $id \in \mathcal{I}$ together with the system parameters and the master secret key (msk) and returns a private key of the user usk, using the master secret key. The identity id is used as the public key while usk is the corresponding private key.

Encrypt This randomized algorithm takes as input an identity $id \in \mathcal{I}$ and a message $M \in \mathcal{M}$ and produces a ciphertext $C \in \mathcal{C}$ encrypted under the public key id using the master public key mpk.

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

These set of algorithms must satisfy the standard consistency requirement:
For (mpk, msk) output by Setup, let usk be a private key generated by the Key derivation algorithm given input the identity id, then

$$\forall M \in \mathcal{M}\text{: Decrypt(Encrypt(}M, id, mpk), usk, mpk) = M \ .$$

In case of public key encryption, security against adaptive chosen ciphertext attack [BDPR98] is the standard notion of security. Boneh and Franklin extended this notion of security to the identity-based setting terming it as ind-id-cca security.

The ind-id-cca security for an identity-based encryption scheme $\mathcal{IBE}$ is defined in terms of a game between a challenger and an adversary as described bellow. The adversary is allowed to place two types of oracle queries – decryption queries to a decryption oracle $\mathcal{O}_d$ and key-extraction queries to a key-extraction oracle $\mathcal{O}_k$.

Setup The challenger takes input a security parameter k and runs the Setup algorithm of $\mathcal{IBE}$. It provides $\mathcal{A}$ with the common system parameters along with the master public key mpk while keeping the master secret key msk to itself.

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

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

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

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

Phase 2 $\mathcal{A}$ now issues additional queries just like Phase 1, with the (obvious) restriction that it cannot place a decryption query for the decryption of C^* under id^* or a key-extraction query for the private key of id^*. All other queries are valid and $\mathcal{A}$ can issue these queries adaptively just like Phase 1. The challenger responds as in Phase 1. There is a limit on how many private key extraction queries and decryption queries $\mathcal{A}$ can make in Phase 1 and 2 together.

Guess $\mathcal{A}$ outputs a guess γ' of γ.

The advantage of the adversary $\mathcal{A}$ in attacking the IBE scheme is defined as:

$$\mathbf{Adv}_{\mathcal{IBE}}^{\text{ind-id-cca}}(\mathcal{A}) = |\Pr[(\gamma = \gamma')] - 1/2| \ .$$

An IBE scheme $\mathcal{IBE}$ is said to be $(t, q_{id}, q_C, \epsilon)$-secure against adaptive chosen ciphertext attack ($(t, q_{id}, q_C, \epsilon)$-ind-id-cca secure) if for any t-time adversary $\mathcal{A}$ that makes at most q_{id} private key queries and at most q_C decryption queries, $\mathbf{Adv}_{\mathcal{IBE}}^{\text{ind-id-cca}}(\mathcal{A}) \leq \epsilon$. In short, we say $\mathcal{IBE}$ is ind-id-cca secure or when the context is clear, simply cca-secure. A restricted version of the above game is obtained when the adversary is disallowed from making any decryption query, i.e., $q_C = 0$. The encryption scheme is then said to be secure against chosen plaintext attack – ind-id-cpa secure or simply cpa-secure.

1.2. Boneh-Franklin IBE

Boneh and Franklin proposed two versions of their protocol. The first version is secure in the sense ind-id-cpa while the second is secure in the sense ind-id-cca. Both use cryptographic hash functions that are modeled as random oracles in the security reduction. While describing the $\mathcal{BF}$-$\mathcal{IBE}$ as well as the later protocols, we concentrate on protocols achieving security against chosen plaintext attack (i.e., cpa-secure). There are mainly two reasons to do so. Firstly, these cpa-secure protocols and their security reduction capture the essential ideas that we are interested in. Secondly, once we get a cpa-secure IBE, there are standard techniques to achieve cca-security.

For all the protocols that we describe in this chapter it is assumed that, given a security parameter k, the descriptions of groups $\mathbb{G}, \mathbb{G}_T$ and the bilinear map $\hat{e}:\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ are publicly available. The descriptions include polynomial time (in k) algorithms to compute the group operations in $\mathbb{G}, \mathbb{G}_T$ as well as $\hat{e}$. We now describe the ind-id-cpa secure version of $\mathcal{BF}$-$\mathcal{IBE}$ followed by the salient features of the security reduction.

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

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

Encrypt To encrypt $M \in \{0,1\}^n$ to id compute $Q_{id} = H_1(id)$, choose a random $r \in \mathbb{Z}_p^*$ and set the ciphertext:

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

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

$$V \oplus H_2(\hat{e}(d_{id}, U)) = M \ .$$

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

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

Hence the decryption algorithm returns M given a valid encryption for id.

Security of the above scheme is based on the hardness of the computational Bilinear Diffie-Hellman (BDH) problem (i.e., given $P, aP, bP, cP \in \mathbb{G}$ for random $a, b, c \in \mathbb{Z}_p^*$ compute $\hat{e}(P, P)^{abc}$). Assuming $H_1()$ and $H_2()$ to be random oracles, [BF03] proves the ind-id-cpa security of the protocol. The proof is a reduction and proceeds in two steps. In the first step, a public key encryption scheme $\mathcal{PKE}$ is defined as follows.

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

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

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

Suppose, for some identity id in $\mathcal{BF}$-$\mathcal{IBE}$, $H_1(id)$ is mapped to Q_{id} of $\mathcal{PKE}$. Then the Key derivation, Encryption and Decryption algorithms of $\mathcal{PKE}$ essentially corresponds to the respective algorithms of $\mathcal{BF}$-$\mathcal{IBE}$ for the identity id.

Let $\mathcal{A}_1$ be an ind-id-cpa adversary against $\mathcal{BF}$-$\mathcal{IBE}$ and $\mathcal{A}_2$ is an ind-cpa adversary against $\mathcal{PKE}$. In the context of PKE, an ind-cpa adversary is one which, given the public key, produces two messages and is able to distinguish when provided with an encryption of one of the two. Let $\mathcal{B}$ be an algorithm that solves the given BDH problem. The reduction proceeds in two steps. In the first step, which we denote as Game 1, $\mathcal{A}_1$ is used to construct $\mathcal{A}_2$. In the next step, which we call Game 2, $\mathcal{A}_2$ is used to construct $\mathcal{B}$.

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

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

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

Given this intuitive understanding, we now proceed for a more formal description. The system is setup as given in the intuitive description above.

Game 1

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

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

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

Any query on $H_2()$ at this stage is passed on to the H_2 oracle and $\mathcal{A}_2$ returns whatever the oracle returns.

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

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

Let $U = rP$ and $V = M_\gamma \oplus H_2(\hat{e}(Q_{id}, P_{\mathsf{pub}})^r)$ for some random $r \in \mathbb{Z}_p$. $\mathcal{A}_2$ sets $U' = b^{-1}U = b^{-1}rP = r'P$. So $\hat{e}(Q_{id}, P_{\mathsf{pub}})^r = \hat{e}(b^{-1}Q, P_{\mathsf{pub}})^r =$

$\hat{e}(Q, P_{\mathsf{pub}})^{b^{-1}r} = \hat{e}(Q, P_{\mathsf{pub}})^{r'}$. Hence, $C^* = \langle U', V \rangle$ is a proper encryption of M_γ under the identity id^*.

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

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

If $\mathcal{A}_2$ does not abort the above game, then from the view point of $\mathcal{A}_1$ the situation is identical to that of a real attack. Boneh and Franklin show that the probability that $\mathcal{A}_2$ does not abort is $\delta^q(1-\delta)$, where q is the number of key extraction queries and a lower bound is $\Pr\left[\mathsf{abort}\right] \geq 1/(e \times (1+q))$, where e is the base of natural logarithms. Suppose $\mathbf{Adv}^{\text{ind-id-cpa}}_{\mathcal{BF}\text{-}\mathcal{IBE}}(\mathcal{A}_1) \leq \epsilon$, then $\mathcal{A}_2$'s advantage against $\mathcal{PKE}$ is at most $\epsilon/(e \times (1+q))$.

In the next step, we construct an algorithm $\mathcal{B}$ which solves the BDH problem given the adversary $\mathcal{A}_2$ against $\mathcal{PKE}$ relating the advantage of $\mathcal{B}$ with that of $\mathcal{A}_2$. The details of the reduction follow.

Game 2

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

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

***H*₂-queries.** To respond to $\mathcal{A}_2$'s queries to the random oracle $H_2()$, $\mathcal{B}$ maintains a list called H_2^{list}. The ith entry to the list is a tuple $\langle X_i, H_i \rangle \in \mathbb{G}^* \times \{0,1\}^n$. $\mathcal{B}$ responds to any query for X_j in the following manner:

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

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

Guess $\mathcal{A}_2$ outputs its guess $\gamma' \in \{0,1\}$.

$\mathcal{B}$ picks a random tuple $\langle X_i, H_i \rangle$ from H_2^{list} and outputs X_i as the solution of the given BDH problem.

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

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

$$\mathbf{Adv}^{\text{ind-id-cpa}}_{\mathcal{BF}\text{-}\mathcal{IBE}}(\mathcal{A}_1) \leq \frac{1}{2} e \times (1+q) q_{H_2} \mathbf{Adv}_{\mathsf{BDH}}(\mathcal{B}) \ .$$

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

Boneh and Franklin next proposed the chosen ciphertext secure version of their IBE, which is obtained by applying the so called Fujisaki-Okamoto transformation [FO99] to the basic scheme. We do not provide details of the construction. Interested readers are referred to the original work of Boneh and Franklin [BF03] and also to Galindo's work [Gal05b], who fixed a bug in the original reduction. Further work on reducing the degradation in the security reduction of the Boneh and Franklin scheme is also available in the literature [KW03,AFG$^+$06].

2. Hierarchical Identity-Based Encryption

The concept of IBE has been generalized to hierarchical identity-based encryption (HIBE) by Horwitz-Lynn and Gentry-Silverberg [HL02,GS02]. Extending in the line of Boneh and Franklin, Horwitz and Lynn gave precise definitions of HIBE and its security model.

Instead of a single identity component, HIBE allows encryption to an ordered identity tuple. In a HIBE having a maximum of h levels, an entity at a level $j \le h$ has an identity tuple of the form $\vec{id} = \langle id_1, \ldots, id_j\rangle$ and any entity having identity $\langle id_1, \ldots, id_i\rangle$, $i < j$ and that has its own private key, can generate a private key for $\vec{id}$. This way HIBE reduces the work load of the PKG by allowing key delegation by an entity to its lower level entities.

Gentry and Silverberg proposed a HIBE [GS02] scheme which was proven secure using random oracles. The $\mathcal{GS}$-$\mathcal{HIBE}$ bears much resemblance with the $\mathcal{BF}$-$\mathcal{IBE}$ in terms of construction as well as security proof. We now detail their scheme followed by the essential idea of the security reduction.

2.1. $\mathcal{GS}$-$\mathcal{HIBE}$

Setup Let P be an arbitrary generator of $\mathbb{G}$. Pick a random $x_0 \in \mathbb{Z}_p^*$ and set $P_{\mathsf{pub}} = x_0P$. Also choose cryptographic hash functions $H_1{:}\{0,1\}^* \to \mathbb{G}$ and $H_2{:}\mathbb{G}_{\mathrm{T}} \to \{0,1\}^n$. The public parameters are $mpk = \langle P, P_{pub}, H_1, H_2\rangle$, while the master secret is x_0.

Key Derivation An identity $\vec{id}$ at the jth level is represented as $\vec{id} = (id_1, \ldots, id_j)$. The PKG chooses random elements $x_1, \ldots, x_{j-1} \in \mathbb{Z}_p^*$ and computes $d_i = x_iP$ for $1 \le i \le j-1$ and $d_j = \sum_{i=1}^{j} x_{i-1}Q_i$, where $Q_i = H_1(id_1, \ldots, id_i)$. It gives $d_{\vec{id}} = \langle d_1, \ldots, d_j\rangle$ to $\vec{id}$.

A private key for $\vec{id}$ can also be generated by its parent. Let $\vec{id}|_{j-1} = (id_1, \ldots, id_{j-1})$ be the parent of $\vec{id}$, i.e., $\vec{id}|_{j-1}$ is one level up in the hierarchy with respect to $\vec{id}$. Let the private key of $\vec{id}|_{j-1}$ be $d_{\vec{id}|_{j-1}} = (d'_1, \ldots, d'_{j-1})$. Then $d_{\vec{id}}$ can be formed by $\vec{id}|_{j-1}$ as follows: Compute $Q_{\vec{id}} = H_1(id_1, \ldots, id_j)$, choose a random $x_{j-1} \in \mathbb{Z}_p^*$ and set $d_j = d'_{j-1} + x_{j-1}Q_{id}$ and set $d_i = d'_i$ for $1 \le i \le j-2$ and $d_{j-1} = x_{j-1}P$. The private key, $d_{\vec{id}} = \langle d_1, \ldots, d_j\rangle$ is given to $\vec{id}$.

Encrypt To encrypt M under the identity $\vec{id} = (id_1, \ldots, id_j)$, compute $Q_i = H_1(id_1, \ldots, id_i)$ for $1 \le i \le j$. Then choose a random $r \in \mathbb{Z}_p^*$ and set the ciphertext

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

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

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

If the HIBE is restricted to the first level only, then this is exactly the $\mathcal{BF}$-$\mathcal{IBE}$ scheme of the preceding section.

Security of $\mathcal{GS}$-$\mathcal{HIBE}$ against chosen plaintext attack (i.e., ind-id-cpa security) can be proved in the same manner as that of $\mathcal{BF}$-$\mathcal{IBE}$. In the first stage, an ind-id-cpa adversary, $\mathcal{A}_1$ against $\mathcal{GS}$-$\mathcal{HIBE}$ is used to construct an ind-cpa adversary $\mathcal{A}_2$ against $\mathcal{PKE}$ (the same public key encryption scheme defined in the context of $\mathcal{BF}$-$\mathcal{IBE}$). In the next stage, this $\mathcal{A}_2$ is utilized to construct an algorithm $\mathcal{B}$ that solves the BDH problem.

However, Gentry and Silverberg first prove the security in the non-adaptive setting and later extend it to the adaptive setting. In the non-adaptive setting, $\mathcal{A}_1$ a-priori fixes a target identity $\vec{id}^* = (id_1^*, \ldots, id_j^*)$, $j \ge 1$. Given $\vec{id}^*$, $\mathcal{A}_2$ forms the H_1^{list} in such a way that given any identity $\vec{id}$ it can generate the private key of $\vec{id}$, provided $\vec{id}$ is not a prefix of $\vec{id}^*$. Similarly in the challenge phase, it can generate a proper encryption of M_γ under $\vec{id}^*$, given a proper encryption of M_γ in $\mathcal{PKE}$. This way, the advantage of $\mathcal{A}_1$ against $\mathcal{GS}$-$\mathcal{HIBE}$ can be directly converted into the advantage of $\mathcal{A}_2$ against $\mathcal{PKE}$ without any degradation.

The security reduction in the adaptive setting, however, suffers from a large degradation factor. Let's briefly see why this is so. Here, instead of a single identity, we have an identity tuple of arbitrary levels. Recall that in the reduction for $\mathcal{BF}$-$\mathcal{IBE}$ of Section 1, a typical entry in H_1^{list} is of the form $\langle id, Q, b, c \rangle$. In case of $\mathcal{GS}$-$\mathcal{HIBE}$ an entry for $\vec{id} = (id_1, \ldots, id_j)$ is of the form $\langle id_i \rangle, \langle Q_{id_i} \rangle, \langle b_i \rangle, \langle c_i \rangle$ where $1 \le i \le j$, i.e., each $\langle \cdots \rangle$ contains j many entries, whereas in case of $\mathcal{BF}$-$\mathcal{IBE}$ of Section 1 it contained only a single term.

Let the target identity be $\vec{id}^* = (id_1^*, \ldots, id_h^*)$ then the corresponding entry in the H_1^{list} has terms $c_1, \ldots, c_h \in \{0,1\}^h$. So, instead of a single c, we have to maintain h many c_is in H_1^{list}. $\mathcal{A}_2$ can generate a proper challenge ciphertext if and only if all these h c_is have the same value, 1. But these c_is are chosen independently at random, so the probability that all are 1 is the product of the probabilities that each is 1. Hence, we get a security degradation which is exponential in the number of levels in the target identity tuple.

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

$$\mathbf{Adv}^{\text{ind-id-cpa}}_{\mathcal{GS}\text{-}\mathcal{HIBE}}(\mathcal{A}_1) \le \frac{(e \times (q+h))^h q_{H_2}}{h} \mathbf{Adv}_{\mathsf{BDH}}(\mathcal{B})$$

where e is the base of natural logarithm.

This means, if there is an ind-id-cpa adversary $\mathcal{A}_1$ in the adaptive setting having advantage ϵ against $\mathcal{GS}$-$\mathcal{HIBE}$ and that makes at most q_{H_2} queries to H_2 and q private key extraction queries and H_1, H_2 are random oracles then there is an algorithm $\mathcal{B}$ that solves the BDH problem in $\langle \mathbb{G}, \mathbb{G}_T \rangle$ with advantage roughly $(\epsilon(q+h)^{-h} q_{H_2}^{-1})$, where h is the number of levels in the target identity. So the security degrades exponentially with the number of levels of the HIBE.

Applying the Fujisaki-Okamoto transformation to this basic scheme, Gentry and Silverberg obtain an ind-id-cca secure HIBE, the construction as well as the security proof of which is analogous to that of $\mathcal{BF}$-$\mathcal{IBE}$ and not detailed here. Similarly, Galindo's observation [Gal05b] with respect to $\mathcal{BF}$-$\mathcal{IBE}$ is also applicable for $\mathcal{GS}$-$\mathcal{HIBE}$.

3. From Random Oracle to Standard Model

Both $\mathcal{BF}$-$\mathcal{IBE}$ and $\mathcal{GS}$-$\mathcal{HIBE}$ depend on the random oracle heuristic for their proof of security. The first move to obtain an IBE without random oracle was taken by Canetti, Halevi and Katz [CHK03,CHK04]. However, they had to weaken the security notion from adaptive to the so called *selective*-ID model while moving in this direction. In this model an adversary has to commit to a target identity even before the system is set up.

We have already observed that two random oracles (namely H_1, H_2) are used to prove ind-id-cpa security of $\mathcal{BF}$-$\mathcal{IBE}$ and $\mathcal{GS}$-$\mathcal{HIBE}$. We need two other random oracles, H_3 and H_4 to achieve cca-security for the above protocols. These four random oracles serve three distinct purposes. H_1 is used to map an identity to a random element of $\mathbb{G}$, H_2 is used to relate the security of the scheme with that of BDH problem while H_3, H_4 come into play because of the Fujisaki-Okamoto transformation to achieve chosen ciphertext security from a given chosen plaintext secure scheme. One can remove H_1 and H_2 by suitably modifying the identity space and the message space and by relying on the (presumably stronger) decisional bilinear Diffie-Hellman assumption (DBDH). Canetti, Halevi and Katz [CHK04] proposed a different generic transformation to achieve cca-security, thereby removing the need to rely on random oracles H_3, H_4. The efficiency of this generic transformation was later improved by Boneh and Katz [BK05]. We discuss this generic transformation in Section 6. Here we concentrate on constructions that achieve security against chosen plaintext attack under the *selective*-ID model.

By restricting the adversary's behavior, it is possible to construct protocols secure in the *selective*-ID model that avoid any degradation in the security reduction. $\mathcal{GS}$-$\mathcal{HIBE}$ [GS02] secure in the non-adaptive setting can be seen as secure in the *selective*-ID model, though the model was not formalized when the scheme was first proposed.

Canetti, Halevi and Katz [CHK03] introduced the notion of binary tree encryption (BTE). A BTE differs from HIBE in the sense that each node in the BTE has two children – the left child and the right child. BTE is a public key encryption scheme and they also show [CHK03] how an (H)IBE can be constructed from any BTE. This gives the first (H)IBE construction that is secure without random oracle in the *selective*-ID model. We do not detail their construction and security proof here. One limitation of the construction is large computational overhead – one pairing computation for each bit of identity during decryption.

3.1. Selective-*ID Secure HIBE*

Boneh and Boyen [BB04a] proposed two efficient IBE schemes that are secure in the *selective*-ID model without random oracle. The first construction was presented as a HIBE which we call $\mathcal{BB}$-$\mathcal{HIBE}$. We give a description of $\mathcal{BB}$-$\mathcal{HIBE}$ and its security reduction below. The algebraic technique introduced in this work for private key extraction was adapted in later works on (H)IBE. This protocol has later been generalized to give two new constructions in [CS06a].

3.1.1. $\mathcal{BB}$-$\mathcal{HIBE}$

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

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

$$mpk = \langle P, P_1, P_2, Q_1, \ldots, Q_h \rangle; \quad msk = xP_2 \ .$$

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

Key Derivation Given an identity $\vec{id} = \langle id_1, \ldots, id_j \rangle$ of depth $j \leq h$, pick random $r_1, \ldots, r_j \in \mathbb{Z}_p$ and compute

$$d_{\vec{id}} = \left(xP_2 + \sum_{i=1}^{j} r_i F_i(id_i), r_1 P, \ldots, r_j P \right) \ .$$

$d_{\vec{id}}$ can also be generated given the private key $d_{\vec{id}|j-1}$ of $\vec{id}|_{j-1} = \langle id_1, \ldots, id_{j-1} \rangle$ as shown below.

Let, $d_{\vec{id}|j-1} = (d'_0, d'_1, \ldots, d'_{j-1})$ be the private key of $\vec{id}|_{j-1}$. Given the identity $\vec{id} = \langle id_1, \ldots, id_j \rangle$, $\vec{id}|_{j-1}$ chooses a random $r_j \in \mathbb{Z}_p$ and computes

$$d_0 = d'_0 + r_j F_j(id_j)$$
$$d_j = r_j P$$

and $d_i = d'_i$ for $1 \leq i \leq j-1$. The resulting $d_{\vec{id}} = (d_0, d_1, \ldots, d_j)$ is a valid private key for the identity $\vec{id}$.

Encrypt Encrypt $M \in \mathbb{G}_T$ for $\vec{id} = \langle id_1, \ldots, id_j \rangle$ as

$$C = (\hat{e}(P_1, P_2)^s \times M, sP, sF_1(id_1), \ldots, sF_j(id_j))$$

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

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

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

3.1.2. Security

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

In the following, we consider security against chosen plaintext attack in the *selective*-ID model, i.e., ind-sid-cpa security only.

Initialization $\mathcal{A}$ commits to a target identity $\vec{id}^* = \langle id_1^*, \ldots, id_{h'}^*\rangle$ of height $h' \le h$. If $h' < h$, $\mathcal{B}$ adds extra random elements from $\mathbb{Z}_p$ to make $\vec{id}^*$ an identity of height h. Let us denote these extra $(h - h')$ elements by $id_{h'+1}^*, \ldots, id_h^*$.

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

Phase 1 $\mathcal{A}$ makes up to q private key queries. In a private key query corresponding to an identity $\vec{id} = \langle id_1, \ldots, id_u\rangle$, with $u \le h$ the only restriction is that $\vec{id}$ is not a prefix of $\vec{id}^*$. Let, j be the smallest index such that $id_j \ne id_j^*$. $\mathcal{B}$ chooses random $r_1, \ldots, r_j \in \mathbb{Z}_p$ and first computes

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

where $\tilde{r}_j = r_j - \frac{b}{id_j - id_j^*}$. So $\mathcal{B}$ forms the private key of $\langle id_1, \ldots, id_j\rangle$ as

$$d_0 = d_{0|j} + \sum_{i=1}^{j-1} r_i F_i(id_i), \quad d_1 = r_1 P, \quad \ldots, \quad d_{j-1} = r_{j-1}P,$$

$$d_j = -\frac{1}{id_j - id_j^*} P_2 + r_j P = \tilde{r}_j P\,.$$

It is easy to verify that $\langle d_0, d_1, \ldots, d_j \rangle$ is a valid private key for $\langle id_1, \ldots, id_j \rangle$. From this $\mathcal{B}$ forms a private key for $\vec{id}$ and returns it to $\mathcal{A}$.

Note that, $\mathcal{B}$ can derive a valid private key for an identity $\vec{id}$ without the knowledge of the master secret. This is possible as long as $\vec{id}$ is not a prefix of $\vec{id}^*$. The above algebraic technique of private key derivation is one of the major technical novelties of the work by Boneh and Boyen. Also note that, if the original target identity $\vec{id}^* = (id_1^*, \ldots, id_{h'}^*)$ is of height less than h, then $\vec{id} = (id_1^*, \ldots, id_{h'}^*, id_{h'+1}^*, \ldots, id_h^*)$ can be a valid query for private key extraction. In such a case, $\mathcal{B}$ has to abort the game. The probability of this event is, however, very low – of the order q/p.

Challenge After completion of Phase 1, $\mathcal{A}$ outputs two messages $M_0, M_1 \in \mathbb{G}_T$. $\mathcal{B}$ chooses a random bit γ and forms the ciphertext $C = \langle M_\gamma \times Z, cP, \alpha_1 cP, \ldots, \alpha_{h'} cP \rangle$. Note that, $F_i(id_i^*) = \alpha_i P$, so

$$C = \langle M_\gamma \times Z, cP, cF_1(id_1^*), \ldots, cF_{h'}(id_{h'}^*) \rangle \ .$$

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

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

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

When $Z = \hat{e}(P, P)^{abc}$, then $\mathcal{A}$'s view in the above game is identical to that in a real attack. On the other hand if Z is a random element of $\mathbb{G}_T$ then $\Pr[\gamma = \gamma'] = 1/2$. Hence we get,

$$\mathbf{Adv}^{\text{ind-sid-cpa}}_{\mathcal{BB}\text{-}\mathcal{HIBE}}(\mathcal{A}) \leq \mathbf{Adv}_{\mathsf{DBDH}}(\mathcal{B}) \ .$$

In other words, if the (t, ϵ)-DBDH assumption holds in $\mathbb{G}, \mathbb{G}_T$ then $\mathcal{BB}$-$\mathcal{HIBE}$ is (t', q, ϵ)-ind-sid-cpa secure for arbitrary h and q and any $t' < t - O(\tau hq)$ where τ is the time for a scalar multiplication in $\mathbb{G}$.

4. From Selective to Adaptive Model (*without* Random Oracle)

Boneh and Boyen were first to propose an IBE [BB04b] whose proof of security in the adaptive model does not rely on the random oracle heuristic. The construction, however, is not very efficient and as the authors observed, should be seen as a proof of concept. We reproduce their construction because of its chronological importance.

4.1. $\mathcal{BB}$-$\mathcal{IBE}$

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

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

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

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

$$C = \langle \hat{e}(P_1, P_2)^s \times M, sP, sU_{1,a_1}, \ldots, sU_{n,a_n}\rangle \in \mathbb{G}_{\mathrm{T}} \times \mathbb{G}^{n+1} \ .$$

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

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

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

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

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

4.2. $\mathcal{W}aters$-$\mathcal{IBE}$

The public parameters, private key and ciphertext sizes are quite large for $\mathcal{BB}$-$\mathcal{IBE}$. Waters introduced a nice protocol [Wat05] where the private key and ciphertext sizes are drastically reduced. In this protocol, identities are represented as bit strings of length n.

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

Key Derivation Let $id = (id_1, \ldots, id_n) \in \{0,1\}^n$ be any identity. A secret key for id is generated as follows. Choose a random $r \in \mathbb{Z}_p^*$, then the private key for id is

$$d_{id} = (xP_2 + rV, rP)$$

where

$$V = U' + \sum_{\{i: id_i = 1\}} U_i \ .$$

Encrypt Any message $M \in \mathbb{G}_\mathrm{T}$ is encrypted for an identity id as

$$C = (\hat{e}(P_1, P_2)^t M, tP, tV)\,,$$

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

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

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

Waters' construction has some similarity with the $\mathcal{BB}$-$\mathcal{HIBE}$ of Section 3. Using a similar algebraic technique of Boneh-Boyen, Waters forms a simulator $\mathcal{B}$, that solves the DBDH problem given an adversary $\mathcal{A}$ against the IBE.

$\mathcal{B}$ is provided with a DBDH problem instance, i.e., $\langle P, aP, bP, cP, Z\rangle$ and its task is to decide whether $Z = \hat{e}(P,P)^{abc}$ or not. To play the ind-id-cpa game with $\mathcal{A}$, $\mathcal{B}$ chooses random $x, x_1, \ldots, x_n \in \mathbb{Z}_m$ where $m = 4q$, q being the maximum number of key extraction queries; random $y, y_1, \ldots, y_n \in \mathbb{Z}_p$ and a random $k \in \{0, \ldots, n\}$. It then defines functions: $F(v) = p - mk + x + \sum_{i=1}^{n} x_i v_i$, $J(v) = y + \sum_{i=1}^{n} y_i v_i$. Next, $\mathcal{B}$ assigns $P_1 = aP$, $P_2 = bP$, $U' = (p - mk + x)P_2 + yP$ and $U_i = x_i P_2 + y_i P$ for $1 \le i \le n$. It provides $\mathcal{A}$ with the public parameters $\langle P, P_1, P_2, U', U_1, \ldots, U_n\rangle$, which is a valid set of parameters.

In response to $\mathcal{A}$'s private key extraction query for an identity, id, $\mathcal{B}$ forms,

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

where r is chosen at random from $\mathbb{Z}_p$. This is a valid private key for id (check this using the technique of $\mathcal{BB}$-$\mathcal{HIBE}$!) and $\mathcal{B}$ can compute this only if $F(id) \neq 0$. In contrast, it can form a proper challenge for an identity id^* only if $F(id^*) = 0$. This way an adversary against the IBE can be used to construct an algorithm to solve the DBDH problem.

We have already observed that this complementary condition for key generation and challenge generation is a hallmark of all the encryption protocols described so far. Because of this complementary condition there are certain identities for which the simulator cannot generate the private key and for some other identities it is unable to generate a proper challenge. In such situations, the simulator has to abort the game and this is the cause of degradation in the security reduction. In Waters' case the degradation is of the order of the maximum number of key extraction queries. There are, however, other complications to tackle. The probability of abort may be correlated to the adversary's probability of success. As a result, the event that the adversary is successful cannot be directly converted into the event that the instance of the DBDH problem is solved. To overcome this, Waters introduced the technique of artificial abort, whereby the simulator goes through an additional phase which does not involve interaction with the adversary. In this phase, the simulator may choose to abort, so that the probability of abort is almost the same irrespective of any set of queries made by the adversary. This technique of artificial abort introduces a new element of probabilistic reasoning in security proofs.

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

$\mathcal{W}aters\text{-}\mathcal{IBE}$ has been generalized independently by Chatterjee-Sarkar [CS05] and Naccache [Nac05]. Chatterjee-Sarkar also proposed a HIBE in the standard model [CS06b] which significantly reduces the size of the public parameter. However, the problem of constructing a HIBE in the adaptive model (with or without random oracle), where the security degradation is less than exponential in the number of levels, still remains open.

5. HIBE with Shortened Ciphertext

5.1. Constant-Size Ciphertext HIBE

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

Let the maximum depth of the HIBE be h. Identities at a depth $u \leq h$ are of the form $\vec{id} = (id_1, \ldots, id_j) \in (\mathbb{Z}_p^*)^j$. Messages are elements of $\mathbb{G}_T$.

Setup Let P be a generator of $\mathbb{G}$. Choose a random $\alpha \in \mathbb{Z}_p$ and set $P_1 = \alpha P$. Choose random elements $P_2, P_3, Q_1, \ldots, Q_h \in \mathbb{G}$. Set the public parameter as $mpk = (P, P_1, P_2, P_3, Q_1, \ldots, Q_h)$ and the master key as $msk = \alpha P_2$.

Key Derivation Given an identity $\vec{id} = (id_1, \ldots, id_j) \in (\mathbb{Z}_p^*)^j$ of depth $j \leq h$, pick a random $r \in \mathbb{Z}_p$ and output

$$d_{\vec{id}} = (\alpha P_2 + r(id_1 Q_1, \ldots, id_k Q_j + P_3), rP, rQ_{j+1}, \ldots, rQ_h) \ .$$

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

Encrypt To encrypt $M \in \mathbb{G}_T$ under the identity $\vec{id} = (id_1, \ldots, id_j) \in (\mathbb{Z}_p^*)^j$, pick a random $s \in \mathbb{Z}_p$ and output

$$C = (\hat{e}(P_1, P_2)^s \times M, sP, s(id_1 Q_1 + \ldots + id_j Q_j + P_3)) \ .$$

Decrypt To decrypt $C = (C_1, C_2, C_3)$ using the private key $d_{\vec{id}} = (a_0, a_1, b_{j+1}, \ldots, b_h)$, compute

$$C_1 \times \frac{\hat{e}(a_1, C_3)}{\hat{e}(C_2, a_0)} = M \ .$$

Note that, apart from the masked message, the ciphertext in $\mathcal{BBG}$-$\mathcal{HIBE}$ consists of only two elements of $\mathbb{G}_T$ irrespective of the number of components in the corresponding identity. In other HIBEs, the length of the ciphertext is proportional to the length of the identity tuple. The $\mathcal{BBG}$-$\mathcal{HIBE}$ offers new and important applications for constructing other cryptographic primitives like forward secure encryption [CHK03] and broadcast encryption [NNL01,DF03a].

Security of $\mathcal{BBG}$-$\mathcal{HIBE}$ against adaptive chosen plaintext attack is proved in the selective-ID model under the so called decisional h-wDBDHI* assumption (given $(P, Q, aP, a^2P, \ldots, a^hP, Z)$, here the task is to decide whether $Z = \hat{e}(P, Q)^{a^{h+1}}$ or Z is random). The security reduction uses an algebraic technique similar to that of $\mathcal{BB}$-$\mathcal{HIBE}$.

5.2. Composite HIBE

Boneh, Boyen and Goh also suggested a “product” construction of $\mathcal{BBG}$-$\mathcal{HIBE}$ and $\mathcal{BB}$-$\mathcal{HIBE}$ [BBG05]. In case of $\mathcal{BBG}$-$\mathcal{HIBE}$ the private key size decreases with the increase in identity level. While in case of $\mathcal{BB}$-$\mathcal{HIBE}$ the private key size increases with the height of an identity. Utilizing the algebraic similarities of both the systems they construct a composite scheme where the inner HIBE is the $\mathcal{BBG}$-$\mathcal{HIBE}$ and the outer HIBE is the $\mathcal{BB}$-$\mathcal{HIBE}$. The composite scheme allows a trade-off between the ciphertext size and the private key size. A variant of $\mathcal{BBG}$-$\mathcal{HIBE}$ and of this composite construction is also available in the literature [CS07].

6. Chosen-Ciphertext Security

Security against chosen-ciphertext attack (ind-id-cca security) is the strongest notion of security for any (hierarchical) identity-based encryption scheme. We have already observed that the initial proposals such as the $\mathcal{BF}$-$\mathcal{IBE}$ and $\mathcal{GS}$-$\mathcal{HIBE}$ used the Fujisaki-Okamoto transformation to their basic schemes secure in the sense of ind-id-cpa to achieve this goal. However, the Fujisaki-Okamoto transformation uses cryptographic hash functions that are modeled as random oracles. Protocols that achieve security against chosen-plaintext attack without random oracle require a different technique to achieve chosen-ciphertext security.

Canetti, Halevi and Katz introduced a generic transformation [CHK04] to achieve cca-security. Given any $(h + 1)$ level HIBE which is secure against chosen-plaintext attack, this generic transformation yields an h level HIBE which is secure against chosen ciphertext attack. This transformation uses a strongly unforgeable one time signature scheme. Boneh and Katz suggested a modification [BK05] where the one time signature scheme is replaced by a MAC, thereby increasing the efficiency of the transformation.

Here we detail the signature based approach. Recall that a signature scheme is defined by three probabilistic polynomial time algorithms as follows:

Key Derivation On input the security parameter k, this probabilistic polynomial time algorithm outputs a pair of signing key (sk) and verification key (vk).

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

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

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

1. Key derivation(1^k) outputs (vk, sk). The adversary $\mathcal{A}$ is given vk.
2. $\mathcal{A}(1^k, vk)$ may take one of the following actions:
   (a) $\mathcal{A}$ outputs a pair (M^*, σ^*) and halts. In this case (M, σ) is undefined.
   (b) $\mathcal{A}$ outputs a message M and in return is given a signature of M under the signing key sk, i.e., $\sigma \leftarrow \mathrm{S}ign_{sk}(M)$. Then $\mathcal{A}$ outputs a pair (M^*, σ^*).

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

Let $\mathcal{H}' = (\mathsf{Setup}, \mathsf{Der}', \mathcal{E}', \mathcal{D}')$ be an $(h+1)$-HIBE for arbitrary $h \geq 1$ handling $(n+1)$-bit identities. Let $\mathsf{Sig} = (\mathsf{Key\ Derivation}, \mathsf{Sign}, \mathsf{Verify})$ be a signature scheme which outputs an n-bit signature. If $\mathcal{H}'$ is secure in the sense ind-sid-cpa and Sig is a strong one-time signature scheme, then one can construct an h-HIBE secure in the sense ind-sid-cca that handles n-bit identities. Given an identity tuple $\vec{id} = (id_1, \ldots, id_j) \in (\{0,1\}^n)^j$ of $\mathcal{H}$ we map it to an identity tuple of $\mathcal{H}'$ as

$$\mathsf{Encode}(\vec{id}) = (0id_1, \ldots, 0id_j) \in (\{0,1\}^{n+1})^j$$

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

Construction of $\mathcal{H}$

Setup Same as the Setup algorithm of $\mathcal{H}'$. The master key of $\mathcal{H}'$, $msk_{\mathcal{H}'}$ is the master key, $msk_{\mathcal{H}}$ of $\mathcal{H}$.

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

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

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

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

Security Given an identity tuple $\vec{id} = (id_1, \ldots, id_j)$, $j \leq h$ in $\mathcal{H}$, the sender encrypts the message M to a $(j+1)$ level identity $\hat{id} = (\mathsf{Encode}(\vec{id}), (1vk))$ of $\mathcal{H}'$ where vk is the verification key of the underlying signature scheme. The receiver having identity $\vec{id}$ first derives the private key of $\hat{id}$ in $\mathcal{H}'$ from the private key $d_{\vec{id}}$ in $\mathcal{H}$ using the key generation algorithm of $\mathcal{H}'$. We assume that the probability of forging a signature, $\Pr[\mathsf{Forge}]$ is negligible. Then by a reductionist security argument Boneh, Canetti, Halevi and Katz show that if there is an ind-sid-cca adversary $\mathcal{A}$ against $\mathcal{H}$ in the *selective*-ID model then one can construct an ind-sid-cpa adversary $\mathcal{A}'$ against $\mathcal{H}'$ in the same model. The argument runs as follows:

1. $\mathcal{A}'$ runs the ind-sid-cca adversary $\mathcal{A}$ which outputs a target identity tuple $\vec{id}^* = \langle id_1^*, \ldots, id_j^*\rangle$, $j \leq h$. $\mathcal{A}'$ next runs the key generation algorithm **Key-Gen** of **Sig** to generate (vk^*, sk^*). It outputs $\mathsf{Encode}(\vec{id}^*)$, $(1vk^*)$ as its target identity.
2. The challenger gives $\mathcal{A}'$ the public parameter mpk, which it relays to $\mathcal{A}$.
3. $\mathcal{A}$ asks for the private key of an identity $\vec{id}$ which is not a prefix of $\vec{id}^*$. $\mathcal{A}'$ asks its challenger for the private key $d_{\hat{id}}$ where $\hat{id} = \mathsf{Encode}(\vec{id})$ and returns it to $\mathcal{A}$.
4. For a decryption query of the form $(\vec{id}, \langle vk, C, \sigma\rangle)$ from $\mathcal{A}$, $\mathcal{A}'$ takes the following action:

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

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

5. In the challenge stage $\mathcal{A}$ outputs two messages M_0, M_1. The same messages are also output by $\mathcal{A}'$. It receives a challenge ciphertext C^*. Now $\mathcal{A}'$ computes $\sigma^* = \mathsf{Sign}_{sk^*}(C^*)$ and returns the ciphertext $\langle vk^*, C^*, \sigma^*\rangle$ to $\mathcal{A}$.
6. In phase 2, $\mathcal{A}$ makes additional decryption queries and private key extraction queries. These queries are answered as before.
7. Finally $\mathcal{A}$ outputs its guess γ'. The same γ' is output by $\mathcal{A}'$.

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

Note that, if $\mathcal{H}'$ is adaptive chosen plaintext secure in the full model (i.e., ind-id-cpa secure), then $\mathcal{H}$ will be adaptive chosen ciphertext secure (i.e., ind-id-cca secure) in the full model.

Given this generic transformation to achieve cca-security, protocol designers generally concentrate on constructing protocols that achieve cpa-security (be it in the full model or the selective-ID model) without random oracles and then apply this transformation to achieve cca-security. Protocols such as the *BB-HIBE* of Section 3.1 and the *BBG-HIBE* of Section 5.1 accomplish this in the selective-ID model, while the *Waters-IBE* of Section 4 accomplishes this in the full model.

Boyen, Mei and Waters proposed an endogenous transformation to achieve cca-security [BMW05]. Their technique uses the structure of the underlying HIBE and thereby avoids the use of one time signature or MAC. When applied to *BB-HIBE* or *BBG-HIBE* this technique yields a selective-ID cca-secure hierarchical identity-based key encapsulation mechanism (HIB-KEM). For Waters protocol or its modification to HIBE, an adaptive identity cca-secure (H)IBE has been obtained in [KG06,SC07] by using the above technique.

7. Inversion-based IBE

The (H)IBE schemes described in Sections 3, 4 and 5 use a proof technique originally developed by Boneh and Boyen in [BB04a]. This is referred to as commutative-blinding IBE framework. Other IBE schemes are also available in the literature which do not fall under this category. Examples are the IBE scheme proposed by Sakai and Kasahara [SK03] and the second selective-ID secure IBE scheme of Boneh and Boyen [BB04a]. Gentry proposed [Gen06] another IBE which has been proven secure in the standard model. These IBE schemes fall under a different paradigm called the inversion-based IBE framework. Here we outline the *Gentry-IBE*.

Setup Let P be a generator of $\mathbb{G}$. Choose a random $\alpha \in \mathbb{Z}_p$ and set $P_1 = \alpha P$. Also choose a random $P_2 \in \mathbb{G}$. Set the public parameter as $mpk = (P, P_1, P_2)$ and the master key is α.

Key Derivation Given an identity $id \in \mathbb{Z}_p$, the PKG generates a random $r_{id} \in \mathbb{Z}_p$ and computes the private key

$$(d_1, d_2) = \left(\frac{1}{\alpha - id}(P_2 - r_{id}P), r_{id} \right) .$$

Encrypt To encrypt $M \in \mathbb{G}_T$ under the identity id, pick a random $s \in \mathbb{Z}_p$ and output

$$C = (\hat{e}(P, P_2)^{-s} \times M, \hat{e}(P, P)^s, s(P_1 - idP)) .$$

Decrypt Given $C = (C_1, C_2, C_3)$ encrypted under id, use the secret key $d_{id} = (d_1, d_2)$ to retrieve M as

$$C_1 \times C_2^{d_2} \times \hat{e}(C_3, d_1) .$$

Note that, $C_2^{d_2}\hat{e}(C_3, d_1) = \hat{e}(P, P)^{sr_{id}}\hat{e}(s(\alpha - id)P, \frac{1}{\alpha - id}(P_2 - r_{id}P)) = \hat{e}(P, P_2)^s$. So, if the encryption is proper then this indeed recovers M.

Security of the above scheme depends on the hardness of the so called decisional truncated q-ABDHE problem. Given $P, Q, aP, a^2P, \ldots, a^qP, a^{q+2}Q, Z$ here the task is to decide whether $Z = \hat{e}(P, Q)^{a^{q+1}}$ or Z is random. In the reduction, q is the maximum number of private key extraction queries allowed to the adversary. Using this assumption Gentry proved that the above IBE is ind-id-cpa secure in the standard model without any degradation. However, construction of an IBE in the standard model with a tight security reduction from a standard assumption still remains an open problem.

8. IBE without Pairing

An elegant solution to the problem of constructing an IBE was given by Cocks [Coc01]. A main feature of this IBE is the fact that it does *not* use bilinear maps. Briefly, the idea is the following. The description is based on the description given in [BGH07a].

The public parameters of the system consists of $N = pq$; a random u from $J(N) \setminus QR(N)$; and a hash function $H()$ which maps identities into $J(N)$. Here $J(N)$ is the set of all elements having Jacobi symbol equal to 1 modulo N and $QR(N)$ is the set of all quadratic residues modulo N.

The secret key corresponding to an identity id is obtained by first computing $R = H(id)$ and then r to be either $\sqrt{R}$ or $\sqrt{uR}$ according as R is a square modulo N or not. The secret key for id is r.

To encrypt a bit m using an identity id, compute R from id as above; randomly choose t_0, t_1 from $\mathbb{Z}_N$ and compute $d_a = (t_a^2 + u^a R)/t_a$ and $c_a = m \cdot (\frac{t_a}{N})$. The ciphertext consists of the two elements $((d_0, c_0), (d_1, c_1))$.

For decryption using identity id and secret key r, first set $a \in \{0, 1\}$ such that $r^2 = u^a R$, where R is obtained from id as above. Set $g = d_a + 2r$. Note that $g = (\frac{(t_a+r)^2}{t_a})$ and hence, $(\frac{g}{N}) = (\frac{t_a}{N})$. So, the receiver can compute $m = c_a \cdot (\frac{g}{N})$.

While this is an interesting protocol, one main problem with it is that the size of the ciphertext is very large. Four elements of $\mathbb{Z}_N$ are produced per bit of the message. Recently, Boneh, Gentry and Hamburg [BGH07a] have given a non-pairing based IBE which is space efficient. The construction is described in two parts. In the first part an IBE is described which encrypts a single bit. This is a general description of which the Cocks-IBE is *not* an instantiation. In the second part, they show how to reuse the random elements for more than one bit. It is such reuse which significantly reduces the size of the ciphertext. The trade-off is a substantial increase in encryption time. Subsequent research is expected to strike a better balance between encryption time and ciphertext size.

9. Summary

In this chapter, we reviewed some of the important (hierarchical) identity-based encryption protocols proposed in the literature. Along with the protocols, in some cases we have discussed the salient features of the proof technique while an intuitive justification of the proof is given for some others. These protocols are proven secure against chosen plaintext attack. Methods for achieving cca-security are also mentioned. Our aim was not an exhaustive survey of the area, but an exposition of the basic issues in identity-based encryption, developments in the proof techniques, the prospects as well as the problems. This being an emerging research area, new concepts are still evolving and will continue to evolve in the near future.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The author and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-65

Chapter V

Flexible IBE and Beyond in the Commutative-Blinding Framework

Xavier BOYEN
Voltage Inc., USA

Abstract. The cryptographic community has, of late, shown much inventiveness in the creation of powerful new IBE-like primitives that go beyond the basic IBE notion and extend it in many new directions. Virtually all of these "super-IBE" schemes rely on bilinear pairings for their implementation, which they tend to use in a surprisingly small number of different ways: three of them as of this writing.

What is interesting is that, among the three main frameworks that we know of so far, one has acted as a veritable magnet for the construction of many of these "generalized IBE" primitives, whereas the other two have not been nearly as fruitful in that respect. This refers to the Commutative Blinding framework defined by the Boneh-Boyen $\mathcal{BB}_1$ IBE scheme from 2004.

The aim of this chapter is to try to shed some light on this approach's popularity, first by comparing its key properties with those of the competing frameworks, and then by providing a number of examples that illustrate how those properties have been used.

Since the first practical constructions of the identity-based encryption (IBE) primitive appeared a few years ago [SOK00,Coc01,BF01], a large body of work has been devoted to creating better realizations of the basic primitive, and to extending it in many interesting ways, involving secret sharing, hierarchies, attributes, etc. With the exception of a few basic IBE schemes [Coc01,BGH07b,GPV07], virtually all IBE-like constructions known to date make more or less extensive use of bilinear pairings on elliptic curves.

The many extensions that have been proposed in the recent years have the common goal to extend the notion of identity from its original atomic meaning, to complex constructs of identity components on which certain operations can be performed, such as hierarchical, fuzzy, and attribute-based identities [HL02,SW05,GPSW06]. What is interesting is that the vast majority of these extensions all fall under the umbrella of one specific family of IBE structures, that of the first Boneh-Boyen ($\mathcal{BB}_1$) scheme, at the exclusion of a few other and very different families. The aim of this chapter is to attempt to shed some light on the properties that made this particular family so successful — at least by the benchmark of flexibility.

Parts of the introductory material through Section 2 were originally published in [Boy08].

1. Preliminaries

Although not all IBE schemes require pairings, all the ones that are known to have interesting extensions beyond plain IBE do require them. For self-containment and consistency of notation, we briefly redefine here the basic notions of bilinear groups and pairings.

Consider a function $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are cyclic groups of prime order p, and $\mathbb{G}_t$ is another group also of order p. Let $P \in \mathbb{G}_1$ and $\hat{P} \in \mathbb{G}_2$ be generators of their respective groups. The function e is a bilinear map or pairing if it satisfies the following conditions:

– *Non-degeneracy:* $e(P, \hat{P}) \neq 1 \in \mathbb{G}_t$.
– *Bilinearity:* $\forall a, b \in \mathbb{Z}, e(aP, b\hat{P}) = e(P, \hat{P})^{ab}$.

The groups $\mathbb{G}_1$ and $\mathbb{G}_2$ are called the bilinear group(s), and $\mathbb{G}_t$ is the target group.

Such pairings can be instantiated using the Weil or Tate pairing on certain types of algebraic curves over finite fields. It is beyond the scope of this chapter to detail how this is done, but the interested reader may refer to [BSS99,Mil04,BSS05,GPS08] and the references therein for details.

Types of Pairings For historical reasons, perhaps because the original Boneh-Franklin scheme required it, the pairing abstraction most commonly used in research papers assumes that the two arguments are interchangeable, and thus that the two input groups $\mathbb{G}_1$ and $\mathbb{G}_2$ are the same. This is the *symmetric* pairing, written $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_t$. This has the advantage of simplifying the notation (no "hat" on the second input group), but unfortunately restricts our choice of groups and prevents us from using the most efficient pairing implementations for schemes that do not require such interchangeability. It is therefore preferable to espouse the already given *asymmetric* definition of the pairing, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$, not least because in practice the known asymmetric pairings tend to be more compact and efficient.

The paper [GPS08] proposes a finer classification of pairings, that takes into account the ease with which we can move from $\mathbb{G}_1$ to $\mathbb{G}_2$ and back, using a group isomorphism $\phi : \mathbb{G}_2 \to \mathbb{G}_1$ and its inverse ϕ^{-1}. Such isomorphisms of course always exist, but are not necessarily easy to compute. If both are efficiently computable, then the pairing is said to be of type 1, and it is a simple matter to rewrite it as a symmetric pairing with $\mathbb{G}_1 = \mathbb{G}_2$. If only one of them is efficiently computable, taken by convention to be ϕ from $\mathbb{G}_2$ to $\mathbb{G}_1$, then the pairing is said to be of type 2, and though asymmetric as defined in $\mathbb{G}_1 \times \mathbb{G}_2$, it can at some cost be transformed into a symmetric pairing $e' : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$. Lastly, if none of the isomorphisms ϕ and ϕ^{-1} is efficiently computable, then the pairing is said to be of type 3, and is irremediably asymmetric.

Although $\mathbb{G}_1$ and $\mathbb{G}_2$ may be the same, it is crucial that $\mathbb{G}_t$ remain distinct, and in particular that once we have landed in $\mathbb{G}_t$, it be infeasible to come back into $\mathbb{G}_1$ or $\mathbb{G}_2$, because doing so efficiently would violate most of the hardness assumptions that make bilinear pairings useful in cryptography.

2. A High-level Classification of IBE Frameworks

The vast and growing number of identity-based encryption and related schemes that exist today can be traced to one of three known pairing-based approaches to IBE. We characterize them all in this section, using the criteria and terminology from [Boy07], and borrowing the exposition from [Boy08].

Each framework is characterized by a fundamentally unique way to build an IBE trapdoor from a pairing, and relies on different kinds of assumptions and proofs to realize this core functionality. Of course, there exist many more than three pairing-based constructions of IBE, but the ones that we currently know all seem to fall within the confines of three general frameworks. These are:

1. **The Full-Domain-Hash approach**, whose prototypes are the Boneh-Franklin "$\mathcal{BF}$" IBE [BF01] and the Sakai-Ohgishi-Kasahara IB key exchange [SOK00] that precedes it.
2. **The Exponent-Inversion approach**, typified by the *second* Boneh-Boyen "$\mathcal{BB}_2$" IBE [BB04a, §5], the (modified) Sakai-Kasahara "$\mathcal{SK}$" IBE [SK03], and, arguably, also the Gentry IBE [Gen06].
3. **The Commutative-Blinding approach**, defined more or less exclusively by the *first* Boneh-Boyen "$\mathcal{BB}_1$" IBE [BB04a, §4].

Although the latter approach and its many uses are the subject of this chapter, first, it is useful to briefly describe them all with their similarities and differences.

2.1. Full-Domain-Hash IBE

The earliest and, perhaps, conceptually simplest approach to IBE from pairings was discovered independently by [SOK00] and by [BF01]. Even though the scheme of [SOK00] was an ID-based key exchange, and that of [BF01] an IBE proper, both teams came up with essentially the same ID-based key extraction technique.

The idea is fairly straightforward: it consists of using a cryptographic hash to map an identity string $\mathsf{ID} \in \{0,1\}^*$ to a bilinear group element $H(\mathsf{ID}) \in \mathbb{G}_2$. With the help of a special public value $\alpha P \in \mathbb{G}_1$, any hash value $H(\mathsf{ID})$ can serve as an encryption public key for the identity ID. The corresponding ID-based decryption key is $\alpha H(\mathsf{ID})$ and can only be computed by the central authority who knows the master key α. (See Chapter IV in this collection for a description of the Boneh-Franklin system.)

We propose the name "full-domain hash" for this approach because it crucially relies on hashing the identity directly into a bilinear group, using a hash function which must be modeled as a random oracle. Several schemes fall under this denomination, being direct extensions of the Boneh-Franklin systems; *e.g.*, we mention the hierarchical constructions of [GS02] and [YFDL04]. A distinguishing feature of all full-domain-hash IBE systems is that they construct an internal session key of the form $e(P, H(\mathsf{ID}))^{\alpha\, r}$, where α is the master secret and r is an ephemeral encryption randomizer chosen by the sender.

One drawback of this approach is that it makes extensive use of cryptographic hashes (modeled as random oracles), and in particular assumes the availability of hash functions with uniformly distributed images in a bilinear group (*i.e.*, on an elliptic curve). Two problems with this kind of hashing are that it is somewhat expensive, and more importantly that it could potentially restrict the choice of curves since it is not always possible to sample uniformly from or hash evenly into the proper subgroup, without involving the knowledge of any discrete logarithm. Fortunately, the Boneh-Franklin method has been successfully adapted to work with all known types of pairing-friendly curves (sometimes by replacing individual group elements by collective cosets); however, the hashing requirement remains a limitation of the full-domain-hash paradigm that could resurface in the future.

Another drawback of the full-domain-hash framework is that it leads to IBE systems that are significantly less efficient than the other, newer approaches. This inefficiency is due in part to the fact that hashing directly on a curve is expensive (originally this required a symmetric pairing, though asymmetric pairings can now be used), and in part because encryption always requires a costly pairing computation that cannot be avoided.

2.2. Exponent-Inversion IBE

The second approach we describe has its roots in a new type of assumption originally proposed by [MSK02] in the context of a traitor tracing scheme. Their intuition was that any coefficient that appears as the exponent of a group element should be hard to invert, *i.e.*, it should be hard to compute $1/x P$ given P and xP. However, with the pairing it is easy to "cancel out" the exponent, by pairing xP with $1/x \hat{P}$, without having to reveal x itself.

To realize an IBE system from this idea, one would have to encode the identity as part of x, and devise a way to make xP computable from public information and $1/x \hat{P}$ from a secret trapdoor. For example, one could define a linear function $x(\mathsf{ID}) = \alpha + \beta\,\mathsf{ID}$, and publish $\alpha\hat{P}$ and $\hat{P}^{\beta}$. A benefit of this "exponent inversion" framework is that we no longer need to hash directly into one of the pairing groups; we merely hash into $\mathbb{Z}_p$ which is an easy operation.

The first scheme based on this approach was proposed by [SK03], without security proof, though a proof was later given by [CC05] in the random-oracle model, based on the proof of $\mathcal{BB}_2$. Indeed, the first provably secure IBE scheme based on the exponent-inversion principle is the $\mathcal{BB}_2$ system of [BB04a], which was designed for the specific purpose of achieving security without random oracles ($\mathcal{BB}_2$ is not to be confused with the completely different $\mathcal{BB}_1$ system from the same paper, and our focus here). More recently, [Gen06] proposed yet another variation on the exponent-inversion theme, with a tighter security proof than the earlier proposals.

Unfortunately, proving the security of exponent-inversion IBE is not a simple affair. A common characteristic of all these schemes is that their security proofs require surprisingly strong assumptions. A typical assumption for these schemes is q-BDHI, which states that it is hard to compute $e(P, \hat{P})^{1/x}$ (and thus $1/x P$ and $1/x \hat{P}$) given a polynomially long sequence of elements: P, xP, $x^2 P$, $x^3 P$, up to $x^q P$. (Gentry's scheme uses an even stronger assumption which requires specific elements to be expressly omitted from the sequence.) The length of the sequence is determined by a parameter q, which for the known IBE constructions must be at least as large as the maximum number of private key owners that an adversary may corrupt in an active attack (*i.e.*, the number of key extraction made by the adversary in the security game). Hence, the value of q must be large for the proofs to have bearing on practice, but then the assumption becomes stronger and less reassuring.

Indeed, a number-theoretic analysis [Che06] shows that these "large-q" assumptions may only provide $O(\sqrt[3]{p})$ concrete security against the generic recovery of x, instead of the larger $O(\sqrt{p})$ concrete security that one would have expected for a generic discrete-log attack. Fortunately, this analysis does not bring into question the credibility of BDHI and similar assumptions, because a matching lower bound $\Omega(\sqrt[3]{p})$ on the generic complexity of breaking BDHI had been previously predicted by [BB04c]. Nevertheless, the results of [Che06] stress the necessity of correcting for the worst-case value of q (which depends on the threat model) when provisioning a system that relies on such "large-q" assumptions.

2.3. Commutative-Blinding IBE

The newest and most appealing IBE framework is that of the $\mathcal{BB}_1$ scheme originally proposed by [BB04a]. Their method sidesteps most or all the problems associated with the earlier approaches: in particular, the $\mathcal{BB}_1$ scheme allows identities to be encoded (or hashed) as integers as in the exponent-inversion approach, but admits a much better security reduction based on the same BDH complexity assumption as in the full-domain-hash approach (hence, no "large-q" assumption).

Very roughly, the IBE framework of [BB04a] is based on the idea of creating, from two or more secret coefficients, two blinding factors that "commute" with each other under the pairing (*i.e.*, the unblinding need not be the reverse of the blinding). The name given to this approach, "commutative blinding", is an attempt to convey the essence of this mechanism.

Perhaps the best quality of the commutative blinding paradigm is the greater flexibility provided by its algebraic structure. Although this approach was the last one to appear, it quickly surpassed the others for the number and variety of extensions to the basic notion of IBE that it has enabled. Here, an "IBE extension" refers to a cryptosystems whose functionality subsumes that of plain IBE at least in some respect. We shall discuss a few examples of extensions in a later section.

2.4. Quadratic-Residuosity and Hard-Lattice IBE (without pairings)

We also merely mention two completely different types of IBE construction. The first one is based on the quadratic residuosity problem, and was originally invented in [Coc01] and later modified in [BGH07b]. The second was recently proposed in [GPV07] and is based on hard lattice problems.

Although they certainly are very creative and mathematically appealing, we do not discuss these approaches further, as both of them suffer from a lack of efficiency and/or compactness compared to the pairing-based frameworks (especially the lattice-based scheme), and also a conspicuous lack of extensions beyond IBE.

3. The Commutative Blinding Framework

As mentioned above, the Commutative-Blinding framework originates with the $\mathcal{BB}_1$ IBE scheme from [BB04a], which only provided a basic identity-based encryption functionality with a security reduction for a rather restricted notion of security (that of security under selective-identity chosen-plaintext attacks). It did, however, manage to do so in the standard model, thanks to a new IBE structure that we now call Commutative Blinding.

3.1. Principal Common Characteristics

The Commutative Blinding framework, as epitomized by the $\mathcal{BB}_1$ IBE scheme from [BB04a], is based on the creation of randomized session keys of the general form $e(P, \hat{P})^{\alpha s}$ where α is the master secret key and s is an ephemeral encryption randomizer. That is, the session key must depend on both the master key α and a randomizer s chosen by the encrypting party. However, it need not depend on the identity ID or a random-oracle hash thereof unlike the Full-Domain-Hash $\mathcal{BF}$ IBE scheme. Also, unlike the Exponent-Inversion schemes discussed in the next chapter, the session key compensates its mandatory dependence on the master key α (which in itself is detrimental to flexibility) with a linear as opposed to fractional dependence on α in all components of the private keys (which is obviously favorable to flexibility).

What makes the Commutative Blinding framework useful for IBE and some variants such as HIBE and ABE (see below) is that it satisfies the following couple of properties:

Commutative Blinding This is the property that the session key $e(P, \hat{P})^{\alpha s}$ for some random s can be computed in several alternative ways, based on disjoint subsets of non-public information (at least between the second and third ways, which are of particular interest for IBE):

1. from the master secret itself α (or $\alpha\hat{P}$) and a bare-bones public ciphertext sP (which in itself is not identity-based); or
2. from a labeled blinded version of the master secret, consisting of elements $\alpha\hat{P} + r f_{\mathbb{G}_2}(\mathsf{ID})$ and $r\hat{P}$ for some random r, and a full-fledged IB matching ciphertext, consisting of elements $s f_{\mathbb{G}_1}(\mathsf{ID})$ and sP; or even
3. from a public key of the form $e(P, \alpha P)$ and the encryption randomizer s.

Full Linearity This is the property that the exponents are all linear in all the non-public scalars. That is, the exponents that appear in the private keys, the ciphertexts, and the session keys, all depend linearly on each of the following variables considered in isolation:

- the master secret α,
- any ephemeral such as r and s, and
- any implicit exponent hidden in the function $f(\cdot) : \mathbb{Z}_p \to \mathbb{G}_1$.

Commutative Blinding by itself is almost sufficient to get identity-based encryption: it is also required that from legitimate blinded keys $\alpha\hat{P} + r_i f_{\mathbb{G}_2}(\mathsf{ID}_i)$ and $r_i\hat{P}$ for some set of labels ID_i one cannot construct another key (or implicit decryption apparatus) for a label ID' not in the original set.

Full Linearity is very helpful not so much for basic IBE per se, but for the many extensions that have been proposed over the years.

3.2. Versatile Cryptographic Applications

A direct consequence of the previous properties is that the "labeled" private keys and ciphertexts can be further decomposed into vectors of the form (on the private-key side) $\vec{\alpha}\vec{\hat{P}} + \vec{r}\vec{f}_{\mathbb{G}_2}(\vec{\mathsf{ID}})$ and $\vec{r}\vec{\hat{P}}$ and (on the ciphertext side) $\vec{s}\vec{f}_{\mathbb{G}_1}(\vec{\mathsf{ID}})$ and $\vec{s}\vec{P}$ (thanks to Full Linearity), while preserving the general structure of the $\mathcal{BB}_1$ IBE scheme with the security of its identity-based key derivation (thanks to Commutative Blinding).

This allows the use of various combinations of secret-sharing techniques for the exponents r, s, and/or α, as well as clever constructions for the function f, thanks to which the commutative-blinding framework lends itself quite nicely to a large number of generalizations of IBE, often with the same kind of security reductions and computational assumptions as the original.

There are many successful applications of Commutative Blinding that rely in one way or another on the above principles (even though this may not have necessarily been clear at the time of their inception). For example, we mention:

Hierarchical IBE Originally proposed in [GS02], this notion allows user private keys at the n-th level to serve as master keys for the issuance of private keys at the $n+1$-th level, where the real master authority occupies the 0-th level. Although the first known instance of HIBE [GS02] is an extension of the Full-Domain-Hash $\mathcal{BF}$ IBE and predates the Commutative Blinding framework, the most efficient and flexible HIBE to date [BBG05] does indeed rely quite heavily on the latter. (It is however not on the usual Decision-BDH assumption from [BF01] but the Decision-BDHI assumption from [BB04a].)

Multi-Hierarchy IBE A two-dimensional generalization of HIBE, first proposed in [YFDL04] to combine hierarchies with forward security, illustrates perhaps even more convincingly the added flexibility of Commutative Blinding over Full-Domain Hash. The latter was indeed stretched to its limits to get a working but somewhat inefficient two-dimensional hierarchy in [YFDL04]. By contrast, the Commutative-Blinding approach pursued in [BBG05] generalizes with ease to multi-dimensional hierarchies, and is much more efficient in the two-dimensional case in particular.

Wildcard IBE Another generalization closely related to HIBE is the notion of IBE with Wildcards, or WIBE [ACD$^+$06], where the delegation to subordinate users can happen not merely in a hierarchical fashion, but more generally by describing identities as a set of attributes, some of which may be left blank at first, and later assigned particular values, as one makes the transition from general to specific. The construction due to [ACD$^+$06] quite clearly exploits the Commutative Blinding framework.

Anonymous IBE The case of anonymous IBE is interesting, since it is one where Commutative Blinding and $\mathcal{BB}_1$ IBE at first appears to be at a big disadvantage over Full-Domain Hash and $\mathcal{BF}$ IBE. The latter is indeed naturally anonymous due to the way it uses random oracles to hash the identity in the ciphertext, whereas in the former the identity can be tested from the ciphertext. (Anonymity here is the desire that a ciphertext should not reveal the identity of its intended recipient.) However, the Full Linearity properties allows us, in the $\mathcal{BB}_1$ scheme for instance, to split the identity function $f(\mathsf{ID})$ into two or more randomized components, using secret-sharing techniques, and conceal the original value until it is reconstituted through a product of pairings, which will only give the final decryption without explicating the identity encoding in the ciphertext. This approach was used in [BW06] to construct an Anonymous IBE in the Commutative Blinding framework (based not on Decision-BDH but on the Decision-Linear assumption from [BBS04]).

Anonymous Hierarchical IBE As a reversal of situation, it turns out that Commutative Blinding is, based on current knowledge, better suited to anonymity than any of the other frameworks. The reason is that, regardless of the complexity of the construction, as long as the linearity property applies, one should be able to use the same kind of linear secret-sharing technique to hide all the attributes of the identity, whichever form it may take. An efficient (though somewhat involved) construction of an Anonymous HIBE that guarantees anonymity at every level of the hierarchy is given in [BW06], based on the Decision-Linear assumption. This is something that is currently not known to be possible in any other of the IBE frameworks.

Adaptive-ID "Fully Secure" IBE Two technical but theoretically important instances of the Commutative Blinding family are the "fully secure" IBE schemes presented, respectively, by Boneh and Boyen in CRYPTO 2004 [BB04b] and by Waters at EUROCRYPT 2005 [Wat05]. Both schemes rely on internally decomposing the identity ID bit-by-bit to enable clever security reductions against "adaptive-identity" attacks, without requiring stronger assumptions than Decision-BDH.

The main technical difference between [BB04b] and [Wat05] is that in the former the bit-by-bit decomposition is done externally based on multiple independent instances of $\mathcal{BB}_1$, whereas in the latter the bit-by-bit decomposition is done internally by tweaking the function $f(\mathsf{ID})$ in a single instance of $\mathcal{BB}_1$. This makes the latter system the most efficient of the two. Both schemes [BB04b,Wat05] rely heavily on the properties of the Commutative-Blinding framework to obtain a security reduction in the case of adaptive-identity attacks. This contrasts with the earlier $\mathcal{BF}$ scheme [BF01] whose security was predicated on the random-oracle model.

We take this opportunity to stress, however, that one should not be too quick to dismiss the original $\mathcal{BB}_1$ in the context of "full-ID security".

The fact that the "full $\mathcal{BB}_1$" can be proven secure in the full model via a reduction to "basic $\mathcal{BB}_1$" with an exponential loss factor says nothing about its ability to achieve arbitrarily strong levels of security in an time- and space- efficient manner. (Fundamentally, the exponential losses in the reduction can be counterbalanced by exponential gains that result from selecting larger group orders, which, in this case, increases the computational cost only by a constant factor: see Section 4.1.) With proper parameterization, the original $\mathcal{BB}_1$ system does in fact achieve the same full notion of IBE security as [BB04b,Wat05], and in a more compact and very efficient manner.

We shall describe the full version of $\mathcal{BB}_1$, among others, in Section 4.3, and discuss the Waters construction in Section 5.3.

Fuzzy IBE Yet another use of the Commutative Blinding framework was proposed in [SW05] with the notion of Fuzzy IBE, where identities are described using a number of binary attributes, only a certain percentage of which are required to match between a private key and a ciphertext in order to allow decryption. The idea is based on t-out-of-n threshold secret sharing within the function $f(\mathsf{ID})$, again relying on the Full Linearity property for the sharing and on the Commutative Blinding property for the security reduction.

Attribute IBE or ABE Identity-based encryption with attributes for identifiers, or attribute-based encryption (ABE) for short, is a generalization of the previous notion, where the attributes need not be aggregated in a single t-out-of-n threshold gate as in Fuzzy IBE, but can be aggregated using more complex structures involving trees or circuits with multiple gates. A number of successive refinements to this notion of ABE involving increasingly general circuits and gates have been proposed in recent years; see the chapter on Attribute-Based Encryption (Chapter X) for a summary. All of them are based on the Commutative Blinding framework, although sometimes with (much) stronger complexity assumption than most other extensions of $\mathcal{BB}_1$.

3.3. Extensions in Other Frameworks

Historically, the first published IBE was the $\mathcal{BF}$ system, in what we now characterize as the Full-Domain Hash approach; and so it is not surprising that it also led to the first working instances of HIBE and Forward-Secure HIBE, as already mentioned.

Much more recently, it was shown in [Boy07] that Fuzzy IBE and ABE are not beholden to the Commutative-Blinding framework, and can instead be realized in the Exponent-Inversion framework using $\mathcal{BB}_2$ or $\mathcal{SK}$ IBE as black-boxes, without too much loss of efficiency. We refer to the chapter on Exponent Inversion (Chapter VI) for details. At the time of this writing, however, it appears that the Commutative-Blinding implementations of ABE remain the most efficient.

4. The Archetype of Commutative Blinding

For concreteness, and to illustrate how the commutative-blinding principles come into play, we describe the $\mathcal{BB}_1$ IBE system, which captures the essence of this approach.

The original $\mathcal{BB}_1$ scheme from [BB04a] came with a security reduction in the standard model against selective-identity, chosen-plaintext attacks (IND-sID-CPA). For practical applications, however, the requirement is often for “full” adaptive-identity, chosen-ciphertext (IND-ID-CCA) security guarantees. To this end, we show how to augment or modify $\mathcal{BB}_1$ to feature this strongest notion of security. We show how to do this in two possible ways:

- Reasonably efficiently, in the standard model (*directly* in the basic $\mathcal{BB}_1$ scheme, *without* decomposing the identities into bits [Wat05] or words [Nac05] for adaptive-identity security);
- Very efficiently, in the random-oracle model (once again,directly in the $\mathcal{BB}_1$ scheme, without using generic hybrid transforms [FO99] for chosen-ciphertext security).

Since adaptive-identity and chosen-ciphertext security are orthogonal properties, we can deal with each of them independently of the other. Adaptive-identity security is achieved through proper design of the private-key generation algorithm $\mathsf{Extract}$, whilst chosen-ciphertext security mostly depends on the encryption and decryption algorithms proper $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$ (or $\mathsf{Encapsulate}$ and $\mathsf{Decapsulate}$ in the case of a KEM).

For maximum generality, we shall decouple the descriptions of key extraction from that of encryption/decryption. For each, we give (at least) a basic version, a fully secure random-oracle version, and a fully secure standard-model version. This will give us a total of nine combinations.

4.1. Security Parameters

Let $\ell \in \mathbb{N}$ be a security parameter. For simplicity, we shall assume that both the identities and the messages (or session keys) are bit strings in $\{0,1\}^\ell$. As usual, we have a pair of bilinear groups $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively generated by $P \in \mathbb{G}_1$ and $\hat{P} \in \mathbb{G}_2$, and of prime order $p = |\mathbb{G}_1| = |\mathbb{G}_2|$. Let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$ be the pairing, where $\mathbb{G}_t$ is the target group generated by $e(P, \hat{P})$ also of order $|\mathbb{G}_t| = p$.

The size of the prime p will depend on the version of $\mathcal{BB}_1$ we consider: if we assume that the fastest discrete-log algorithms in $\mathbb{G}_1$ and $\mathbb{G}_2$ are generic, then it is generally sufficient to require $p > 2^{2\ell}$. There is one exception: for the version of $\mathcal{BB}_1$ with adaptive identity in the standard model [BB04a, §7], we shall need $p > 2^{4\ell}$ in order to meet the level of security prescribed by the security parameter ℓ.

For all known pairing implementations, which are based on the Weil or the Tate pairing, $\mathbb{G}_t$ is the multiplicative group μ_p of p-th roots of 1 in a finite field $\mathbb{F}_{p'}$, and so it is necessary that $|\mathbb{F}| = p' > \Omega(p^{3/2})$ to prevent the number-field sieve in $\mathbb{F}$, which has complexity $\Theta(L_{p'}(1/3))$, from becoming a faster algorithm for the discrete logarithm in $\mathbb{G}_t$ than the generic algorithms, which in groups of prime order p such as $\mathbb{G}_t$ have complexity $\Theta(L_p(1/2)) = \tilde{\Theta}(\sqrt{p})$.

We emphasize that since all pairing and exponentiation algorithms run in total cubic-log time $\Theta(\lceil \log_2 p \rceil^2 \lceil p' | \rceil)$, switching from a requirement that $p > 2^{2\ell}$ to a requirement that $p > 2^{4\ell}$ only makes the computations slower by a constant factor — it is asymptotically the same.

In practical terms, the user's normal crypto*graphic* operations will be slowed down by a constant factor $2^5 = 32$, independently of the security parameter ℓ. In contrast, the adversary's crypt*analytic* operations will become exponentially slower, to the tune of $\Theta(p^{2\ell/3})$ based on the complexity of the number-field sieve. This exponential gap is what lets us turn selective-ID $\mathcal{BB}_1$ into adaptive-ID $\mathcal{BB}_1$ at the same security level, efficiently. (See [BB04a, §7] for the general theorem.)

4.2. System Setup

In the description that follows, we consider four placeholders, H_1, H_2, H_3, H_4, for "hash" functions with tunable properties — they are always assumed to be collision-resistant though not necessarily one-way:

- $H_1 : \{0,1\}^\ell \to \mathbb{Z}_p$ mapping recipient-identity strings into unique integers;
- $H_2 : \mathbb{G}_t \to \{0,1\}^\ell$ morphing target-group elements into binary session-keys;
- $H_3 : \mathbb{G}_t \times \{0,1\}^\ell \times \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{Z}_p$ making digests out of $\mathcal{BB}_1$ ciphertexts;
- $H_4 : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{Z}_p$ mapping partial ciphertexts to sub-identities (see below).

These functions will be either modeled as random oracles, or merely assumed to be collision-resistant (or merely injective) without any one-wayness requirement. H_3 is optional and always a random oracle when needed. H_4 is also optional and always merely collision-resistant when needed.

The $\mathcal{BB}_1$ setup is the same regardless of our choice of application algorithms. For concentrated (non-distributed) key extraction authorities, the most efficient approach is to remember all scalar exponent, as follows:

BB$_1$.Setup To generate IBE system parameters, pick $\omega, \alpha, \beta, \gamma \in \mathbb{Z}_p$, and output,

$$\textit{params} = \Big(P,\ P_1 = \alpha P,\ P_2 = \beta P,\ P_3 = \gamma P,\ v_0 = e(P, \hat{P})^\omega\Big) \in {\mathbb{G}_1}^4 \times \mathbb{G}_t\,,$$

$$\textit{masterk} = (\hat{P}, \omega, \alpha, \beta, \gamma) \in \mathbb{G}_2 \times {\mathbb{Z}_p}^4\ .$$

The element P_3 and the corresponding secret γ are only needed for the KEM with "full CCA security in the standard model" (see below); γ and P_3 may otherwise be omitted.

4.3. Private-Key Extraction

We give three modular versions of $\mathcal{BB}_1$ private-key extraction $\mathsf{Extract}$:

1. the basic version providing only selective-ID security,
2. a version with efficient adaptive-ID security in the random-oracle model,
3. a version with less efficient adaptive-ID security in the standard model.

Each of them can be used indifferently with the various CPA- and CCA-secure encryption/decryption algorithms described later.

The three versions of $\mathsf{Extract}$ are in fact the same, except for the choice of function $H_1 : \{0,1\}^\ell \to \mathbb{Z}_p$ mapping identities to integers, and which indirectly affects the size of p for a given choice of ℓ. The extraction algorithm is as follow.

BB$_1$.Extract To extract from *masterk* a private key d_{ID} for an identity $\mathsf{ID} \in \{0,1\}^\ell$, pick a random $r \in \mathbb{Z}_p$ and output,

$$d_{\mathsf{ID}} = \Big(d_0 = (\omega + (\alpha H_1(\mathsf{ID}) + \beta) r)\hat{P},\ d_1 = r\hat{P}\Big) \in \mathbb{G}_2 \times \mathbb{G}_2\ .$$

For the "full CCA security in the standard model" KEM version (see below), output instead,

$$d_{\mathsf{ID}} = \begin{pmatrix} d_0 = (\omega + (\alpha H_1(\mathsf{ID}) + \beta) r)\hat{P},\ d_1 = r\hat{P},\ d_2 = r\hat{P}_3, \\ \hat{P},\ \hat{P}_1 = \alpha\hat{P},\ \hat{P}_3 = \gamma\hat{P} \end{pmatrix} \in {\mathbb{G}_2}^6\ .$$

Either way, the function $H_1 : \{0,1\}^\ell \to \mathbb{Z}_p$ depends on the desired level of security against chosen identity attacks. The three options are given next.

4.3.1. Selective sID security in the standard model

Requires $H_1 : \{0,1\}^\ell \to \mathbb{Z}_p$ an arbitrary collision-resistant or injective map.

4.3.2. Adaptive ID security in the random-oracle model

Requires a cryptographic hash $H_1 : \{0,1\}^\ell \to \mathbb{Z}_p$ viewed as a random oracle.

4.3.3. Adaptive ID security in the standard model

Requires an arbitrary injective map $H_1 : \{0,1\}^\ell \to \mathbb{Z}_p$ but with $\log_2 \lceil p \rceil \geq 4\ell$.

As already discussed in Section 4.1, $\mathcal{BB}_1$ with "full" adaptive-ID security in the standard model will be about 32 times slower than either the basic version with selective-ID security or the full version in the random-oracle model. The penalty ratio of 32 holds for any concrete target security level ℓ, regardless of ℓ (and, thus, regardless of the bit-size of the identities, which grows with ℓ).

Observe also that one can "hedge" the security of $\mathcal{BB}_1$ and get the best of all worlds, if for H_1 we use a cryptographic map that may or may not be adequately modeled as a random oracle, but that is guaranteed to be collision-resistant (*e.g.*, by building H_1 as a wrapper over the AES cipher). An injective H_1 will still be compatible with the "random-oracle" argument.

4.4. Encryption and Decryption

We now describe four versions of the $\mathcal{BB}_1$ encryption and decryption algorithms, with various levels of security and efficiency. The first one appeared explicitly in [BB04a], the second was mentioned in [Boy08], and the third is implied by [BB04a] and [BMW05].

In all cases, the function $H_1 : \{0,1\}^\ell \to \mathbb{Z}_p$ is left unspecified, as its choice will depend on the selected variant of **Extract**, as explained above.

4.4.1. CPA security in the standard model (regular encryption version)

BB$_1$.Encrypt Using for $H_2 : \mathbb{G}_t \to \{0,1\}^\ell$ a fixed arbitrary collision-resistant map, output,

$$C = \begin{pmatrix} c = M \oplus H_2(v_0^s), \\ c_0 = sP, \\ c_1 = H_1(\mathsf{ID})\, sP_1 + sP_2 \end{pmatrix} \in \{0,1\}^\ell \times \mathbb{G}_1 \times \mathbb{G}_1 \,,$$

where $M \in \{0,1\}^\ell$ is the message, $\mathsf{ID} \in \{0,1\}^\ell$ is the recipient identifier, and $s \in \mathbb{Z}_p$ is a random ephemeral integer.

BB$_1$.Decrypt Given a ciphertext C and a private key $d_{\mathsf{ID}} = (d_0, d_1)$, output,

$$M = c \oplus H_2\left(\frac{e(c_0, d_0)}{e(c_1, d_1)}\right) \in \{0,1\}^\ell \ .$$

4.4.2. CCA security in the random-oracle model (regular encryption version)

BB$_1$.Encrypt Using for $H_2 : \mathbb{G}_t \to \{0,1\}^\ell$ and $H_3 : \mathbb{G}_t \times \{0,1\}^\ell \times \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{Z}_p$ two "random-oracle" cryptographic hash functions fixed at setup, output,

$$C = \begin{pmatrix} c = M \oplus H_2(k = v_0^s), \\ c_0 = sP, \\ c_1 = H_1(\mathsf{ID})\, sP_1 + sP_2, \\ t = s + H_3(k, c, c_0, c_1) \bmod p \end{pmatrix} \in \{0,1\}^\ell \times \mathbb{G}_1 \times \mathbb{G}_1 \times \mathbb{Z}_p \,,$$

where $M \in \{0,1\}^\ell$ is the message, $\mathsf{ID} \in \{0,1\}^\ell$ is the recipient identifier, and $s \in \mathbb{Z}_p$ is a random ephemeral integer.

BB$_1$.Decrypt Given a ciphertext C and a private key $d_{\mathsf{ID}} = (d_0, d_1)$, compute,

$$k = \frac{e(c_0, d_0)}{e(c_1, d_1)} \in \mathbb{G}_t \,, \qquad s = t - H_3(k, c, c_0, c_1) \in \mathbb{Z}_p \ .$$

If $(k, c_0) \neq \big(v_0^s, sP\big)$, output $\perp$; otherwise, output, $M = c \oplus H_2(k)$.

4.4.3. CCA security in the standard model (key encapsulation “KEM” version)

BB$_1$.Encapsulate Using for $H_2 : \mathbb{G}_t \to \{0,1\}^\ell$ and for $H_4 : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{Z}_p$ two arbitrary collision-resistant maps, both fixed at setup, output,

$$K = H_2\left(v_0^s\right) \in \{0,1\}^\ell, \qquad C = \begin{pmatrix} c_0 = sP, \\ c_1 = H_1(\mathsf{ID})\, sP_1 + sP_2, \\ c_2 = H_4(c_0, c_1)\, sP_1 + sP_3 \end{pmatrix} \in {\mathbb{G}_1}^3,$$

where $\mathsf{ID} \in \{0,1\}^\ell$ is the recipient identifier, $s \in \mathbb{Z}_p$ is a random ephemeral, K the induced session key, and C its encapsulation.

BB$_1$.Decapsulate Given a ciphertext C and a private key $d_{\mathsf{ID}} = (d_0, d_1, d_2)$, output,

$$K = H_2\left(\frac{e(c_0,\ d_0 + r\hat{P}_3 + H_4(c_0,c_1)r\hat{P}_1)}{e(c_1,d_1)\cdot e(c_2, r\hat{P})}\right) \in \{0,1\}^\ell,$$

where $r \in \mathbb{Z}_p$ is an ephemeral randomizer chosen by the recipient.
Observe that for matching identifiers in C and d_{ID} the decryption randomness r vanishes and the output K becomes deterministic.
Barring that, K is random and statistically independent of C and d_{ID}.

4.5. Security of the Modular $\mathcal{BB}_1$ IBE

The following theorem summarizes the security properties of all the possible combinations of the $\mathcal{BB}_1$ functions we just described.

Theorem V.1 (from [BB04a]) *The previously described modular $\mathcal{BB}_1$ scheme realizes the $\{$IBE$|$IBKEM$\}$ functionality, securely in the IND-$\{$sID$|$ID$\}$-$\{$CPA$|$CCA2$\}$ sense, at the concrete $(\ell - \log_2 q)$-bit level, for a total of q adversarial queries, provided that the bilinear groups $(\mathbb{G}_1, \mathbb{G}_2)$ of prime order p uphold the $\{$decisional$|$computational$\}$ BDH assumption, for $\{p \geq 2^{2\ell} | p \geq 2^{4\ell}\}$ as specified.*

More informally, this means that, if, on expectation, solving the BDH problem in $(\mathbb{G}, \mathbb{G}_2)$ has a cost of $\sqrt{p}$, then distinguishing a challenge $\mathcal{BB}_1$ ciphertext from random will cost of $2^\ell/q$, in the attack model that corresponds to the chosen versions of the $\mathcal{BB}_1$ algorithms, where q is the total number of queries made by the adversary, including key extraction queries, decryption queries (if allowed), and random-oracle queries (if applicable).

5. Beyond Identity-Based Encryption

The already numerous possible variants of the $\mathcal{BB}_1$ scheme from the previous section, are nothing next to the wild generalizations of IBE that have been proposed in the past four years, exploiting the core properties of Commutative Blinding.

5.1. Distributed Authorities

We start by describing a very simple extension of $\mathcal{BB}_1$, that enables distributed authorities and verifiable secret-sharing applications. For VSS in particular, it is possible to use as master key not the actual exponents $\omega, \alpha, \ldots$, but only the group elements $\hat{P}^\omega, \hat{P}^\alpha, \ldots$, from which the exponents themselves are presumably irretrievable.

The extraction algorithm will need to be adapted slightly, but since it extracts private keys with the same distribution as before, the encryption and decryption algorithms can remain unchanged. The modified setup and key extraction algorithms are as follows.

BB$_1$.DistSetup Pick random $\omega, \alpha, \beta, \gamma \in \mathbb{Z}_p$, and output,

$$\mathit{params} = \Big(P,\ P_1 = \alpha P,\ P_2 = \beta P,\ P_3 = \gamma P,\ v_0 = e(P, \hat{P})^\omega\Big) \in {\mathbb{G}_1}^4 \times \mathbb{G}_t\,,$$
$$\mathit{masterk}' = \Big(\hat{P},\ \hat{P}_1 = \alpha \hat{P},\ \hat{P}_2 = \beta \hat{P},\ \hat{P}_3 = \gamma \hat{P},\ \hat{P}_0 = \omega \hat{P}\Big) \in {\mathbb{G}_2}^5 \ .$$

The real master key is just $\hat{Q}_0$. The other elements need not be kept secret, but need not be published either (except for the "full CCA security in the standard model" KEM; see below).

BB$_1$.DistExtract Pick a random $r \in \mathbb{Z}_p$ and output,

$$d_{\mathsf{ID}} = \Big(d_0 = \hat{P}_0 + H_1(\mathsf{ID})\, r\hat{P}_1 + r\hat{P}_2,\ d_1 = r\hat{P}\Big) \in \mathbb{G}_2 \times \mathbb{G}_2\,,$$

or, for use with the "full CCA security in the standard model" KEM,

$$d_{\mathsf{ID}} = \begin{pmatrix} d_0 = \hat{P}_0 + H_1(\mathsf{ID})\, r\hat{P}_1 + r\hat{P}_2,\ d_1 = r\hat{P},\ d_2 = r\hat{P}_3, \\ \hat{P},\ \hat{P}_1,\ \hat{P}_3 \end{pmatrix} \in {\mathbb{G}_2}^6 \ .$$

As before, the choice of function $H_1 : \{0,1\}^\ell \to \mathbb{Z}_p$ will depend on the desired protection against chosen identity attacks. The three options are the same as before.

This variant is beneficial for applications based on Verifiable Secret Sharing (VSS), such as the cryptosystem of [BBH06] which provides chosen-ciphertext security with non-interactive threshold decryption. Key extraction in the VSS variant of $\mathcal{BB}_1$ requires three fixed-base exponentiations in $\mathbb{G}_2$ instead of two.

5.2. Hierarchical Identities

Another useful extension of $\mathcal{BB}_1$, which was in fact proposed at the same time as the basic scheme [BB04a], is to support hierarchical identities.

The basic $\mathcal{BB}_1$ HIBE scheme (for CPA-secure encryption) is as follows:

Setup To generate HIBE system parameters for a hierarchy with L levels, pick $\omega, \alpha, \beta_1, \ldots, \beta_L \in \mathbb{Z}_p$ at random, and output,

$$\mathit{params} = \begin{pmatrix} P,\ P_1 = \alpha P,\ Q_1 = \beta_1 P,\ \dots,\ Q_L = \beta_L P \\ \hat{P},\ \hat{P}_1 = \alpha \hat{P},\ \hat{Q}_1 = \beta_1 \hat{P},\ \dots,\ \hat{Q}_L = \beta_L \hat{P} \\ v_0 = e(P, \hat{P})^{\omega} \end{pmatrix} \in {\mathbb{G}_1}^{2+L} \times {\mathbb{G}_2}^{2+L} \times \mathbb{G}_t ,$$

$$\mathit{masterk} = (\hat{P}_0 = \omega \hat{P}) \ \in \mathbb{G}_2 \ .$$

Extract To extract from *masterk* a private key for a level-j hierarchical identity $\mathsf{ID} = (\mathrm{I}_1, \dots, \mathrm{I}_j) \in (\{0,1\}^{\ell})^j$, pick random $r_1, \dots, r_j \in \mathbb{Z}_p$ and output,

$$d_{\mathsf{ID}} = \left(\hat{P}_0 + \sum_{i=1}^{j} r_i H_i(\mathrm{I}_i) \hat{P}_1 + r_i \hat{Q}_i,\ \ r_1 \hat{P},\ \ \dots,\ \ r_j \hat{P} \right) \in {\mathbb{G}_2}^{1+j} \ .$$

Here, the maps $H_i : \{0,1\}^{\ell} \to \mathbb{Z}_p^{\times}$ for $1 \le i \le L$ are fixed at setup, as, either,

- arbitrary injective maps, for basic "sHID" hierarchical selective-ID security in the standard model, or
- independent cryptographic hash functions, if we seek "HID" hierarchical adaptive-ID security in the random-oracle model.

Derive To derive a private key d_{ID} for $\mathsf{ID} = (\mathrm{I}_1, \dots, \mathrm{I}_j)$ given a "parent" private key $d_{\mathsf{ID}|j-1} = (d_0, \dots, d_{j-1}) \in {\mathbb{G}_2}^{j}$ for the identity $\mathsf{ID}_{|j-1} = (\mathrm{I}_1, \dots, \mathrm{I}_{j-1})$, pick random $r_1, \dots, r_j \in \mathbb{Z}_p$ and output,

$$d_{\mathsf{ID}} = \begin{pmatrix} (d_0 + r_j H_j(\mathrm{I}_j) \hat{P}_1 + r_j \hat{Q}_j) + \sum_{i=1}^{j} r_i H_i(\mathrm{I}_i) \hat{P}_1 + r_i \hat{Q}_i, \\ (d_1 + r_1 \hat{P}),\ \dots,\ (d_{j-1} + r_{j-1} \hat{P}),\ r_j \hat{P} \end{pmatrix} \in {\mathbb{G}_2}^{1+j} \ .$$

The maps $H_i : \{0,1\}^{\ell} \to \mathbb{Z}_p^{\times}$ for $1 \le i \le L$ are as above.

Encrypt To encrypt a given message $M \in \{0,1\}^{\ell}$ under a hierarchical identity $\mathsf{ID} = (\mathrm{I}_1, \dots, \mathrm{I}_j) \in (\{0,1\}^{\ell})^j$, pick a random $s \in \mathbb{Z}_p$ and output,

$$C = \Big(M \oplus H(v^s),\ \ sP,\ \ s\, H_1(\mathrm{I}_1) P_1 + sQ_1,\ \ \dots,\ \ s\, H_j(\mathrm{I}_j) P_1 + sQ_j \Big) \in \{0,1\}^{\ell} \times {\mathbb{G}_1}^{1+j} \ .$$

Here, $H : \mathbb{G}_t \to \{0,1\}^{\ell}$ is a collision-resistant map fixed at setup.

Decrypt To decrypt a ciphertext $C = (c, c_0, c_1, \dots, c_j) \in \{0,1\}^{\ell} \times {\mathbb{G}_1}^{1+j}$ using the private key $d_{\mathsf{ID}} = (d_0, d_1, \dots, d_j) \in {\mathbb{G}_2}^{1+j}$, output,

$$M = c \oplus H \left(\prod_{i=1}^{j} e(c_i, d_i) \Big/ e(c_0, d_0) \right) \in \{0,1\}^{\ell} \ .$$

5.3. Adaptive-ID Security

A nice technical application of the Commutative Blinding framework is the Waters IBE scheme [Wat05], which by a slight modification of $\mathcal{BB}_1$ and a more involved security proof, manages to achieve security against adaptive-identity adversaries in smaller bilinear groups than the Full-$\mathcal{BB}_1$ given in Section 4.3.

The main practical difference between the adaptive-ID $\mathcal{BB}_1$ from Section 4.3 and Waters' IBE and is that the former requires fewer elements in bigger groups to achieve the same levels of "full ID" security. For IND-ID-CPA security,

- Full-$\mathcal{BB}_1$ IBE ciphertexts and public keys respectively consist of 2 and 4 group elements of 4ℓ bits each;
- Waters IBE ciphertexts and public keys respectively consist of 2 and $\ell + 3$ group elements of 2ℓ bits each.

At a very high level, the Waters proof of security (and the much more complex "secure IBE without random oracles" by Boneh and Boyen that preceded it [BB04b]) relies on the same principle as the simple Full-$\mathcal{BB}_1$ given in Section 4.3: a restriction (of sorts) of the effective space of identities.

The key difference, however, is that in the Waters scheme the restriction is not absolute as in Full-$\mathcal{BB}_1$, but gradual, and based on a bitwise decomposition of the identities. The greater the number of bits, the less secure the Waters scheme becomes — with a slope that starts lower than, but that is not as steep as, $\mathcal{BB}_1$.

Next follows a brief description of the Waters scheme. We only need to give the setup and key extraction functions; the encryption and decryption operations are essentially the same as $\mathcal{BB}_1$.

Setup Pick $\omega, \alpha_1, \ldots, \alpha_\ell, \beta \in \mathbb{Z}_p$, and output,

$$\mathit{params} = \begin{pmatrix} P,\ P_1 = \alpha_1 P,\ P_2 = \alpha_2 P,\ \ldots,\ P_\ell = \alpha_\ell P, \\ P_o = \beta P,\ v_0 = e(P, \hat{P})^\omega \end{pmatrix} \in {\mathbb{G}_1}^{2+\ell} \times \mathbb{G}_t\,,$$

$$\mathit{masterk} = (\hat{P}, \omega, \alpha_1, \ldots, \alpha_\ell, \beta) \ \in \mathbb{G}_2 \times {\mathbb{Z}_p}^{2+\ell} \quad .$$

Extract To extract a private key d_{ID} for an identity $\mathsf{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_\ell) \in \{0,1\}^\ell$, pick a random $r \in \mathbb{Z}_p$ and output,

$$d_{\mathsf{ID}} = \Big((\omega + (\sum_{i=1}^{\ell} \alpha_i \mathrm{I}_i + \beta) r) \hat{P},\ r\hat{P} \Big) \ \in \mathbb{G}_2 \times \mathbb{G}_2 \quad .$$

5.4. Anonymity

Ciphertext anonymity for IBE is analogous to the usual notion of message privacy, except that it bears on the addressee rather than the content. It is the property of recipient indistinguishability by anyone other than the recipient himself and the key-issuing authority.

Far from being an exclusivity of the Commutative Blinding framework, ciphertext anonymity was already satisfied in the very first $\mathcal{BF}$ IBE scheme of the Full-Domain

Hash variety [BF01]. (The anonymity property itself, however, was only defined in [Boy03].) Moreover, the $\mathcal{BB}_1$ IBE and most schemes based on the Commutative Blinding principles, in fact, do not seem to satisfy this property naturally. Could it be that the Full-Domain Hash approach were indeed superior for purposes of anonymity?

The answer, as already mentioned in Section 3, is that, even if anonymity does not come *automatically* in Commutative Blinding schemes as it did in $\mathcal{BF}$ IBE, it is quite easy to retrofit after the fact, using a simple linear decomposition in the exponent. Furthermore, since it is engineered rather than fortuitous, it can be replicated in larger structures, such as in a hierarchy at all levels.

Without going into such complex structures, it is useful to show how to make $\mathcal{BB}_1$ anonymous, by exploiting the structure of Commutative Blinding. The idea is to split one of the two "redundant" components c_0 and c_1 of the $\mathcal{BB}_1$ ciphertext (which, together, provide the means to distinguish the recipient ID) into two random terms that can no longer be recognized without the proper key. For technical reasons, one has to split the selected component twice independently, resulting in four new elements.

The construction, proposed in [BW06], and based not on the BDH but the Linear assumption [BBS04], is as follows.

Setup Pick $\omega, \alpha, \beta, t_1, t_2, t_3, t_4 \in \mathbb{Z}_p$ and output,

$$\mathit{params} = \begin{pmatrix} P,\ P_1 = \alpha P,\ P_2 = \beta P, \\ Q_1 = t_1 P,\ Q_2 = t_2 P,\ Q_3 = t_3 P,\ Q_4 = t_4 P, \\ v_0 = e(P, \hat{P})^{\omega t_1 t_2} \end{pmatrix} \in {\mathbb{G}_1}^7 \times \mathbb{G}_t\,,$$

$$\mathit{masterk} = (\hat{P}, \hat{P}_1 = \alpha\hat{P}, \hat{P}_2 = \beta\hat{P}, \omega, t_1, t_2, t_3, t_4) \in {\mathbb{G}_2}^3 \times {\mathbb{Z}_p}^5 \ .$$

Extract Pick $r_1, r_2 \in \mathbb{Z}_p$, and output,

$$d_{\mathsf{ID}} = \begin{pmatrix} (r_1 t_1 t_2 + r_2 t_3 t_4)\hat{P}, \\ -\omega t_2 \hat{P} + -r_1 t_2(\hat{P}_1 + \mathsf{ID}\hat{P}_2),\ -\omega t_1 \hat{P} + -r_1 t_1(\hat{P}_1 + \mathsf{ID}\hat{P}_2), \\ -r_2 t_4(\hat{P}_1 + \mathsf{ID}\hat{P}_2),\ -r_2 t_3(\hat{P}_1 + \mathsf{ID}\hat{P}_2) \end{pmatrix} \in \hat{\mathbb{G}}^5 \ .$$

Encrypt Using for $H : \mathbb{G}_t \to \{0,1\}^\ell$ a fixed arbitrary collision-resistant map, pick $s, s_1, s_2 \in \mathbb{Z}_p$, and output,

$$C = \begin{pmatrix} c = M \oplus H(v_0^s),\ c_0 = s(P_1 + \mathsf{ID}P_2), \\ c_1 = (s - s_1)Q_1,\ c_2 = s_1 Q_2, \\ c_3 = (s - s_2)Q_3,\ c_4 = s_2 Q_4, \end{pmatrix} \in \{0,1\}^\ell \times {\mathbb{G}_1}^5\,,$$

Decrypt Given a ciphertext $C = (c, c_0, c_1, c_2, c_3, c_4)$ and a private key $d_{\mathsf{ID}} = (d_0, d_1, d_2, d_3, d_4)$, output,

$$M = c \oplus H\left(\prod_{i=0}^{4} e(c_i, d_i)\right) \in \{0,1\}^\ell \ .$$

5.5. *Fuzzy Attributes*

Another early interesting extension of the Commutative Blinding approach (and the selective-ID secure $\mathcal{BB}_1$ IBE scheme in particular), is to substitute for identities, vectors of binary attributes that only require an imperfect match between key and ciphertext to enable decryption. This notion was first proposed in [SW05]; we refer to that paper for a description.

5.6. *Attribute-Based Encryption*

A further generalization of the previous notion is to consider more complex matching rules between the attributes required to decrypt a ciphertext, and those made available in one's private key. The decryption policy may involve rather complex circuits of threshold gates to define a particular access policy, which can be embedded either in the private key (leading to the notion of "key policy") or in the ciphertext (leading to the notion of "ciphertext policy").

The idea of generalizing IBE for complex policies first appeared in [Sma03], albeit without tackling the issue of collusion attacks, whereby multiple users with different attributes can join forces to circumvent the access policy. Collusion-resistant "attribute-based encryption" was defined in [GPSW06] first, and quickly followed by many extensions and successively refined implementations in a recent series of papers. We refer the reader to Chapter X (in this book) for details and references.

We note that most of the ABE constructions known to date are based on the Commutative Blinding framework, for they generally consist of layers of tricky linear algebra wrapped up in a (selective-ID) basic $\mathcal{BB}_1$ identity-based structure. However, this is not a universal rule: for example, see [Boy07] and also Chapter VI (next in this book) for generic constructions of Fuzzy and ABE schemes from Exponent Inversion rather than Commutative Blinding IBE.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The author and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-83

Chapter VI

Generalized IBE in the Exponent-Inversion Framework

Xavier BOYEN
Voltage Inc., USA

Abstract. The purpose of this chapter is to provide an abstraction for the class of Exponent-Inversion IBE exemplified by the $\mathcal{BB}_2$ and $\mathcal{SK}$ schemes, and, on the basis of that abstraction, to show that those schemes do support interesting and useful extensions such as HIBE and ABE.

Our results narrow, if not entirely close, the "flexibility gap" between the Exponent-Inversion and Commutative-Blinding IBE concepts.

As discussed in the previous chapter, the various pairing-based IBE schemes that are currently known can be classified into three broad categories: (1) Full-Domain Hash, (2) Exponent Inversion, and (3) Commutative Blinding. More of them may of course appear in the future. Each family embodies a particular way that the pairing is made into the identity-based trapdoor mechanism, that matches private keys and ciphertexts conditionally upon their indicated "identities" being the same. Depending on how this is done, one gets different types of IBE constructions, which often translates into very different kinds of complexity assumptions needed for the security proofs. A more practical difference between the three families has to do with the "flexibility" of the core identity-based mechanism, which determines how easily they can be generalized beyond mere IBE.

Over the past several years, the Commutative Blinding approach defined by the $\mathcal{BB}_1$-IBE scheme [BB04a] has shown itself to be the most amenable to generalization (see for instance the chapter on Attribute-Based Encryption: Chapter X in this collection). The Full-Domain Hash family of the $\mathcal{BF}$-IBE scheme [BF03] used to come as a distant second with much fewer known extensions, some of which are nevertheless quite interesting and include broadcast encryption and forward-secure hierarchies (see Chapter VII in this collection). In stark contrast, until recently, the Exponent Inversion family did not afford any extension beyond basic IBE, despite the fact the IBE functionality itself is arguably

Most of the material in this chapter originally appeared in [Boy07].

simpler and has a broader range of core implementations, as $\mathcal{BB}_2$-IBE [BB04a], $\mathcal{SK}$-IBE [CCMLS05,SK03], or Gentry-IBE [Gen06], than in the other two families.[1]

The aim of this chapter is to show that the exponent inversion paradigm is more flexible than had been previously recognized. In particular, we will show how to build hierarchical, fuzzy, and attribute-based IBE systems using generic transformations from any core IBE of the Exponent Inversion variety.

To this end, we shall start by presenting an abstraction of the Exponent Inversion family, that captures the basic IBE functionality along with some additional properties, such as: linearity in the exponent, a sampling property, and the lack of dependence of the session keys on the master key. We call this, Linear IBE. We also model a few simple security properties that such schemes should fulfill, depending on the final goal of the construction; one such property has to do with the possibility of running multiple independent instances of the same core IBE scheme in parallel, and combining the results somehow. This parallel execution trick is a general technique that we use in all our constructions. Last, we exploit our abstraction to transform any black-box Linear IBE with the requisite properties into various kinds of "generalized IBE"; in particular, we shall explicitly consider the cases of hierarchical, fuzzy, and attribute-based IBE.

1. Abstractions for the Exponent-Inversion Framework

Recall that the identity-based trapdoor in exponent-inversion schemes involves a secret polynomial $\theta(\cdot)$ known only to the key-generation authority. Enough information about θ is published that anyone can create a ciphertext $\theta(\mathsf{Id})\, sP$ for any identity Id and any chosen random ephemeral s, and the corresponding session key $e(P, \hat{P})^s$. The private keys given out to the users by the key-generation authority are of the form $\frac{1}{\theta(\mathsf{Id})}\hat{P}$. Pairing those two values would cause a cancellation of the exponents $\theta(\mathsf{Id})$ and $\theta(\mathsf{Id})^{-1}$, yielding a target-group element $e(\theta(\mathsf{Id})\, sP, \theta(\mathsf{Id})^{-1}\hat{P}) = e(P, \hat{P})^s$, revealing the session key.

We start by describing the general structure of exponent-inversion IBE schemes and some of their properties. Three properties of particular interest are: first, a double linearity property (namely, the linear dependence of the session keys in both the private key and the ciphertext); second, independence of session keys from the master secret; and, third, a public sampling property that allows anyone to create a vector of fake private keys for a vector of arbitrary identities, such that the contribution of the vector cancels out completely under a specified linear combination. The last property essentially means that one can publicly sample from the "kernel" of the linear operator defined by application of the "inner product" between a vector of private keys viewed as variables and a vector of ciphertexts viewed as constants.

Double Linearity The first property, double linearity, is the fact that raising either the ciphertext $\theta(\mathsf{Id})\, sP$ or the private key $\frac{1}{\theta(\mathsf{Id})}\hat{P}$ to some power t results in a session key being raised to the same power t. This will come very handy for secret sharing whereupon attribute-based and fuzzy IBE can be implemented. Because session keys constructed this way are linear in both the private key and the ciphertext, it will be easy to construct

[1]The Gentry-IBE scheme does not exactly fit the Exponent Inversion abstraction described here, due to a peculiarity of its security reduction to the q-ABDHE assumption; the structure of the scheme itself does fit the abstraction under appropriately stronger assumptions.

secret sharing schemes in the exponent either in the ciphertext or on the private key side. This is the first property we need (we shall precise and generalize it momentarily).

Session-Key Independence The second property, session-key independence from the master key, is simply the fact that the session key $e(P, \hat{P})^s$ no longer depends on the key-generating authority's root key embodied by the secret polynomial θ. More precisely, provided that the generators P and $\hat{P}$ are fixed and not chosen by the key-generating authority, then the authority cannot influence the session key $e(P, \hat{P})^s$ in any way since its only degree of freedom is the random ephemeral s selected by the encryptor. Observe that this independence property is unique to the Exponent-Inversion family. (Full-Domain-Hash schemes have session keys of the form $e(H(\mathsf{Id}), \alpha\hat{P})^s$, and Commutative-Blinding schemes of the form $e(P, \alpha\hat{P})^s$, where in either case α is a master key chosen by the authority.)

Private-Key "Kernel" Sampling The third property, the existence of a public sampling algorithm for reciprocal or mutually-canceling fake private keys, asks that anyone be able to produce two well-formed but invalid private keys $\vec{d}'_1$ and $\vec{d}'_2$ for identities Id_1 and Id_2 that cancel out when used as private keys on ciphertexts with matching identities. For the well-formed-ness condition, we simply require that the fake private keys be randomized linear combinations of two real private keys Pvk_1 and Pvk_2 for Id_1 and Id_2 under some set of linear coefficients $t_{11}, t_{12}, t_{21}, t_{22}$, though we obviously do not require the sampler to know the value of such coefficients (since being able to compute such coefficients would lead to the public recovery of Pvk_1 and Pvk_2 and hence violate IBE security).

1.1. Linear IBE Template

In order to formalize the preceding properties, we define the following notion of Linear IBE scheme template. This definition was originally proposed in [Boy07].

Definition VI.1 Let Setup, Extract, Encrypt, Decrypt, be a quadruple of algorithms that perform the following operations.

Setup$(\boldsymbol{e}, \boldsymbol{P}, \hat{\boldsymbol{P}}, \boldsymbol{v}, \boldsymbol{\omega})$ on input a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$, generators $P \in \mathbb{G}_1$, $\hat{P} \in \mathbb{G}_2$, $v \in \mathbb{G}_t$, and a random seed ω, outputs a master key pair $\Big(\mathsf{Msk},\ \mathsf{Pub} = (e, P, \hat{P}, v, \ldots)\Big)$.

We require key pairs generated from independent random seeds $\omega_1, \omega_2, \ldots$ to be mutually independent. We allow key pairs generated from the same inputs $e, P, \hat{P}, v, \omega$ to be mutually independent, as the setup algorithm is permitted to use its own source of randomness.

Extract(Msk, Id) on input Msk and an identity Id, outputs a private key $\mathsf{Pvk}_{\mathsf{Id}} = (\mathsf{Id}, R, \vec{d})$, which can be deterministic or randomized.

Here, $\mathsf{Id} \in \mathcal{I}d$, the domain of identities; $R \in \mathcal{R}d$, some non-empty auxiliary domain; and $\vec{d} = (d_1, \ldots, d_n) \in \mathcal{D}$, a vector space of n coordinates, where each coordinate belongs to one of the groups, $\mathbb{F}_p$, $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_t$.

Encrypt(Pub, Id, Msg, $\boldsymbol{s}$**)** on input Pub, a recipient Id, a plaintext Msg, and a randomization exponent $s \in \mathbb{F}_p$, outputs a ciphertext $\mathsf{Ctx} = (\mathsf{Id}, S, c_0, \vec{c})$.

Here, we require that $\mathsf{Msg} \in \mathbb{G}_t$, that $c_0 = \mathsf{Msg} \cdot v^s$, and that $\vec{c} = (c_1, \ldots, c_m) \in \mathcal{C}$, where $\mathcal{C}$ is a vector space of m coordinates, where each coordinate belongs to one of the groups, $\mathbb{F}_p$, $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_t$. Finally, we assume that $S \in \mathcal{S}d$, with $\mathcal{S}d$ some non-empty auxiliary domain.

Decrypt(Pub, Pvk$_{\mathsf{Id}}$, Ctx) on input Pub, a private key $\mathsf{Pvk}_{\mathsf{Id}} = (\mathsf{Id},\ R,\ \vec{d})$, and a ciphertext $\mathsf{Ctx} = (\mathsf{Id},\ S,\ \mathsf{Msg} \cdot v^s,\ \vec{c})$, outputs Msg provided the inputs are well-formed and the identities match.

Any such a quadruple, Setup, Extract, Encrypt, Decrypt, is said to constitute a Linear IBE scheme, if it further satisfies the two following properties.

1. **Double Linearity and Session-Key Independence**
   There exists a (publicly) efficiently computable function, $\mathbf{f}_{\mathsf{Pub}} : \mathcal{I}d \times \mathcal{R}d \times \mathcal{S}d \times \mathcal{C} \times \mathcal{D} \to \mathbb{G}_t$, linear in each of its last two arguments (with respect to the group law in the vector spaces $\mathcal{C}$ and $\mathcal{D}$, and in $\mathbb{G}_t$), such that, for all well-formed $\mathsf{Pvk}_{\mathsf{Id}} = (\mathsf{Id}, R, \vec{d})$ and $\mathsf{Ctx} = (\mathsf{Id}, S, c_0, \vec{c})$,

$$\mathbf{f}_{\mathsf{Pub}}\left(\mathsf{Id},\ R,\ S,\ \vec{c},\ \vec{d}\right) = v^{-s}\ ,$$

   where v is the generator of $\mathbb{G}_t$ given as input to the Setup function prior to the choice of Msk. This Property 1 causes the Linear IBE decryption algorithm to reduce to,

$$\mathsf{Decrypt}(\mathsf{Pvk}_{\mathsf{Id}}, \mathsf{Ctx}) \leftarrow c_0 \cdot \mathbf{f}_{\mathsf{Pub}}(\mathsf{Id}, R, S, \vec{c}, \vec{d})\ .$$

2. **Private-Key "Kernel" Public Sampling**
   Given two (possibly identical) public keys Pub_1 and Pub_2 derived from the same set of parameters $(e, P, \hat{P}, v, \omega)$, auxiliary values R'_1 and R'_2, and identities Id_1 and Id_2 satisfying the condition $\mathsf{Pub}_1 \neq \mathsf{Pub}_2 \vee \mathsf{Id}_1 \neq \mathsf{Id}_2$, one can publicly and efficiently find a pair of "reciprocal private keys" $\vec{d}'_1 = (\hat{d}'_{1,1}, \ldots, \hat{d}'_{1,n})$ and $\vec{d}'_2 = (\hat{d}'_{2,1}, \ldots, \hat{d}'_{2,n})$ such that:

   (a) The distribution of $\vec{d}'_1$ is the same as the distribution of $t_{11}\vec{d}_{11} + t_{12}\vec{d}_{12}$ and the distribution of $\vec{d}'_2$ the same as the distribution of $t_{21}\vec{d}_{21} + t_{22}\vec{d}_{22}$ for some unspecified constants $t_{11}, t_{12}, t_{21}, t_{22}$, where the $\vec{d}_{ij}$ are drawn from genuine private keys $(\mathsf{Id}_j, R, \vec{d}_{ij}) \leftarrow \mathsf{Extract}(\mathsf{Msk}_j, \mathsf{Id}_j)$ for the given identity Id_j conditioned on a matching auxiliary value $R = R'_i$. Thus, we must have an equality of distributions:

$$[\vec{d}'_1] \sim [t_{11}\vec{d}_{11} + t_{12}\vec{d}_{12}]\ ,$$
$$[\vec{d}'_2] \sim [t_{21}\vec{d}_{21} + t_{22}\vec{d}_{22}]\ .$$

   This Requirement 2a ensures that each of $\vec{d}'_1$ and $\vec{d}'_2$ is well-formed, i.e., properly randomized and compatible with the function $\mathbf{f}_{\mathsf{Pub}}$ for the respective auxiliary values R'_1 and R'_2. Notice that the constants $t_{11}, t_{12}, t_{21}, t_{22}$ need not (and must not) be publicly computable; they merely have to exist.

(b) For any two well-formed ciphertexts for Id_1 and Id_2 that have the same exponent s, written $\mathsf{Ctx}_1 = (\mathsf{Id}_1,\, S_1,\, \mathsf{Msg}_1 \cdot v^s,\, \vec{c}_1)$ and $\mathsf{Ctx}_2 = (\mathsf{Id}_2,\, S_2,\, \mathsf{Msg}_2 \cdot v^s,\, \vec{c}_2)$, we have,

$$\mathsf{f}_{\mathsf{Pub}}(\mathsf{Id}_1,\, R'_1,\, S_1,\, \vec{c}_1,\, \vec{d}'_1) \cdot \mathsf{f}_{\mathsf{Pub}}(\mathsf{Id}_2,\, R'_2,\, S_2,\, \vec{c}_2,\, \vec{d}'_2) = v^0 = 1 \ .$$

This Requirement 2b states that the session keys derived from applying the "fake" random private keys $\vec{d}'_1$ and $\vec{d}'_2$ to ciphertexts created with the same ephemeral exponent s, must cancel each other under some fixed linear combination (arbitrarily taken with both coefficients set to 1).

1.2. Parallel IBE Security

Since our goal is to realize complex schemes by combining multiple instances of a basic scheme in various ways, we need to ensure that the multiple instances of the basic scheme can be run simultaneously, with the same group generators and the same encryption ephemeral s. This would give us a weak notion of parallelism for the IBE security reduction, without necessarily requiring full concurrency.

The following definition, taken from [Boy07], captures such notion of parallel IBE semantic security under selective-identity chosen plaintext attack. There are two versions: a selective-ID one, and an adaptive-ID one.

Definition VI.2 Consider the following game played between a Parallel-IBE attacker $\mathcal{A}$ and a Parallel-IBE challenger $\mathcal{B}$. The game parameter ℓ controls the number of IBE instances to be invoked in parallel.

Target (selective-ID) $\mathcal{A}$ announces the "parallel" identities $\mathsf{Id}_1^*, \ldots, \mathsf{Id}_\ell^*$ it intends to attack: one target for each instance of the basic IBE scheme.

Setup $\mathcal{B}$ chooses a common public bilinear context $(e, P, \hat{P}, v)$, randomly picks a secret seed ω, and independent invokes the IBE setup algorithm ℓ times to get $(\mathsf{Msk}_i, \mathsf{Pub}_i) \leftarrow \mathsf{Setup}(e, P, \hat{P}, v, \omega)$ for $i = 1, \ldots, \ell$. All calls to Setup use the same parameters and seed, but different internal random coins. $\mathcal{A}$ gets the single context $(e, P, \hat{P}, v)$ and the ℓ public keys $\mathsf{Pub}_1, \ldots, \mathsf{Pub}_\ell$, which may or may not be the same.

Queries I $\mathcal{A}$ makes private-key extraction queries in all of the ℓ instances of the IBE scheme. Each query may indicate any one of the public keys Pub_i and any adaptively chosen identity Id such that $\mathsf{Id} \neq \mathsf{Id}_i^*$. $\mathcal{B}$ responds to a fresh query with the value $\mathsf{Pvk}_{\mathsf{Id},i} \leftarrow \mathsf{Extract}(\mathsf{Msk}_i, \mathsf{Id})$. It responds to a duplicate query by recalling the earlier answer $\mathsf{Pvk}_{\mathsf{Id},i}$ from storage.

Target (adaptive-ID) $\mathcal{A}$ announces the "parallel" identities $\mathsf{Id}_1^*, \ldots, \mathsf{Id}_\ell^*$ it intends to attack: one target for each instance of the basic IBE scheme.

Challenge $\mathcal{A}$ indicates the two distinct messages Msg_1 and Msg_2 on which it asks to be challenged. $\mathcal{B}$ randomly selects $b \in \{1, 2\}$ and $s \in \mathbb{F}_p$, and creates ℓ ciphertexts $\mathsf{Ctx}_i \leftarrow \mathsf{Encrypt}(\mathsf{Pub}_i, \mathsf{Id}_i^*, \mathsf{Msg}_b, s)$. The challenge consists of the ℓ ciphertexts $\mathsf{Ctx}_1, \ldots, \mathsf{Ctx}_\ell$.

Queries II $\mathcal{A}$ makes more private-key queries, under the same rules as before. The challenger responds as before. For each IBE public key Pub_i, the total number of queries over the two phases may not exceed q.

Guess $\mathcal{A}$ outputs a guess $b' \in \{1, 2\}$, and wins the game if $b' = b$.

An IBE scheme is said to be $(q, \ell, \tau, \epsilon)$-$\mathsf{Par\text{-}IND\text{-}sID\text{-}CPA}$ secure if there is no adversary $\mathcal{A}$ that and wins the preceding selective-ID game in time τ with probability at least $\frac{1}{2} + \epsilon$.

An IBE scheme is said to be $(q, \ell, \tau, \epsilon)$-$\mathsf{Par\text{-}IND\text{-}ID\text{-}CPA}$ secure if there is no adversary $\mathcal{A}$ that and wins the preceding adaptive-ID game in time τ with probability at least $\frac{1}{2} + \epsilon$.

1.3. Parallel-Simulation IBE Security

As in [Boy07], we also define a stronger version of the preceding security notion by giving the adversary access to a supplemental key extraction oracle. The new oracle captures the notion that, even though the adversary is not allowed to ask for private keys on the target identities, it may ask to be given a vector of such keys all raised to the same random power (though useless for a single key, this may potentially leak exploitable information for a vector of keys).

To be secure in the sense of Parallel-Simulation IBE security, the adversary must not gain any significant advantage even when the new queries are allowed. We give a separate definition for this stronger security notion because it is not needed for all generic constructions (even though all the basic IBE schemes we consider do satisfy it).

Definition VI.3 Consider the following game played between a Parallel-IBE attacker $\mathcal{A}$ and a Parallel-IBE challenger $\mathcal{B}$.

Target (selective-ID), as in the Parallel IBE game.

Setup, as in the Parallel IBE game.

Queries I, as in the Parallel IBE game.

Queries I' $\mathcal{A}$ can also make adaptive "parallel simulation" queries across all IBE instances at once. In such query, $\mathcal{A}$ outputs $k + 1$ pairs (i_j, Id_{i_j}) where $\{i_0, \ldots, i_k\} \subseteq \{1, \ldots, \ell\}$ and $\mathsf{Id}_{i_j} \neq \mathsf{Id}^*_{i_j}$ for $j = 1, \ldots, k$. Notice that we explicitly allow $\mathsf{Id}_{i_0} = \mathsf{Id}^*_{i_0}$. In reaction, $\mathcal{B}$ randomly picks $\gamma \in_\$ \mathbb{F}_p^\times$ and for each $j = 0, \ldots, k$ sets $\mathsf{Pvk}_{i_j} = (\mathsf{Id}_{i_j}, R_{i,j}, \vec{d}_{i,j}) \leftarrow \mathsf{Extract}(\mathsf{Msk}_{i_j}, \mathsf{Id}_{i_j})$, or recalls the earlier value from storage if it was ever computed before. The response given to $\mathcal{A}$ is the vector $(\mathsf{Id}_{i_j}, R_{i,j}, \gamma\vec{d}_{i,j})$ for $j = 0, \ldots, k$ and the common γ. Each computation of $\mathsf{Extract}(\mathsf{Msk}_{i_j}, \mathsf{Id}_{i_j})$ counts toward the total quota of q private key queries. Storage recalls from earlier computations are free, and also mandatory if available since $\mathsf{Pvk}_{\mathsf{Id},i}$ is never to be computed twice under different randomizations.

Target (adaptive-ID), as in the Parallel IBE game.

Challenge, as in the Parallel IBE game.

Queries II, as in the Parallel IBE game.

Queries II' $\mathcal{A}$ makes more parallel-simulation queries, under the same rules as in Queries I'. The challenger responds as before. For each IBE public key Pub_i, the total number of queries of all types may not exceed q.

Guess, as in the Parallel IBE game.

An IBE scheme is said to be $(q, \ell, \tau, \epsilon)$-$\mathsf{ParSim\text{-}IND\text{-}sID\text{-}CPA}$ secure if there is no adversary $\mathcal{A}$ that and wins the preceding selective-ID game in time τ with probability at least $\frac{1}{2} + \epsilon$.

An IBE scheme is said to be $(q, \ell, \tau, \epsilon)$-$\mathsf{ParSim\text{-}IND\text{-}ID\text{-}CPA}$ secure if there is no adversary $\mathcal{A}$ that and wins the preceding adaptive-ID game in time τ with probability at least $\frac{1}{2} + \epsilon$.

2. Concrete Instantiations

We now give actual examples of schemes that fit the Linear-IBE template while satisfying the Parallel and Parallel-Simulation security properties of the previous section.

2.1. The "Second" Boneh-Boyen $\mathcal{BB}_2$-IBE

Our prime example is the second IBE by Boneh and Boyen in [BB04a], denoted $\mathcal{BB}_2$, and reproduced below. It originally came with a proof of security against selective-identity attacks from the parametric BDHI assumption in the standard model.

$\mathcal{BB}_2.\mathsf{Setup}$ outputs $\mathsf{Msk} \leftarrow (a,\ b)$ and $\mathsf{Pub} \leftarrow \Big(P,\ P_a = aP,\ P_b = bP,\ v = e(P, \hat{P})\Big)$ for $a, b \in_\$ \mathbb{F}_p$ chosen at random.

$\mathcal{BB}_2.\mathsf{Extract}(\mathsf{Msk}, \mathsf{Id})$ outputs $\mathsf{Pvk}_{\mathsf{Id}} \leftarrow \Big(r_{\mathsf{Id}} = r,\ \hat{d}_{\mathsf{Id}} = \frac{-1}{a+\mathsf{Id}+b\,r}\hat{P}\Big)$ for $r \in_\$ \mathbb{F}_p$.

$\mathcal{BB}_2.\mathsf{Encrypt}(\mathsf{Pub}, \mathsf{Id}, \mathsf{Msg}, s)$ outputs

$$\mathsf{Ctx} \leftarrow (c_0 = \mathsf{Msg} \cdot v^s,\ c_1 = sP_a + s\,\mathsf{Id}P,\ c_2 = sP_b)\ .$$

$\mathcal{BB}_2.\mathsf{Decrypt}(\mathsf{Pub}, \mathsf{Pvk}_{\mathsf{Id}}, \mathsf{Ctx})$ outputs $\mathsf{Msg}' \leftarrow c_0 \cdot e(c_1 + r_{\mathsf{Id}}c_2,\ \hat{d}_{\mathsf{Id}}) \in \mathbb{G}_t$.

Note that the seed ω from the Linear-IBE template is vacant in $\mathcal{BB}_2$; the master key (a, b) is generated by $\mathcal{BB}_2.\mathsf{Setup}$ from independent random coins.

The original proof of security from [BB04a] is extended in [Boy07] to show that $\mathcal{BB}_2$ also satisfies the Parallel-Simulation security property against selective-identity attacks, again from the BDHI assumption in the standard model. The following lemmas are proven in [Boy07].

Lemma VI.4 *$\mathcal{BB}_2$-IBE is a Linear IBE scheme.*

Lemma VI.5 *$\mathcal{BB}_2$-IBE is $(q, \ell, \tau, \epsilon)$-$\mathsf{ParSim\text{-}IND\text{-}sID\text{-}CPA}$ secure in any bilinear context that satisfies the Decision (q', τ', ϵ)-BDHI assumption with $q' > q\,\ell$ and $\tau' < \tau - \Theta(q^2\,\ell^2)$.*

2.2. A Modified "Sakai-Kasahara" $\mathcal{SK}$-IBE

Our next example is that of an identity-based key encapsulation mechanism (IBKEM) studied in [CCMLS05], but whose authors attribute to [SK03], with a security proof based on [BB04a]. Here, we consider an extension of that IBKEM into a full-featured IBE scheme, which, following common usage, we denote $\mathcal{SK}$-IBE.[2]

[2]We note that the Sakai-Kasahara scheme [SK03] is for identity-based key exchange (IBKE) and requires pre-enrollment of the sender. An important difference with $\mathcal{BB}_2$-IBE and $\mathcal{SK}$-IBE is that the actual SK private keys from [SK03] lack both randomization and hashing; for this reason, the original SK scheme has, to this day, no known security proof from non-interactive assumptions.

$\mathcal{SK}$.Setup outputs $\mathsf{Msk} \leftarrow a \in_{\$} \mathbb{F}_p$ and $\mathsf{Pub} \leftarrow \Big(P,\ P_a = aP,\ v = e(P, \hat{P}),\ H\Big)$ where $H : \{0,1\}^* \to \mathbb{F}_p$ is a cryptographic hash function modeled as a random oracle.
$\mathcal{SK}$.$\mathsf{Extract}(\mathsf{Msk}, \mathsf{Id})$ outputs $\mathsf{Pvk}_{\mathsf{Id}} \leftarrow \frac{1}{a+H(\mathsf{Id})}\hat{P}$.
$\mathcal{SK}$.$\mathsf{Encrypt}(\mathsf{Pub}, \mathsf{Id}, \mathsf{Msg}, s)$ gives $\mathsf{Ctx} \leftarrow (c_0 = \mathsf{Msg} \cdot v^s,\ c_1 = (sP_a + s\,H(\mathsf{Id})P))$.
$\mathcal{SK}$.$\mathsf{Decrypt}(\mathsf{Pub}, \mathsf{Pvk}_{\mathsf{Id}}, \mathsf{Ctx})$ outputs $\mathsf{Msg}' \leftarrow c_0 \,/\, e(c_1,\ \mathsf{Pvk}_{\mathsf{Id}}) \in \mathbb{G}_t$.

As in $\mathcal{BB}_2$ above, the Linear-IBE setup seed ω remains uninstantiated in $\mathcal{SK}$; the master key a is generated from independent randomness.

The security proof of $\mathcal{SK}$ is virtually identical to that of $\mathcal{BB}_2$ and rests upon the same parametric BDHI assumption, except for a random-oracle hash "$a + H(\mathsf{Id})$" being thrown in as a substitute for $\mathcal{BB}_2$'s private-key randomization "$a + \mathsf{Id} + b\,r$". We quote the two following lemmas, proven in [Boy07].

Lemma VI.6 *$\mathcal{SK}$-IBE is a Linear IBE scheme.*

Lemma VI.7 *$\mathcal{SK}$-IBE is $(q, \ell, \tau, \epsilon)$-ParSim-IND-ID-CPA secure in any bilinear context that satisfies the Decision (q', τ', ϵ')-BDHI assumption with $q' > q\,\ell$ and $\tau' < \tau - \Theta(q^2\,\ell^2)$, in the random oracle model, where $\epsilon'/\epsilon \geq \prod_{i=1}^{\ell} q_i$, where q_i is the number of adversarial queries to the random oracle that hashes the identities in the i-th IBE subsystem.*

3. Generic Constructions

From any scheme $\mathsf{IBE} = (\mathsf{IBE.Setup}, \mathsf{IBE.Extract}, \mathsf{IBE.Encrypt}, \mathsf{IBE.Decrypt})$ with the above properties, we show how to construct various Generalized IBE systems of interest and prove their security in a generic manner.

3.1. Hierarchical Identities

In a HIBE scheme [HL02,GS02], identities form a multi-level hierarchical tree, and from any identity and its private key one can derive private keys for all the identities that are subordinate to it in the hierarchy, in a process called delegation.

We can turn Linear IBE into HIBE in the following, generic manner [Boy07].

HIBE.Setup(L) Given a security parameter and a number L of levels for the hierarchy:

1. Create bilinear group parameters, $e, P, \hat{P}, v$, at the desired level of security. Also pick an ephemeral shared random seed ω which is kept secret.
2. Generate L sets of IBE master key pairs with common bilinear parameters, $e, P, \hat{P}, v$, by running L instances $(\mathsf{IBE.Msk}_i,\ \mathsf{IBE.Pub}_i) \leftarrow \mathsf{IBE.Setup}(e, P, \hat{P}, v, \omega)$ for $i = 1, \ldots, L$.
3. Select L collision-resistant hash functions (or UOWHFs) from vectors of IBE identities to single identities, $H_i : \mathcal{I}^i \to \mathcal{I}$ for $i = 1, \ldots, L$, where $\mathcal{I}$ is the domain of IBE identities.

4. Output the HIBE master key pair:

$$\mathsf{HIBE.Msk} = (\mathsf{IBE.Msk}_1,\ \ldots,\ \mathsf{IBE.Msk}_L)\ ,$$
$$\mathsf{HIBE.Pub} = (\mathsf{IBE.Pub}_1,\ \ldots,\ \mathsf{IBE.Pub}_L,\ H_1,\ \ldots,\ H_L)\ .$$

HIBE.Extract(Msk, Id) Given $\mathsf{HIBE.Msk}$ and a target identity $\mathsf{Id} = (I_1, \ldots, I_\ell)$ at level $\ell \leq L$:

1. For each $i = 1, \ldots, \ell$, let $h_i = H_i(I_1, \ldots, I_i)$ be the hash of the first i components.
2. For each $i = 1, \ldots, \ell$, extract an IBE private key:

$$(h_i, R_i, \vec{d_i}) \leftarrow \mathsf{Extract}(\mathsf{IBE.Msk}_i, h_i)\ .$$

3. Select $r_1, \ldots, r_\ell \in \mathbb{F}_p$ under the constraint that $\sum_{i=1}^{\ell} r_i = 1 \pmod p$.
4. Output the HIBE private key:

$$\mathsf{HIBE.Pvk}_{\mathsf{Id}} = \Big((I_1, R_1, r_1\vec{d_1}),\ \ldots,\ (I_\ell, R_\ell, r_\ell\vec{d_\ell})\Big)\ .$$

The constraint $\sum_{i=1}^{\ell} r_i = 1 \pmod p$ binds the components to each other to ensure that private keys given to different users are impervious to collusion. It is also what makes decryption work at the lower levels of the hierarchy, as we shall see.

HIBE.Derive(Pvk$_{\mathsf{Id}}$, I'). Given $\mathsf{HIBE.Pvk}_{\mathsf{Id}}$ for an ℓ-level HIBE "parent" identity Id with $\ell < L$, and an IBE identity I' to act as the $(\ell + 1)$-th component of the HIBE "child" identity:

1. Decompose $\mathsf{HIBE.Pvk}_{\mathsf{Id}}$ as a list of triples $(I_i, R_i, \vec{d_i})$ for $i = 1, \ldots, \ell$. Let also $I_{\ell+1} = I'$.
2. For each $i = 1, \ldots, \ell + 1$, let $h_i = H_i(I_1, \ldots, I_i)$ be the hash of the first i components.
3. For each $i = 1, \ldots, \ell$:
   (a) Find two vectors $\vec{d'_{1,i}}$ and $\vec{d'_{2,i}}$ that satisfy Property 2 for $\mathsf{Id}_1 = h_i$ and $\mathsf{Id}_2 = h_{i+1}$ (and the auxiliary R_i and R_{i+1}) relative to the public keys $\mathsf{IBE.Pub}_i$ and $\mathsf{IBE.Pub}_{i+1}$.
   (b) Select $r_i \in \mathbb{F}_p^\times$ and observe that $r_i(\vec{d'_{1,i}})$ and $r_i(\vec{d'_{2,i}})$ also satisfy Property 2.
4. For $i = 1, \ldots, \ell + 1$, define $\vec{d''_i} = \begin{cases} r_1(\vec{d'_{1,1}}) & \text{if } i = 1 \\ r_{i-1}(\vec{d'_{2,i-1}}) + r_i(\vec{d'_{1,i}}) & \text{if } 2 \leq i \leq \ell \\ r_\ell(\vec{d'_{2,\ell}}) & \text{if } i = \ell + 1 \end{cases}$.
5. Output the HIBE private key:

$$\mathsf{HIBE.Pvk}_{\mathsf{Id}'} =$$
$$\Big((I_1, R_1, \vec{d_1} \cdot \vec{d''_1}),\ \ldots,\ (I_\ell, R_\ell, \vec{d_\ell} \cdot \vec{d''_\ell}),\ (I_{\ell+1}, R_{\ell+1}, \vec{d''_{\ell+1}})\Big)\ .$$

Private keys from $\mathsf{HIBE.Derive}$ have the same distribution as if they were created by $\mathsf{HIBE.Extract}$ directly from the master key. Notice that the principle of this key delegation procedure is based on Property 2 of the Linear-IBE template (the existence of a public sampling procedure for vectors of reciprocal, or mutually canceling, private-key factors).

HIBE.Encrypt(Pub, Id, Msg) Given $\mathsf{HIBE.Pub}$, an ℓ-level identity $\mathsf{Id} = (I_1, \ldots, I_\ell)$ where $\ell \leq L$, and a message $\mathsf{Msg} \in \mathbb{G}_t$:

1. Pick a random exponent $s \in \mathbb{F}_p$.
2. $\forall i = 1, \ldots, \ell$, let $h_i = H_i(I_1, \ldots, I_i)$ be the hash of the first i components.
3. $\forall i = 1, \ldots, \ell$, build an IBE ciphertext:

$$\mathsf{Ctx}_i = (h_i, S_i, c_0, \vec{c}_i) \leftarrow \mathsf{Encrypt}(\mathsf{IBE.Pub}_i, h_i, \mathsf{Msg}, s) \ .$$

4. Output the HIBE ciphertext:

$$\mathsf{HIBE.Ctx} = ((h_1, \ldots, h_\ell),\ c_0,\ (S_1, \ldots, S_\ell),\ (\vec{c}_1, \ldots, \vec{c}_\ell)) \ .$$

Notice that $c_0 = \mathsf{Msg} \cdot v^s$ is the same in all the IBE ciphertexts, which is why it is only included once in $\mathsf{HIBE.Ctx}$.

HIBE.Decrypt(Pub, Pvk$_{\mathsf{Id}}$, Ctx) Given $\mathsf{HIBE.Pub}$ and a key $\mathsf{Pvk}_{\mathsf{Id}} = (\mathsf{Pvk}_1, \ldots, \mathsf{Pvk}_\ell)$ and ciphertext $\mathsf{Ctx} = ((h_1, \ldots, h_\ell), c_0, (S_1, \ldots, S_\ell), (\vec{c}_1, \ldots, \vec{c}_\ell))$ for matching ℓ-level identities:

1. $\forall i = 1, \ldots, \ell$, assemble $\mathsf{Ctx}_i = (h_i, 1, S_i, \vec{c}_i)$, using $1 \in \mathbb{G}_t$ in lieu of c_0.
2. $\forall i = 1, \ldots, \ell$, compute the IBE decryption:

$$v_i \leftarrow \mathsf{IBE.Decrypt}(\mathsf{IBE.Pub}_i, \mathsf{Pvk}_i, \mathsf{Ctx}_i) \ .$$

3. Output the decrypted plaintext:

$$\mathsf{Msg} = c_0 \cdot \prod_{i=1}^{\ell} v_i \ .$$

The principle of the decryption algorithm is based on Property 1 of the Linear-IBE template, which implies that $v_i = v^{-s\,r_i}$ on the correct algorithm inputs. Since $\sum_i r_i = 1$ by construction of the hierarchical private keys, the product of the v_i yields the session key v^{-s}.

In the above, $H_1, \ldots, H_L$ are collision-resistant hash functions whose only purpose is to enforce the hierarchical ordering from top to bottom.

This HIBE construction is quite efficient when the underlying IBE is instantiated with $\mathcal{BB}_2$ or $\mathcal{SK}$. Either way, the resulting system requires ℓ pairings for decryption at level ℓ, and is competitive with GS-HIBE [GS02] and $\mathcal{BB}_1$-HIBE [BB04a]. The BBG-HIBE scheme [BBG05], an extension of $\mathcal{BB}_1$, has constant-size ciphertexts and remains more efficient.

We have the following generic security propositions, quoted from [Boy07]. The theorem pertains to selective-ID security, and its corollary to adaptive-ID security.

Theorem VI.8 *The generic* HIBE *scheme is* $(q, \ell, \tau, \epsilon)$-*IND-sHID-CPA secure provided that the underlying* IBE *scheme is a Linear IBE with* $(q, \ell, \tau', \epsilon)$-*Par-IND-sID-CPA security for some* $\tau' \approx \tau$.

Corollary VI.9 *The generic* HIBE *scheme is* $(q, \ell, \tau, \epsilon)$-*IND-HID-CPA secure provided that the underlying* IBE *scheme is a Linear IBE with* $(q, \ell, \tau', \epsilon)$-*Par-IND-ID-CPA security for some* $\tau' \approx \tau$.

PROOF OF THEOREM VI.8: Suppose that there exists an IND-sHID-CPA attacker $\mathcal{A}$ for the HIBE scheme. We construct from it, in a black-box manner, a Par-IND-sID-CPA attacker $\mathcal{B}$ for the IBE scheme.

Target $\mathcal{A}$ starts by giving to $\mathcal{B}$ the HIBE identity Id^* that it intends to attack. W.l.o.g., we assume that Id^* is at the deepest level and thus contains L components. $\mathcal{B}$ expands Id^* into the component identities $I_1^*, \ldots, I_L^*$, computes $h_i^* = H_i(I_1^*, \ldots, I_i^*)$ for each $i = 1, \ldots, L$, and forwards $h_1^*, \ldots, h_L^*$ to the challenger as the IBE identities it is targeting in the parallel IBE game.

Setup $\mathcal{B}$ receives from the challenger a set of bilinear group parameters $(e, P, \hat{P}, v)$ and L public keys for the IBE scheme, $\mathsf{IBE.Pub}_1, \ldots, \mathsf{IBE.Pub}_L$, all based on the stated bilinear parameters. $\mathcal{B}$ assembles them into a public HIBE key in the obvious way, and gives it to $\mathcal{A}$.

Queries I $\mathcal{A}$ makes up to q private key extractions in the HIBE scheme, one at a time. Let $\mathsf{Id} = (\mathsf{Id}_1, \ldots, \mathsf{Id}_\ell)$ be one such HIBE query. $\mathcal{B}$ computes $h_i = H_i(I_1, \ldots, I_i)$ for each $i = 1, \ldots, \ell$. It makes one query to the challenger on the identity h_ℓ under the IBE public key $\mathsf{IBE.Pub}_\ell$, and obtains a private key $\mathsf{IBE.Pvk}_\ell = (h_\ell, R_\ell, \vec{d}_\ell)$. It then computes ℓ vectors $\vec{d}_i''$ for $i = 1, \ldots, \ell$ as in the $\mathsf{HIBE.Derive}$ algorithm, based on Property 2 of Linear IBE schemes. $\mathcal{B}$ then gives to $\mathcal{A}$ the requested private key, $\mathsf{HIBE.Pvk}_{\mathsf{Id}} = \Big((I_1, R_1, \vec{d}_1\, \vec{d}_1''),\ \ldots,\ (I_\ell, R_\ell, \vec{d}_\ell\, \vec{d}_\ell'')\Big)$.

Challenge $\mathcal{A}$ discloses two messages Msg_1 and Msg_2 when it is ready to accept a challenge. $\mathcal{B}$ forwards these messages to its own challenger, and receives L challenge IBE ciphertexts $\mathsf{Ctx}_i^* = (I_i, S_i, c_0, \vec{c}_i)$ on the same plaintext Msg_b for undisclosed $b \in \{1, 2\}$, and created using the same randomization exponent $s \in \mathbb{F}_p$ unknown to $\mathcal{B}$. Notice that all c_0 must be the same. $\mathcal{B}$ challenges $\mathcal{A}$ on the HIBE ciphertext $\mathsf{HIBE.Ctx}^* = ((h_1^*, \ldots, h_L^*),\ c_0,\ (S_1, \ldots, S_L),\ (\vec{c}_1, \ldots, \vec{c}_L))$.

Queries II $\mathcal{A}$ submits a complement of private key extractions queries to reach a total of q over the query phases I and II. $\mathcal{B}$ answers each query as before.

Guess $\mathcal{A}$ finally makes its guess $b' \in \{1, 2\}$, which $\mathcal{B}$ forwards to the challenger as its own guess.

Note that at no time can $\mathcal{A}$ make a query on an identity Id that is a prefix (i.e., a parent or ancestor) of Id^*. It follows from collision resistance that the targets h_i^* will be distinct of all identities h_i on which $\mathcal{B}$ makes a query to the challenger (otherwise we obtain a collision under H_i). The theorem follows. □

PROOF OF COROLLARY VI.9: An analogous proof applies for adaptive-identity security, except that in this case the attacker does not reveal the target to the simulator until the challenge phase, and likewise the simulator to the challenger. □

3.2. Fuzzy Identities

In a Fuzzy IBE scheme [SW05], private keys and ciphertexts pertain to multiple identities (or attributes) at once, and decryption is predicated on a certain number of those attributes being the same in the ciphertext and the private key used to decrypt it. In order to have resistance to collusion, private keys containing different sets of attributes must not be combinable to perform a decryption for a larger set than any of the keys already provided.

Two different schemes are defined in [SW05]: one supporting only an enumerated set of possible attributes, or "small universe", and the other able to represent a fixed number of attributes from an exponentially vast set, or "large universe". In both versions from [SW05], the attributes are merely boolean (either present or absent), which we refer to as "small domain".

Here, we construct a Fuzzy IBE over a "small universe", but generalized to a "large domain", meaning that the enumerated attributes are now key/value pairs that range in all of $\mathbb{F}_p$. Such generalization of Fuzzy IBE may be of independent interest (besides the generic construction), *e.g.*, in biometric encryption systems whose attributes are based on physical measurements.

Follows the generic construction of a small-universe, large-domain, Fuzzy IBE.

FuzzyIBE.Setup(n) Given a security parameter, and the number n of attribute types to support:

1. Create bilinear group parameters, $e, P, \hat{P}, v$, at the desired level of security, and a secret random string ω.
2. Generate n independent IBE master key pairs with shared bilinear parameters, $e, P, \hat{P}, v$, by executing, for $i = 1, \ldots, n$:

$$(\mathsf{IBE.Msk}_i,\ \mathsf{IBE.Pub}_i) \leftarrow \mathsf{IBE.Setup}(e, P, \hat{P}, v, \omega) \ .$$

3. Output the Fuzzy IBE master key pair:

$$\mathsf{FuzzyIBE.Msk} = (\mathsf{IBE.Msk}_1,\ \ldots,\ \mathsf{IBE.Msk}_n) \ ,$$
$$\mathsf{FuzzyIBE.Pub} = (\mathsf{IBE.Pub}_1,\ \ldots,\ \mathsf{IBE.Pub}_n) \ .$$

FuzzyIBE.Extract(Msk, Id, t) On input a master key $\mathsf{FuzzyIBE.Msk}$, a vector $\mathsf{Id} = (I_1, \ldots, I_n)$ of (positionally sensitive) attributes $I_i \in \mathbb{F}_p$, and a threshold parameter t with $1 \leq t \leq n$:

1. Pick $f_1, \ldots, f_{t-1} \in \mathbb{F}_p$ and let $f(x) = 1 + \sum_{i=1}^{t-1} f_i\, x^i$ of degree $t-1$. Note that $f(0) = 1$.
2. For all $i = 1, \ldots, n$, extract an IBE private key:

$$(I_i, R_i, \vec{d_i}) \leftarrow \mathsf{Extract}(\mathsf{IBE.Msk}_i, I_i) \ .$$

3. Output the Fuzzy IBE private key:

$$\mathsf{FuzzyIBE.Pvk}_{\mathsf{Id}} = \Big(t,\ (I_1,\ R_1,\ f(1)\vec{d_1}),\ \ldots,\ (I_n,\ R_n,\ f(n)\vec{d_n})\Big) \ .$$

FuzzyIBE.Encrypt(Pub, Id, Msg) On input a public key $\mathsf{FuzzyIBE.Pub}$, a vector $\mathsf{Id} = (I_1, \ldots, I_n)$ of (positionally sensitive) attributes $I_i \in \mathbb{F}_p$, and a message $\mathsf{Msg} \in \mathbb{G}_t$:

1. Pick a random exponent $s \in \mathbb{F}_p$.
2. $\forall i = 1, \ldots, n$, build an IBE ciphertext:

$$\mathsf{Ctx}_i = (I_i, S_i, c_0, \vec{c}_i) \leftarrow \mathsf{Encrypt}(\mathsf{IBE.Pub}_i, I_i, \mathsf{Msg}, s) \ .$$

3. Output the Fuzzy IBE ciphertext (using $c_0 = \mathsf{Msg} \cdot v^s$ common to all IBE ciphertexts):

$$\mathsf{FuzzyIBE.Ctx} = (\mathsf{Id},\ c_0,\ (S_1, \ldots, S_n),\ (\vec{c}_1, \ldots, \vec{c}_n)) \ .$$

FuzzyIBE.Decrypt(Pub, Pvk$_{\mathsf{Id}}$, Ctx) Given $\mathsf{FuzzyIBE.Pub}$, a private key $\mathsf{Pvk}_{\mathsf{Id}}$, and a ciphertext Ctx:

1. Determine t attributes $I_{i_1}, \ldots, I_{i_t}$ that appear in both $\mathsf{Pvk}_{\mathsf{Id}}$ and Ctx in matching positions.
   (a) If there are fewer than t such "key/value" pairs that match, output $\perp$ and halt.
   (b) Otherwise, select any t matching attributes $I_{i_1}, \ldots, I_{i_t}$ and define $T = \{i_1, \ldots, i_t\}$.
2. For $j = 1, \ldots, t$:
   (a) Extract the IBE private key $(I_{i_j}, R_{i_j}, \vec{d}_{i_j})$ from $\mathsf{Pvk}_{\mathsf{Id}}$ and call it Pvk_j.
   (b) Assemble the IBE ciphertext $(I_{i_j}, 1, S_{i_j}, \vec{c}_{i_j})$ from Ctx and call it Ctx_j.
   (c) Let $\Lambda_{T,i}(x) = \prod_{i' \in T \setminus \{i\}} \frac{x-i'}{i-i'}$ be the Lagrange interpolation coefficients from T to x.
   (d) Compute the IBE decryption:

$$v_j \leftarrow \mathsf{IBE.Decrypt}(\mathsf{IBE.Pub}_j, \mathsf{Pvk}_j, \mathsf{Ctx}_j) \ .$$

3. Output the plaintext:

$$\mathsf{Msg} = c_0 \cdot \prod_{j=1}^{t} {v_j}^{\Lambda_{T,i_j}(0)} \ .$$

By Property 1 of the Linear-IBE schemes, we know that $v_j = v^{-s\, f(i_j)}$ when the algorithm are given the right inputs. The correctness of decryption is easy to see as an application of Lagrange polynomial interpolation "in the exponent":

$$\sum_j f(i_j)\, \Lambda_{T,i_j}(0) = f(0) = 1 \ .$$

When instantiated with $\mathcal{BB}_2$ or $\mathcal{SK}$, our generic construction gives a Fuzzy IBE scheme in the exponent-inversion framework that is competitive with [SW05] — itself an extension of $\mathcal{BB}_1$ in the commutative-blinding framework as discussed in the previous chapter.

The following security theorem and its proof are borrowed from [Boy07].

Theorem VI.10 *The generic* FuzzyIBE *scheme is* (q, n, τ, ϵ)*-IND-sFuzID-CPA secure provided that the base* IBE *scheme is a Linear IBE with* (q, n, τ', ϵ)*-ParSim-IND-sID-CPA security for* $\tau' \approx \tau$.

PROOF: Suppose there is an IND-sFuzID-CPA attacker $\mathcal{A}$ for the Fuzzy IBE scheme. We use it as a black-box to make a ParSim-IND-sID-CPA attacker $\mathcal{B}$ for the underlying IBE scheme.

Target $\mathcal{A}$ starts by giving to $\mathcal{B}$ the Fuzzy IBE identity Id^* that it intends to attack, which consists of a list of n target attributes. $\mathcal{B}$ expands Id^* and forwards the component attributes $I_1^*, \ldots, I_n^*$ to the challenger as the n IBE identities it intends to target in the parallel IBE game.

Setup $\mathcal{B}$ receives from the challenger a set of bilinear group parameters $(e, P, \hat{P}, v)$, and n public keys for the IBE scheme, $\mathsf{IBE.Pub}_1, \ldots, \mathsf{IBE.Pub}_n$, based on those bilinear parameters and an undisclosed common random seed ω. The simulator assembles them into a public Fuzzy IBE key in the obvious way, and gives it to $\mathcal{A}$.

Queries I $\mathcal{A}$ makes up to q private key extractions in the Fuzzy IBE scheme, one at a time. Each such query consists of an identity $\mathsf{Id} = (I_1, \ldots, I_n)$ and a threshold parameter t. We require that the query identity Id match the target identity Id^* in no more than $t - 1$ positions. Suppose w.l.o.g. that the first $t - 1$ attributes are matching, i.e., $I_i = I_i^*$ for $i = 1, \ldots, t-1$, and $I_i \neq I_i^*$ for $i = t, \ldots, n$. To answer the query, $\mathcal{B}$ proceeds as follows.

First, $\mathcal{B}$ makes $n - t$ private key queries to the challenger on identities $I_t, \ldots, I_n$ under the public keys $\mathsf{IBE.Pub}_t$ through $\mathsf{IBE.Pub}_n$. It obtains $n - t$ private keys $\mathsf{IBE.Pvk}_j = (I_j, R_j, \vec{d}_j^{(0)})$ for $j = t, \ldots, n$. Additionally, it finds the (uniquely determined) polynomial $f_0(x)$ of degree $t - 1$ such that $f_0(j) = 0$ for $j \in \{1, \ldots, t-1\}$ and $f_0(0) = 1$.

Next, $\mathcal{B}$ makes, for each $i = 1, \ldots, t-1$, a "parallel simulation" query on the list of identities $I_i, I_t, \ldots, I_n$ in their respective IBE subsystems. Notice that except for I_i none of these is a target identity, so the query is legal. For each i, it obtains $n - t + 2$ fake "private keys" $\mathsf{Pvk}_j^{(i)} = (I_j, R_j, \vec{d}_j^{(i)})$ for $j \in \{i\} \cup \{t, \ldots, n\}$: these are incorrect keys that all share the same discrete log relationship to the right key. Additionally, for each i, it determines the polynomial $f_i(x)$ of degree $t - 1$ such that $f_i(j) = 0$ for $j \in \{0, \ldots, t-1\} \setminus \{i\}$ and $f_i(i) = 1$.

Last, $\mathcal{B}$ gives to $\mathcal{A}$ the Fuzzy IBE private key assembled as,

$$\mathsf{FuzzyIBE.Pvk}_{\mathsf{Id}} =$$

$$\left(t,\ \left(I_1,\ R_1,\ \sum_{i=0}^{t-1} f_i(1)(\vec{d}_1^{(i)})\right),\ \ldots,\ \left(I_n,\ R_n,\ \sum_{i=0}^{t-1} f(n)(\vec{d}_n^{(i)})\right)\right) \quad .$$

The key is correctly distributed because the $t - 1$ "parallel simulation" queries have contributed $t - 1$ independent random exponents $\gamma^{(1)}, \ldots, \gamma^{(t-1)}$ into the respective $\vec{d}_i^{(1)}, \ldots, \vec{d}_i^{(t-1)}$.

Challenge $\mathcal{A}$ announces two messages Msg_1 and Msg_2 when it is ready to accept a challenge.

$\mathcal{B}$ forwards these messages to its own challenger, and receives n challenge IBE ciphertexts $\mathsf{Ctx}_i^* = (I_i, S_i, c_0, \vec{c}_i)$ on the same plaintext Msg_b for undisclosed $b \in \{1, 2\}$, and created using the same randomization exponent $s \in \mathbb{F}_p$ unknown to $\mathcal{B}$. Notice that all c_0 must be the same.

$\mathcal{B}$ challenges $\mathcal{A}$ on the Fuzzy IBE ciphertext,

$$\mathsf{FuzzyIBE.Ctx}^* = (\mathsf{Id}^*,\ c_0,\ (S_1, \ldots, S_n),\ (\vec{c}_1, \ldots, \vec{c}_n)) \ .$$

Queries II $\mathcal{A}$ submits a complement of private key extractions queries to reach a total of q over the query phases I and II (including the "parallel simulation" queries, each of which counts as 0 or 1 regular key extraction since for each such query, a key will have already been extracted for all identities $I_t, \ldots, I_n$ except I_i). $\mathcal{B}$ answers each query as before.

Guess $\mathcal{A}$ finally makes its guess $b' \in \{1, 2\}$, which $\mathcal{B}$ forwards to the challenger as its own guess.

Note that $\mathcal{B}$ never made a (direct) private key query on any of the n selected IBE identities that correspond to its own challenge. Thus, the theorem follows from the black-box reduction. □

3.3. Attribute-Based Encryption

Attribute-based encryption (ABE) is a generalization of Fuzzy IBE recently proposed in [GPSW06]. Instead of allowing decryption conditionally on the satisfaction of a number of matching attributes above a certain threshold (which can be thought of as a threshold gate whose inputs are the matching attributes in the ciphertext and the key), ABE allows that condition to be evaluated not as a single gate but as a tree of threshold gates.

The construction given in [GPSW06] generalizes the Fuzzy IBE construction of [SW05] in the commutative-blinding approach, and is based on the use of not one but multiple interpolation polynomials $f(x)$, each of which applies to a subset of the input attributes. The degrees of the random polynomials and their inputs determine the access structure in the ABE scheme. In key-policy (KP-ABE) schemes, the access structure is chosen by the authority that issues the private keys.

Our generic exponent-inversion framework can mirror the KP-ABE construction of [GPSW06], in the same way that it did for the Fuzzy IBE construction of [SW05]. And, as before, as a bonus we actually obtain a "large domain" generalization of ABE, which allows key/value pairs for attributes instead of booleans.

Both Fuzzy IBE and KP-ABE are based on the implementation of an access policy in the private keys given to the users. Whereas in Fuzzy IBE the access policy is a single t-out-of-n threshold gate, in KP-ABE it can comprise many threshold gates arranged in a tree. Notice that conjunctions and disjunctions are special cases of threshold gate with $t = n$ and $t = 1$ respectively. A fairly large variety of policies can be encoded compactly using such an access structure; we refer to [KW93,GPSW06] for more details on access trees.

The generic construction is mainly a transcription of [GPSW06] to use our exponent-inversion abstraction, and is as follows [Boy07].

kpABE.Setup(n) Given a security parameter, and the total number N of attributes to support:

1. Create bilinear group parameters, $e, P, \hat{P}, v$, at the desired level of security.
2. Generate N independent IBE master key pairs for the same bilinear parameters, $e, P, \hat{P}, v$, from N executions $(\mathsf{IBE.Msk}_i,\ \mathsf{IBE.Pub}_i) \leftarrow \mathsf{IBE.Setup}(e, P, \hat{P}, v)$ for $i = 1, \ldots, N$.
3. Output the KP-ABE master key pair:

$$\mathsf{kpABE.Msk} = (\mathsf{IBE.Msk}_1,\ \ldots,\ \mathsf{IBE.Msk}_N)\ ,$$
$$\mathsf{kpABE.Pub} = (\mathsf{IBE.Pub}_1,\ \ldots,\ \mathsf{IBE.Pub}_N)\ .$$

kpABE.Extract(Msk, Id, $\mathcal{T}$**)** This algorithm takes as inputs the master key, a vector $\mathsf{Id} = (I_1, \ldots, I_N)$ of attribute values $I_i \in \mathbb{F}_p$, and a policy $\mathcal{T}$.
The policy is represented as a tree $\mathcal{T}$, whose interior nodes are threshold gates with multiple boolean inputs and a single (non-duplicated) boolean output, and whose N leaf nodes are comparison gates that test equality of each input attribute to the respective target value I_i.
The private key should enable its owner to decrypt a ciphertext encrypted for $\mathsf{Id}' = (I'_1, \ldots, I'_N)$ if and only if $\mathcal{T}_{I_1,\ldots,I_N}(I'_1, \ldots, I'_N)$ evaluates to true. The key is constructed recursively using a similar method as in [GPSW06], as follows.

1. Give each node of $\mathcal{T}$ a unique index in $\mathbb{F}_p$ (use 0 for the root and $1, \ldots, N$ for the leaves).
2. Visit all interior nodes in a top-down order, and assign to each a polynomial $f_i(x)$ of degree $t_i - 1$, where t_i is the threshold of the gate at node i, under the constraints:
   (a) For $i = 0$, we require that $f_0(0) = 1$.
   (b) For $i > N$, we require that $f_i(0) = f_{\text{parent}}(i)$.
   (f_{parent} is the previously defined polynomial given to the parent node of i.)
3. Assign to each leaf node $i = 1, \ldots, N$, the constant $f_i = f_{\text{parent}}(i)$.
4. For each $i = 1, \ldots, N$, extract an IBE private key

$$(I_i, R_i, \vec{d_i}) \leftarrow \mathsf{Extract}(\mathsf{IBE.Msk}_i, I_i)\ .$$

5. Output the KP-ABE private key:

$$\mathsf{kpABE.Pvk}_{\mathsf{Id}} = \Big(\mathcal{T},\ (I_1, R_1, f_1\vec{d_1}),\ \ldots,\ (I_N, R_N, f_N\vec{d_N})\Big)\ .$$

We have assumed that all N attribute positions appear in the policy, with one value each. If an attribute is missing, its exponent f_i will be undetermined, and its tuple $(I_i, R_i, f_i\vec{d_i})$ can be omitted from the private key. Conversely, multiple appearances of an attribute in the policy (with the same or different values) can be accommodated using multiple, independently extracted tuples with their own values of f_i.

kpABE.Encrypt(Pub, Id, Msg) On input the public key, a vector $\mathsf{Id} = (I_1, \dots, I_N)$ of attribute values $I_i \in \mathbb{F}_p$, and a message $\mathsf{Msg} \in \mathbb{G}_t$:

1. Pick a random exponent $s \in \mathbb{F}_p$.
2. $\forall i = 1, \dots, N$, build an IBE ciphertext:

$$\mathsf{Ctx}_i = (I_i, S_i, c_0, \vec{c}_i) \leftarrow \mathsf{Encrypt}(\mathsf{IBE.Pub}_i, I_i, \mathsf{Msg}, s) \ .$$

3. Output the KP-ABE ciphertext (using the common $c_0 = \mathsf{Msg} \cdot v^s$):

$$\mathsf{kpABE.Ctx} = (\mathsf{Id},\ c_0,\ (S_1, \dots, S_N),\ (\vec{c}_1, \dots, \vec{c}_N)) \ .$$

This description assumes that each attribute takes exactly one value. As with the private key, attributes can be omitted or replicated with multiple values, simply by excluding or including the corresponding IBE ciphertexts in the obvious way.

kpABE.Decrypt(Pub, Pvk$_{\mathsf{Id}}$, Ctx) Given the public key, a private key $\mathsf{Pvk}_{\mathsf{Id}}$, and a ciphertext Ctx, we decrypt using a recursive algorithm that can be viewed as a generic version of [GPSW06].

For a node i, we define the boolean function $\mathcal{T}_i(I_1, \dots, I_N)$ to be the output value of the gate at node i, when the inputs to the leaves of $\mathcal{T}$ are the attribute values $I_1, \dots, I_N$.

We first define the recursive function $\mathsf{DecryptNode}(i)$:

1. If $\mathcal{T}_i(I_1, \dots, I_N) = \texttt{false}$, then return $v_i \leftarrow \perp$.
2. If i is a leaf node, i.e., $1 \le i \le N$, then:
   (a) Parse $\mathsf{Pvk}_i = (I_i, R_i, \vec{d}_i)$ from $\mathsf{Pvk}_{\mathsf{Id}}$.
   (b) Build $\mathsf{Ctx}_i = (I_i, 1, S_i, \vec{c}_i)$ from Ctx. Note the "1".
   (c) Return $v_i \leftarrow \mathsf{IBE.Decrypt}(\mathsf{IBE.Pub}_i, \mathsf{Pvk}_i, \mathsf{Ctx}_i)$.
3. If i is an interior node with a t-out-of-n threshold gate, then:
   (a) Gather a set of t children, $J = \{i_1, \dots, i_t\}$, such that $\forall j \in J$, $\mathcal{T}_j(I_1, \dots, I_N) = \texttt{true}$. Such a set J must exist since $\mathcal{T}_i(I_1, \dots, I_N) = \texttt{true}$.
   (b) Let $\Lambda_{J,j}(x) = \prod_{j' \in J \setminus \{j\}} \frac{x-j'}{j-j'}$ be the Lagrange interpolation coefficients from J to x.
   (c) For each $j \in J$, compute $v_j \leftarrow \mathsf{DecryptNode}(j)$. We must get all $v_j \neq \perp$.
   (d) Return $v_i \leftarrow \prod_{j \in J} (v_j)^{\Lambda_{J,j}(0)}$.

We then decrypt by outputting $\mathsf{Msg} \leftarrow \mathsf{DecryptNode}(0)$.

One can easily verify that the system works as expected.

The construction is indeed very similar to the KP-ABE scheme of [GPSW06] in the commutative blinding framework. The main difference is that the "polynomial algebra in the exponent" is applied on the generic private keys from the underlying IBE systems. The result is a semi-generic construction that can be instantiated with any Exponent Inversion compliant IBE scheme.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-100

Chapter VII

Forward-Secure Hierarchical IBE with Applications to Broadcast Encryption

Danfeng (Daphne) YAO [a], Nelly FAZIO [b],
Yevgeniy DODIS [c] and Anna LYSYANSKAYA [d]

[a] *Rutgers University, USA*
[b] *IBM Almaden Research Center, USA*
[c] *New York University, USA*
[d] *Brown University, USA*

Abstract. A forward-secure encryption scheme protects secret keys from exposure by evolving the keys with time. Forward security has several unique requirements in hierarchical identity-based encryption (HIBE) scheme: (1) users join dynamically; (2) encryption is joining-time-oblivious; (3) users evolve secret keys autonomously.

We define and construct a scalable pairing-based forward-secure HIBE (fs-HIBE) scheme satisfying all of the above requirements. We also show how our fs-HIBE scheme can be used to realize a forward-secure public-key broadcast encryption scheme, which protects the secrecy of prior transmissions in the broadcast encryption setting. We further generalize fs-HIBE into a collusion-resistant multiple hierarchical ID-based encryption scheme, which can be used for secure communications with entities having multiple roles in role-based access control. The security of our schemes is based on the bilinear Diffie-Hellman assumption in the random oracle model.

Keywords. Forward security, ID-based encryption, broadcast encryption

1. Forward Security

The idea of an identity-based encryption (IBE) scheme is that an arbitrary string can serve as a public key. The main advantage of this approach is to largely reduce the need for public key certificates and certificate authorities, because a public key is associated with identity information such as a user's email address. A first scheme for identity-based encryption ($\mathcal{BF}$-$\mathcal{IBE}$) was based on the bilinear Diffie-Hellman assumption in the ran-

The preliminary version of this chapter has been published in the Proceedings of the ACM Conference on Computer and Communications Security (CCS '04) [YFDL04].

dom oracle model by Boneh and Franklin [BF03]. In IBE schemes private key generator (PKG) is responsible for generating private keys for all users, and therefore is a performance bottleneck for organizations with large number of users. Hierarchical identity-based encryption (HIBE) schemes [BBG05,GS02] were proposed to alleviate the workload of a root PKG by delegating private key generation and identity authentication to lower-level PKGs. In a HIBE scheme, a root PKG needs only to generate private keys for domain-level PKGs, who in turn generate private keys for users in their domains in the next level. The organization of PKGs and users forms a hierarchy that is rooted by the root PKG. To encrypt a message, Alice needs to obtain the public parameters of Bob's root PKG, and the ID for Bob and for those domain-level PKGs that are on the path from the root to Bob; there are no lower-level parameters. Gentry and Silverberg [GS02] extended $\mathcal{BF}$-$\mathcal{IBE}$ scheme and presented a fully scalable hierarchical identity-based encryption ($\mathcal{GS}$-$\mathcal{HIBE}$) scheme. A HIBE construction with a weaker notion of security was given by Boneh and Boyen [BB04a]. Later, new IBE and HIBE constructions that can be proved to have the full security without the random oracle model [BB04b,Wat05] were given.

Due to the inherent key-escrow property, the standard notion of HIBE security crucially depends on secret keys remaining secret. Key exposure is a realistic threat over the lifetime of such a scheme. To mitigate the damage caused by the exposure of secret key information in HIBE, one way is to construct a forward-secure hierarchical identity-based encryption (fs-HIBE) scheme that allows each user in the hierarchy to refresh his or her private keys periodically while keeping the public key the same. A forward-secure public-key encryption scheme has recently been presented by Canetti, Halevi and Katz [CHK03]. But surprisingly, a practical fs-HIBE scheme has several unique requirements that cannot be achieved by trivial combinations of the existing fs-PKE schemes [CHK03,Kat02] and HIBE schemes [GS02].

Apart from being interesting on its own, fs-HIBE is a useful tool that lends itself to several applications. One such application is the implementation of forward secrecy for public-key broadcast encryption. While forward secrecy is an important requirement in any context, it is especially needed for broadcast encryption [FN93,GSW00]. This is because by design an adversary can freely listen to any broadcast and store it. Then, should the adversary ever succeed in recovering *any* user's secret key, she will manage to decrypt all past broadcasts that such user was authorized to receive *unless* we have forward secrecy.

In our preliminary version [YFDL04], we posed an interesting open question that whether a general fs-HIBE scheme with linear or even sub-linear complexity can be realized. Shortly afterward, Boneh, Boyen, and Goh were able to construct an efficient HIBE system with constant-size ciphertexts [BBG05] under a different security assumption. Their HIBE scheme can also be extended to achieve forward-security with constant ciphertexts. The security of Boneh, Boyen, and Goh's system is based on a weaker version of Diffie-Hellman Inversion (BDHI) assumption. The BDHI assumption was previously used to construct a selective-ID secure IBE without random oracles [BB04a]. In comparison, our system is only based on the Bilinear Diffie-Hellman (BDH) assumption. The 1-BDHI assumption is equivalent to the standard BDH assumption. It is not known if the h-BDHI assumption, for $h > 1$, is equivalent to BDH [BB04a].

Below, we discuss the notion of forward security for HIBE in more detail, and then explain why it cannot be trivially achieved by existing techniques such as a combination of fs-PKE [CHK03] and HIBE [BB04a,GS02] schemes.

1.1. Forward Secrecy

The central idea of forward secrecy is that the compromise of long-term keys does not compromise past session keys and therefore past communications. This notion was first proposed by Günther [Gün90] and later by Diffie *et al.* [DvW92] in key exchange protocols. The notion of non-interactive forward security was proposed by Anderson [And00] in 1997 and later formalized by Bellare and Miner [BM99], who also gave a forward-secure signature scheme followed by a line of improvement [AMN01,MMM02]. In this model, secret keys are updated at regular intervals throughout the lifetime of the system; furthermore, exposure of a secret key corresponding to a given interval does not enable an adversary to break the system (in the appropriate sense) for any prior time period. The model inherently cannot prevent the adversary from breaking the security of the system for any subsequent time period. Bellare and Yee [BY03] provided a comprehensive treatment of forward security in the context of private key based cryptographic primitives.

The first forward-secure public-key encryption (fs-PKE) scheme was given by Canetti, Halevi, and Katz [CHK03] based on the Gentry-Silverberg HIBE [GS02] scheme. The fs-PKE scheme constructs a binary tree, in which a tree node corresponds to a time period and has a secret key. Children of a node w are labeled $w0$ and $w1$, respectively. Given the secrets corresponding to a prefix of a node representing time t, one can compute the secrets of time t. In order to make future keys computable from the current key, the secrets associated with a prefix of a future time are stored in the current key. After the key for the next time period is generated, the current decryption key is erased. The state-of-the-art fs-PKE scheme [CHK03] is based on the decisional bilinear Diffie-Hellman assumption [BF03] in the standard model. Canetti, Halevi and Katz also gave a more efficient scheme in the random oracle model [CHK03].

1.2. Requirements of a fs-HIBE Scheme

Intuitively, forward security in a HIBE scheme implies that compromise of the current secret key of a user only leads to the compromise of the user and his descendants' subsequent communications. We will give a formal definition of security in Section 2.2. Our design of a forward-secure HIBE scheme also takes system properties such as scalability and efficiency into consideration. This is essential in the management of large scale distributed systems. Below, we define the requirements for a scalable forward-secure HIBE scheme.

- New users should be able to join the hierarchy and receive secret keys from their parent nodes *at any time*.
- Encryption is *joining-time-oblivious*, which means that the encryption does not require knowledge of when a user or any of his ancestors joined the hierarchy. The sender can encrypt the message as long as he knows the current time and the ID-tuple of the receiver, along with the public parameters of the system.
- The scheme should be forward-secure.

- Refreshing secret keys can be carried out *autonomously*, that is, users can refresh their secret keys on their own to avoid any communication overhead with any PKG.

Surprisingly, the design of a fs-HIBE scheme that fulfills the above system requirements turns out to be non-trivial, despite the fact that both HIBE [GS02] scheme and fs-PKE [CHK03] scheme are known. Intuitive combinations of the two schemes fail to achieve all the desired system features. Next, we explain why this is the case.

1.3. Some Forward-Secure HIBE Attempts

In this section, we make three simple forward-secure HIBE constructions based on HIBE scheme [GS02] and fs-PKE scheme [CHK03], and explain why these naive schemes do not satisfy the requirements of a practical fs-HIBE scheme.

1.3.1. Scheme I

Consider a scheme based on the HIBE [GS02] scheme. The user with a given ID tuple $(ID_1, \ldots, ID_h)$ maintains two sub-hierarchies (subtrees): the time subtree that evolves over time for forward security (as in fs-PKE [CHK03]), and the ID subtree to which other nodes are added as children join the hierarchy. To encrypt a message for this user at time t, use the HIBE with identity $(ID_1, \ldots, ID_h, t)$. The user can decrypt this message using HIBE decryption, using the fact that he knows the key from the time subtree. The user's children are added to the hierarchy into the ID subtree.

However, Scheme I has the following issue. Suppose a user never erases the secret key corresponding to the root of his ID subtree. Then should this key ever be exposed, the forward secrecy of his children is compromised. On the other hand, if this secret key is ever erased, then no nodes can be added as children of $(ID_1, \ldots, ID_h)$ in the hierarchy, and so this scheme will not support dynamic joins.

The lesson we learn from this failed scheme is that all keys must be evolved together.

1.3.2. Scheme II

Let us try to repair Scheme I by making sure that the key from which children's keys are derived is also evolving over time. In Scheme II, the public key of a user consists of alternating ID-tuples and time strings, which is referred to as an *ID-time-tuple*. The private key of a user serves three purposes: decryption, generating private keys for new children, and deriving future private keys of the user. The public key of a newly joined child is the parent's ID-time-tuple appended with the child's ID. That key is in turn used for generating keys for lower-level nodes further down the hierarchy. For example, if Alice joins Bob, the root, at time (*January, Week 1*) and Eve joins Alice at time (*January, Week 2*), Eve's public key is (Bob, *January, Week 1*, Alice, *January, Week 2*, Eve). Encrypting a message to Eve requires the sender to know when Eve and all her ancestors joined the system. Therefore Scheme II is not joining-time-oblivious.

The lesson we learn from the failed Scheme II is that the keys must evolve in a way that is transparent to the encryption algorithm.

1.3.3. Scheme III

In our final unsuccessful attempt, Scheme III, a user adds a child to the hierarchy by giving him or her secret keys that depend both on the current time and on the child's position in the hierarchy. This is achieved by requiring that messages may only be decrypted by those who know two keys: one corresponding to the current time and the other corresponding to their positions in the hierarchy. Each user autonomously evolves his time key, and gives his newly joined children his time key in addition to their ID keys.

It is easy to see that this scheme is *n*ot forward-secure. An adversary who joins the hierarchy at the beginning of time can corrupt a user at any future time and obtain his or her ID key. Moreover, this adversary can derive any past time key (because he joined at the beginning of time). Thus, this adversary may decrypt any past message addressed to the exposed user.

For the same reason, the multiple hierarchical identity-based encryption (MHIBE) scheme generalized from Scheme III is not collusion-resistant, where the ciphertext for a user with multiple identities can be decrypted if some other individuals collude. MHIBE scheme is useful for secure communications with entities having multiple identities, and is described in Sections 1.4.3 and 5.

1.3.4. Comparisons

All the above trivial approaches fail. Constructing a forward-secure hierarchical ID-based encryption scheme that is both secure and scalable is not so straightforward. Our implementation, which is described in next section, is still based on $\mathcal{GS}$-$\mathcal{HIBE}$ [GS02] scheme and fs-PKE [CHK03] scheme. Yet, it overcomes the problems existing in naive combinations of the two schemes, and satisfies the requirements of supporting dynamic joins, joining-time-obliviousness, forward security, and autonomous key updates.

1.4. Overview

We describe several cryptographic constructions. First, we present a scalable and joining-time-oblivious forward-secure hierarchical identity-based encryption scheme that allows keys to be updated autonomously. Second, we show how our fs-HIBE scheme can be used to obtain a forward-secure public-key broadcast encryption (fs-BE) scheme. Third, we generalize our fs-HIBE scheme and discuss its application in secure communications with entities having multiple roles in role-based access control (RBAC) [SCFY96].

1.4.1. Forward-secure HIBE scheme

Our fs-HIBE protocol is based on the HIBE scheme by Gentry and Silverberg [GS02] and forward-secure public-key encryption (fs-PKE) [CHK03] scheme due to Canetti, Halevi and Katz. It satisfies the requirements of dynamic joins, joining-time-obliviousness, forward security, and autonomous key updates.

A HIBE scheme involves only one hierarchy, whereas a fs-HIBE scheme has two hierarchies: ID and time. Each (ID-tuple, time) pair can be thought of as a point on the two-dimensional grid as follows. On the x-axis, we start with the identity of the root Public Key Generator in the ID hierarchy (e.g. Hospital), then in position (1,0) we have the identity of the first-level PKG (e.g. ER). In position (2,0) there is the identity of the second level PKG (e.g. Doctor), and in position (3,0) there may be another PKG or an individual user (e.g. Bob). Thus the x-axis represents an ID-tuple, for example (Hospital, ER, Doctor, Bob). Similarly, the y-axis represents the time. Divide a duration of time into multiple time periods and arrange them as leaf nodes of a tree. Internal nodes of the tree represent the time spans associated with their child nodes. Then, the origin of the grid corresponds to the root of the time hierarchy (e.g. 2005). In position (0, 1) we have the first level of the time hierarchy (e.g. January), and in position (0, 2) there is the next level of time hierarchy (e.g. Week 1). Thus a time period can be expressed as a tuple on the y-axis, for example (2005, January, Week 1). Figure VII.1 gives a schematic drawing of the correspondence between the tuples and keys in fs-HIBE.

In a fs-HIBE scheme, the secret key of an (ID-tuple, time) pair is associated with some path on the grid. For each grid point on that path, there is a corresponding element in this secret key. Such a path (secret key) is *not* joining-time-oblivious: it depends on when the user, as well as the nodes higher up, join the system. However, when encrypting, the sender does not have to know the path. What is non-trivial here is that, the path (secret key) and ciphertext of our fs-HIBE scheme are designed in such a way that we do not need to come up with a separate ciphertext for each possible path in order to achieve joining-time-obliviousness.

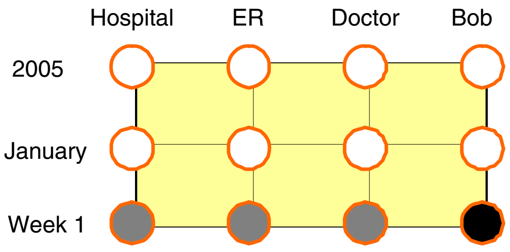


Figure VII.1. A schematic drawing of keys for ID-tuple (Hospital, ER, Doctor, Bob) at time period (2005, January, Week 1) in a forward-secure HIBE scheme. The ID-tuple (Hospital, ER, Doctor, Bob) of Bob is on x-axis. The tuple representing time period (2005, January, Week 1) is on y-axis. The origin represents the root identity (Hospital) and the highest-level time period (2005). The black node represents Bob's key at Week 1. The gray nodes correspond to keys of Bob's ancestors at Week 1. Each white node represents an intermediate key. Secret keys at both the grey and white nodes can be used to compute private keys for Bob.

Our fs-HIBE scheme has collusion resistance and chosen ciphertext security in the random oracle model [BR93] assuming the difficulty of the bilinear Diffie-Hellman problem [BF03,CHK03,GS02], provided that the depths of the ID hierarchy and time hierarchy are bounded by constants. Our fs-HIBE scheme is provable secure under full-identity chosen-ciphertext model (ind-id-cca). The complexities of various parameters in our fs-HIBE scheme are summarized in Table VII.1 and are discussed in Section 6.

1.4.2. Forward-secure broadcast encryption scheme

We show how our fs-HIBE scheme can be used to construct a scalable forward-secure public-key broadcast encryption (fs-BE) scheme, which protects the secrecy of prior transmissions. A broadcast encryption (BE) [DF03a,DF03b,GST04,HS02,KHL03, NNL01,NP00,TT01] scheme allows content providers to securely distribute digital contents to a dynamically changing user population. Each active user is issued a distinct secret key when he joins the system, by a trusted center. In comparison with the symmetric-key setting, a public-key BE scheme of [DF03a] has a single public key associated with the system, which allows the distribution of the broadcast workload to untrusted third parties.

In a scalable forward-secure public-key broadcast encryption (fs-BE) scheme, users should be able to update their secret keys autonomously, and the trusted center should allow users to dynamically join the broadcast system at any time while achieving forward security. In addition, each content provider does not need to know when each user joins the system in order to broadcast the encrypted contents. The encryption algorithm of a fs-BE scheme should only depend on the current time and the set of authorized users, and thus be joining-time-oblivious. Applying our fs-HIBE to the public-key BE scheme [DF03a] yields such a fs-BE scheme.

1.4.3. Multiple hierarchical ID-based encryption

We further generalize our forward-secure hierarchical ID-based encryption scheme into a collusion-resistant multiple hierarchical identity-based encryption (MHIBE) scheme, and describe its application in secure communications with individuals who have multiple roles in role-based access control (RBAC) [SCFY96]. In large-scale organizations, a user may own multiple identities, each of which is represented by an ID-tuple. In MHIBE, a message can be encrypted under multiple ID-tuples (identities) and can be decrypted only by those who have *all* the required identities. The collusion-resistant property cannot be achieved using separate HIBE schemes. We note that the fs-HIBE scheme is a special case of our MHIBE scheme, in that in fs-HIBE scheme, time can be viewed as another identity of a user. Therefore the identities in MHIBE scheme capture a broad sense of meaning.

2. Forward-Secure HIBE (fs-HIBE)

This section defines the notion of forward secrecy for HIBE scheme and the related security. In a fs-HIBE scheme, secret keys associated with an ID-tuple are evolved with time. At any time period i an entity joins the system (hierarchy), its parent node computes its decryption key corresponding to time period i and other values necessary for the entity to compute its own future secret keys. Once the newly joined entity receives this secret information, at the end of each period it updates its secret key and erases the old key. During time period i, a message is encrypted under an ID-tuple and the time i. Decryption requires the secret key of the ID-tuple at time i.

Time Period As usual in forward-secure public-key encryption [CHK03] scheme, we assume for simplicity that the total number of time periods N is a power of 2; that is $N = 2^l$.

ID-tuple An entity has a position in the hierarchy, defined by its tuple of IDs: $(\mathrm{ID}_1, \ldots, \mathrm{ID}_h)$. The entity's ancestors in the hierarchy are the users / PKGs whose ID-tuples are $\{(\mathrm{ID}_1, \ldots, \mathrm{ID}_i){:}1 \leq i < h\}$. ID_1 is the ID for the root PKG.

2.1. fs-HIBE: Syntax

Forward-Secure Hierarchical ID-Based Encryption Scheme (fs-HIBE) A fs-HIBE scheme is specified by five algorithms: Setup, KeyDer, Upd, Enc, and Dec.

Setup The root PKG takes a security parameter k and the total number of time periods N, and returns $params$ (system parameters) and the initial root key $SK_{0,1}$. The system parameters include a description of the message space $\mathcal{M}$ and the ciphertext space $\mathcal{C}$. The system parameters will be publicly available, while only the root PKG knows the initial root key.

KeyDer This algorithm is run by the parent of a newly joined child at time i to compute the child's private key. During a time period i, a lower-level entity (user or lower-level PKG) joins in the system at level h. Its parent at level $h - 1$ computes the entity's key $SK_{i,h}$ associated with time period i. The inputs are the parent's private key $SK_{i,h-1}$, time i, and the ID-tuple of the child.

Upd During the time period i, an entity (PKG or individual) with ID-tuple $(\mathrm{ID}_1, \ldots, \mathrm{ID}_h)$ uses $SK_{i,h}$ to compute his key $SK_{(i+1),h}$ for the next time period $i + 1$, and erases $SK_{i,h}$.

Enc A sender inputs $params$, the index i of the current time period, $M \in \mathcal{M}$ and the ID-tuple of the intended message recipient, and computes a ciphertext $C \in \mathcal{C}$.

Dec During the time period i, a user with the ID-tuple $(\mathrm{ID}_1, \ldots, \mathrm{ID}_h)$ inputs $params$, $C \in \mathcal{C}$, and its secret key $SK_{i,h}$ associated with time period i and the ID-tuple, and returns the message $M \in \mathcal{M}$.

Encryption and decryption must satisfy the standard consistency constraint, namely when $SK_{i,h}$ is the secret key generated by algorithm KeyDer for ID-tuple $(\mathrm{ID}_1, \ldots, \mathrm{ID}_h)$ and time period i, then: $\forall M \in \mathcal{M}$, decryption of the ciphertext C with $params$ and the key $SK_{i,h}$ yields the message M, where C is the result of the encryption of M under time i and $(\mathrm{ID}_1, \ldots, \mathrm{ID}_h)$.

2.2. fs-HIBE: Security

We allow an attacker to make *key derivation queries*. Also, we allow the adversary to choose the time period and the identity on which it wishes to be challenged. Notice that an adversary may *choose* the time period and the identity of its targets adaptively or non-adaptively. An adversary that chooses its targets adaptively first makes key derivation queries and decryption queries, and then chooses its targets based on the results of these queries. A nonadaptive adversary, on the other hand, chooses its targets independently from the results of the queries he makes. Security against an adaptive-chosen-target adversary, which is captured below, is the stronger notion of security than the non-adaptive one. It is also stronger than the selective-node security defined in the fs-PKE scheme by Canetti *et al.* [CHK03].

*Full-identity chosen-ciphertext security (*ind-id-cca*)* We say a fs-HIBE scheme is semantically secure against adaptive chosen ciphertext, time period, and identity attack, if no polynomial time bounded adversary $\mathcal{A}$ has a non-negligible advantage against the challenger in the following game.

Setup The challenger takes a security parameter k, and runs Setup algorithm. It gives the adversary the resulting system parameters *params*. It keeps the root secrets to itself.

Phase 1 The adversary issues queries $q_1, \ldots, q_m$, where q_i is one of the followings:[1]

1. Key derivation query $(t_i, \text{ID-tuple}_i)$: the challenger runs the KeyDer algorithm to generate the private key $SK_{(t_i,\,\text{ID-tuple}_i)}$ corresponding to $(t_i, \text{ID-tuple}_i)$, and sends $SK_{(t_i,\text{ID-tuple}_i)}$ to the adversary.
2. Decryption query $(t_i, \text{ID-tuple}_i, C_i)$: the challenger runs KeyDer algorithm to generate the private key $SK_{(t_i,\text{ID-tuple}_i)}$ corresponding to the pair $(t_i, \text{ID-tuple}_i)$, runs the Dec algorithm to decrypt C_i using $SK_{(t_i,\text{ID-tuple}_i)}$, and sends the resulting plaintext to the adversary.

These queries may be asked adaptively. Also, the queried ID-tuple$_i$ may correspond to a position at any level in the ID hierarchy, and the adversary is allowed to query for a future time and then for a past time.

Challenge Once the adversary decides that **Phase 1** is over, it outputs two equal length plaintexts $M_0, M_1 \in \mathcal{M}$, a time period t^* and an ID-tuple* on which it wishes to be challenged. The constraint is that no key derivation query has been issued for ID-tuple* or any of its ancestors for any time $t \leq t^*$.
The challenger picks a random bit $b \in \{0, 1\}$, and sets $C^* = \mathsf{Enc}(params, t^*, \text{ID-tuple}^*, M_b)$. It sends C^* as a challenge to the adversary.

Phase 2 The adversary issues more queries $q_{m+1}, \ldots, q_n$, where q_i is one of:[2]

1. Key derivation query $(t_i, \text{ID-tuple}_i)$, where the time period t_i and ID-tuple$_i$ are under the same restriction as in **Challenge**: the challenger responds as in **Phase 1**.
2. Decryption query $(t_i, \text{ID-tuple}_i, C_i) \neq (t^*, \text{ID-tuple}^*, C^*)$: the challenger responds as in **Phase 1**.

Guess The adversary outputs a guess $b' \in \{0, 1\}$. The adversary wins the game if $b = b'$. We define its advantage in attacking the scheme to be $|\Pr[\, b = b' \,] - \frac{1}{2}|$.

3. A Forward-Secure HIBE Scheme

Here, we present a forward-secure hierarchical identity-based encryption scheme. Following the presentation standard in the IBE literature [BF03,GS02], we first present a fs-HIBE with *one-way security*. One-way security is the weakest notion of security. It means that it is hard to recover a plaintext with a passive attack. A standard technique, due to Fujisaki and Okamoto [FO99], converts one-way security to chosen-ciphertext security in the random oracle model. The definition of one-way security and the Fujisaki-Okamoto conversion of the one-way secure fs-HIBE are omitted here.

[1]In the random oracle model, the adversary may also issue public key queries. Public key query $(t_i, \text{ID-tuple}_i)$: challenger runs a hash algorithm on $(t_i, \text{ID-tuple}_i)$ to obtain the public key $H(t_i \circ \text{ID-tuple}_i)$ corresponding to $(t_i, \text{ID-tuple}_i)$, where H is a random oracle.

[2]In the random oracle model, the adversary may also issue public key query. Public key query $(t_i, \text{ID-tuple}_i)$: the challenger responds as in **Phase 1**.

Our scheme, which is based on the HIBE scheme of Gentry and Silverberg [GS02] and the fs-PKE scheme of Canetti, Halevi and Katz [CHK03], overcomes the scalability and security problems that exist when naively combining the two schemes as described in Section 1.3. Next, we first give the number theoretic assumptions needed in our scheme, and then describe the algorithms in our construction.

3.1. Assumptions

The security of our fs-HIBE scheme is based on the difficulty of the bilinear Diffie-Hellman (BDH) problem [BF03]. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups of some large prime order q. We write $\mathbb{G}_1$ additively and $\mathbb{G}_2$ multiplicatively. Our schemes make use of a bilinear pairing.

Admissible Pairings Following Boneh and Franklin [BF03], we call $\hat{e}$ an admissible pairing if $\hat{e}$: $\mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is a map with the following properties:

1. Bilinear: $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1$ and all $a, b \in \mathbb{Z}$.
2. Non-degenerate: The map does not send all pairs in $\mathbb{G}_1 \times \mathbb{G}_1$ to the identity in $\mathbb{G}_2$.
3. Computable: There is an efficient algorithm to compute $\hat{e}(P, Q)$ for any $P, Q \in \mathbb{G}_1$.

We refer the readers to papers by Boneh and Franklin [BF03] for examples and discussions of groups that admit such pairings.

Bilinear Diffie-Hellman (BDH) Parameter Generator As in IBE [BF03] scheme, a randomized algorithm $\mathcal{IG}$ is a BDH parameter generator if $\mathcal{IG}$ takes a security parameter $k > 0$, runs in time polynomial in k, and outputs the description of two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of the same prime order q and the description of an admissible paring $\hat{e}$: $\mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$.

BDH Problem As in IBE [BF03] scheme, given a randomly chosen $P \in \mathbb{G}_1$, as well as aP, bP, and cP (for unknown randomly chosen $a, b, c \in \mathbb{Z}q$), compute $\hat{e}(P, P)^{abc}$.

For the BDH problem to be hard, $\mathbb{G}_1$ and $\mathbb{G}_2$ must be chosen so that there is no known algorithm for efficiently solving the Diffie-Hellman problem in either $\mathbb{G}_1$ or $\mathbb{G}_2$. Note that if the BDH problem is hard for a paring $\hat{e}$, then it follows that $\hat{e}$ is non-degenerate.

BDH Assumption As in IBE [BF03] scheme, we say a BDH parameter generator $\mathcal{IG}$ satisfies the BDH assumption if the following probability is negligible in k for all PPT algorithm $\mathcal{A}$: $\Pr\left[\mathcal{A}(\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, aP, bP, cP) = \hat{e}(P, P)^{abc}\right]$ where $(\mathbb{G}_1, \mathbb{G}_2, \hat{e}) \leftarrow \mathcal{IG}(1^k); P \leftarrow \mathbb{G}_1; a, b, c \leftarrow \mathbb{Z}q$.

3.2. fs-HIBE: Implementation

For simplicity of description, our fs-HIBE construction makes use of a version of fs-PKE scheme due to Katz [Kat02]. In Katz's scheme, time periods are associated with the *leaf* nodes of a binary tree (Rather than with all tree nodes as in the scheme by Canetti *et al.* [CHK03]. Our fs-HIBE scheme can also be realized based on the fs-PKE scheme by Canetti *et al.*, which will give faster key update time. The complexity discussion of our scheme is in Section 6). Without loss of generality, we give the root PKG ID_1, where ID_1 can just be an empty string.

Keys There are two types of keys: $sk_{w,(\mathrm{ID}_1,\ldots,\mathrm{ID}_h)}$ and $SK_{i,(\mathrm{ID}_1,\ldots,\mathrm{ID}_h)}$. The node key $sk_{w,(\mathrm{ID}_1,\ldots,\mathrm{ID}_h)}$ is the key associated with some prefix w of the bit representation of a time period i and a tuple $(\mathrm{ID}_1,\ldots,\mathrm{ID}_h)$. $SK_{i,(\mathrm{ID}_1,\ldots,\mathrm{ID}_h)}$ denotes the key associated with time i and an ID-tuple $(\mathrm{ID}_1,\ldots,\mathrm{ID}_h)$. It consists of sk keys as follows: $SK_{i,(\mathrm{ID}_1,\ldots,\mathrm{ID}_h)} = \{sk_{i,(\mathrm{ID}_1,\ldots,\mathrm{ID}_h)}, sk_{w1,(\mathrm{ID}_1,\ldots,\mathrm{ID}_h)}\colon w0 \text{ is a prefix of } i\}$. When this causes no confusion, we denote the keys as $sk_{w,h}$ and $SK_{i,h}$, respectively.

fs-HIBE Construction Let $\mathcal{IG}$ be a BDH parameter generator for which the BDH assumption holds.

Setup($1^k, N = 2^l$) The root PKG with ID_1 does the following:

1. $\mathcal{IG}$ is run to generate groups $\mathbb{G}_1, \mathbb{G}_2$ of order q and bilinear map $\hat{e}$.
2. A random generator $P \leftarrow \mathbb{G}_1$ is selected along with random $s_\epsilon \leftarrow \mathbb{Z}q$. Set $Q = s_\epsilon P$.
3. Choose a cryptographic hash function H_1: $\{0,1\}^* \to \mathbb{G}_1$. Choose a cryptographic hash function H_2: $\mathbb{G}_2 \to \{0,1\}^n$ for some n. The security analysis will treat H_1 and H_2 as random oracles [BR93]. The message space is $\mathcal{M} = \{0,1\}^n$. The ciphertext space is $\mathcal{C} = \mathbb{G}_1^{l\times h} \times \{0,1\}^n$ where h is the level of the recipient. The system parameters are *params* = $(\mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, Q, H_1, H_2)$. All operations of fs-HIBE are performed under *params*. The master key is $s_\epsilon \in \mathbb{Z}q$. The root PKG needs to generate not only the sk key associated with the current time period 0, but also the sk keys corresponding to the internal nodes on the binary tree whose bit representations are all 0 except the last bit. The sk key for time 0 is denoted as $sk_{0^l,1}$. The rest of sk values are used by the root PKG to generate keys for future time periods, and are represented as $\{sk_{1,0}, sk_{(01),1}, \ldots, sk_{(0^{l-1}1),1}\}$. These values are generated recursively as follows.

   (a) Set the secret point $S_{0,1}$ to $s_\epsilon H_1(0 \circ \mathrm{ID}_1)$, and $S_{1,1}$ to $s_\epsilon H_1(1 \circ \mathrm{ID}_1)$.
   (b) Set secret key $sk_{0,1} = (S_{0,1}, \emptyset)$ and $sk_{1,1} = (S_{1,1}, \emptyset)$. Root PKG uses $sk_{0,1}$ to recursively call algorithm CompNext (defined below) to generate its secret keys. Let $(sk_{w00,1}, sk_{w01,1}) = \mathsf{CompNext}(sk_{w0,1}, w0, \mathrm{ID}_1)$, for all $1 \leq |w0| \leq l-1$.
   (c) Set the root PKG's secret key for time period 0 as $SK_{0,1} = (sk_{0^l,1}, \{sk_{1,1}, sk_{(01),1}, \ldots, sk_{(0^{l-1}1),1}\})$, and erase all other information.

CompNext($sk_{w,h}, w, (\mathrm{ID}_1 \ldots \mathrm{ID}_h)$) This is a helper method and is called by the Setup and Upd algorithms. It takes a secret key $sk_{w,h}$, a node w, and an ID-tuple, and outputs keys $sk_{(w0),h}$, $sk_{(w1),h}$ for time nodes $w0$ and $w1$ of $(\mathrm{ID}_1 \ldots \mathrm{ID}_h)$.

1. Parse w as $w_1 \ldots w_d$, where $|w| = d$. Parse ID-tuple as $\mathrm{ID}_1, \ldots, \mathrm{ID}_h$. Parse $sk_{w,h}$ associated with time node w as $(S_{w,h}, \mathcal{Q}_{w,h})$, where $S_{w,h} \in \mathbb{G}_1$ and $\mathcal{Q}_{w,h} = \{Q_{k,j}\}$ for all $1 \leq k \leq d$ and $1 \leq j \leq h$, except for $k = 1$ and $j = 1$.
2. Choose random $s_{(d+1),j} \in \mathbb{Z}q$ for all $1 \leq j \leq h$.
3. Set $S_{(w0),h} = S_{w,h} + \sum_{j=1}^{h} s_{(d+1),j} H_1(w0 \circ \mathrm{ID}_1 \ldots \mathrm{ID}_j)$.
4. Set $S_{(w1),h} = S_{w,h} + \sum_{j=1}^{h} s_{(d+1),j} H_1(w1 \circ \mathrm{ID}_1 \ldots \mathrm{ID}_j)$.
5. Set $Q_{(d+1),j} = s_{(d+1),j} P$ for all $j \in [1,h]$.
6. Set $\mathcal{Q}_{(w0),h}$ and $\mathcal{Q}_{(w1),h}$ to be the union of $\mathcal{Q}_{w,h}$ and $Q_{(d+1),j}$ for all $1 \leq j \leq h$.

7. Output $sk_{(w0),h} = (S_{(w0),h}, \mathcal{Q}_{(w0),h})$ and $sk_{(w1),h} = (S_{(w1),h}, \mathcal{Q}_{(w1),h})$.
8. Erase $s_{(d+1),j}$ for all $1 \le j \le h$.

KeyDer($SK_{i,(h-1)}, i, (\mathrm{ID}_1 \ldots \mathrm{ID}_h)$) Let E_h be an entity that joins the hierarchy during the time period $i < N - 1$ with ID-tuple $(\mathrm{ID}_1, \ldots, \mathrm{ID}_h)$. E_h's parent generates E_h's key $SK_{i,h}$ using its key $SK_{i,(h-1)}$ as follows:

1. Parse i as $i_1 \ldots i_l$ where $l = \log_2 N$.
   Parse $SK_{i,(h-1)}$ as $(sk_{i,(h-1)}, \{sk_{(i|_{k-1}1),(h-1)}\}_{i_k=0})$.
2. For each value $sk_{w,(h-1)}$ in $SK_{i,(h-1)}$, E_h's parent does the following to generate E_h's key $sk_{w,h}$:
   (a) Parse w as $w_1 \ldots w_d$, where $d \le l$, and parse the secret key $sk_{w,(h-1)}$ as $(S_{w,(h-1)}, \mathcal{Q}_{w,(h-1)})$.
   (b) Choose random $s_{k,h} \in \mathbb{Z}q$ for all $1 \le k \le d$. Recall that $s_{k,j}$ is a shorthand for $s_{w|_k,(\mathrm{ID}_1 \ldots \mathrm{ID}_j)}$ associated with time node $w|_k$ and tuple $(\mathrm{ID}_1 \ldots \mathrm{ID}_j)$.
   (c) Set the child entity E_h's secret point
   $S_{w,h} = S_{w,(h-1)} + \sum_{k=1}^{d} s_{k,h} H_1(w|_k \circ \mathrm{ID}_1 \ldots \mathrm{ID}_h)$.
   (d) Set $Q_{k,h} = s_{k,h}P$ for all $1 \le k \le d$. Let $\mathcal{Q}_{w,h}$ be the union of $\mathcal{Q}_{w,(h-1)}$ and $Q_{k,h}$ for all $1 \le k \le d$.
   (e) Set $sk_{w,h}$ to be $(S_{w,h}, \mathcal{Q}_{w,h})$.
3. E_h's parent sets E_h's $SK_{i,h} = (sk_{i,h}, \{sk_{(i|_{k-1}1),h}\}_{i_k=0})$, and erases all other information.

Upd($SK_{i,h}, i + 1, (\mathrm{ID}_1 \ldots \mathrm{ID}_h)$) (where $i < N - 1$) At the end of time i, an entity (PKG or individual) with ID-tuple $(\mathrm{ID}_1, \ldots, \mathrm{ID}_h)$ does the following to compute its private key for time $i + 1$, as in the fs-PKE scheme [CHK03,Kat02].

1. Parse i as $i_1 \ldots i_l$, where $|i| = l$. Parse $SK_{i,h}$ as $(sk_{(i|_l),h}, \{sk_{(i|_{k-1}1),h}\}_{i_k=0})$. Erase $sk_{i|_l,h}$.
2. We distinguish two cases. If $i_l = 0$, simply output the remaining keys as the key $SK_{(i+1),h}$ for the next period for ID-tuple $(\mathrm{ID}_1, \ldots, \mathrm{ID}_h)$. Otherwise, let $\tilde{k}$ be the largest value such that $i_{\tilde{k}} = 0$ (such $\tilde{k}$ must exist since $i < N - 1$). Let $i' = i|_{\tilde{k}-1}1$. Using $sk_{i',h}$ (which is included as part of $SK_{i,h}$), recursively apply algorithm CompNext to generate keys $sk_{(i'0^d1),h}$ for all $0 \le d \le l - \tilde{k} - 1$, and $sk_{(i'0^{d-\tilde{k}},h)}$. The key $sk_{(i'0^{d-\tilde{k}},h)}$ will be used for decryption in the next time period $i + 1$; the rest of sk keys are for computing future keys. Erase $sk_{i',h}$ and output the remaining keys as $SK_{(i+1),h}$.

Enc($i, (\mathrm{ID}_1, \ldots, \mathrm{ID}_h), M$) (where $M \in \{0, 1\}^n$)

1. Parse i as $i_1 \ldots i_l$. Select random $r \leftarrow \mathbb{Z}q$.
2. Denote $P_{k,j} = H_1(i|_k \circ \mathrm{ID}_1 \ldots \mathrm{ID}_j)$ for all $1 \le k \le l$ and $1 \le j \le h$. Output $\langle i, (\mathrm{ID}_1, \ldots, \mathrm{ID}_h), C \rangle$, where $C = (rP, rP_{2,1}, \ldots, rP_{l,1}, rP_{1,2}, \ldots, rP_{l,2}, \ldots, rP_{1,h}, \ldots, rP_{l,h}, M \oplus H_2(\hat{e}(Q, H_1(i_1 \circ \mathrm{ID}_1))^r))$.

Dec($i, (\mathrm{ID}_1, \ldots, \mathrm{ID}_h), SK_{i,h}, C$)

1. Parse i as $i_1 \ldots i_l$.
   Parse $SK_{i,h}$ associated with the ID-tuple as $(sk_{i,h}, \{sk_{(i|_{k-1}1),h}\}_{i_k=0})$ and the key $sk_{i,h}$ as $(S_{i,h}, \mathcal{Q}_{i,h})$.

Parse $\mathcal{Q}_{i,h}$ as $\{Q_{k,j}\}$ for all $1 \leq k \leq l$ and $1 \leq j \leq h$, except for $k = 1$ and $j = 1$.

2. Parse C as $(U_0, U_{2,1}, \ldots, U_{l,1}, U_{1,2}, \ldots, U_{l,2}, \ldots, U_{1,h}, \ldots, U_{l,h}, V)$.
3. $M = V \oplus H_2(\frac{\hat{e}(U_0, S_{i,h})}{g})$, where g is: $\prod_{k=1}^{l} \prod_{j=2}^{h} \hat{e}(Q_{k,j}, U_{k,j}) \prod_{k=2}^{l} \hat{e}(Q_{k,1}, U_{k,1})$.

Using Fujisaki-Okamoto padding [FO99] and the help of random oracles H_3 and H_4, algorithm Enc and Dec can be converted into ones with chosen ciphertext security, as in $\mathcal{BF}$-$\mathcal{IBE}$ [BF03] and $\mathcal{GS}$-$\mathcal{HIBE}$ [GS02]. We summarize the security of our fs-HIBE scheme in Theorems VII.1 and VII.2.

Theorem VII.1 *Suppose there is a nonadaptive adversary $\mathcal{A}$ that has advantage ϵ against the one-way secure fs-HIBE scheme for some* fixed *time t and ID-tuple, and that makes $q_{H_2} > 0$ hash queries to the hash function H_2 and a finite number of key derivation queries. If the hash functions H_1, H_2 are random oracles, then there is an algorithm $\mathcal{B}$ that solves the BDH in groups generated by $\mathcal{IG}$ with advantage $(\epsilon - \frac{1}{2^n})/q_{H_2}$ and running time $\mathcal{O}(time(\mathcal{A}))$.*

Theorem VII.2 *Suppose there is an adaptive adversary $\mathcal{A}$ that has advantage ϵ against the one-way secure fs-HIBE scheme targeting some time and some ID-tuple at level h, and that makes $q_{H_2} > 0$ hash queries to the hash function H_2 and at most $q_E > 0$ key derivation queries. Let $l = \log_2 N$, where N is the total number of time periods. If the hash functions H_1, H_2 are random oracles, then there is an algorithm $\mathcal{B}$ that solves the BDH in groups generated by $\mathcal{IG}$ with advantage $(\epsilon(\frac{h+l}{e(2lq_E+h+l)})^{(h+l)/2} - \frac{1}{2^n})/q_{H_2}$ and running time $\mathcal{O}(time(\mathcal{A}))$.*

4. Application: Forward-Secure Broadcast Encryption

In this section, we show how the fs-HIBE scheme can be used to build a scalable forward-secure public-key broadcast encryption (fs-BE) scheme which is joining-time-oblivious. In what follows, N denotes the total number of time periods, $\mathcal{E}$ denotes the universe of users and $E = |\mathcal{E}|$.

4.1. fs-BE: Syntax

Forward-Secure Broadcast Encryption Scheme (fs-BE) An fs-BE scheme is specified by five poly-time algorithms Setup, KeyDer, Upd, Enc, Dec:

Setup The setup algorithm is a probabilistic algorithm run by the center to set up the parameters of the scheme. Setup takes as input a security parameter k and possibly $r_{\max}$ (where $r_{\max}$ is a revocation threshold, i.e. the maximum number of users that can be revoked). The input also includes the total number E of users in the system and the total number of time periods N. Setup generates the public key PK and the initial master secret key MSK_0.

KeyDer The key derivation algorithm is a probabilistic algorithm run by the center to compute the secret initialization data for a new user. KeyDer takes as input the master secret key MSK_t at time t, the identity u of the user and the current time period $t < N - 1$ and outputs the new secret key $USK_{t,u}$.

Upd The key update algorithm is a deterministic algorithm run by an entity (center or user) to update its own secret key of time t into a new secret key valid for the following time period $t+1$. For a user, Upd takes as input the public key PK, the identity u of a user, the current time period $t < N-1$, and the user's secret key $USK_{t,u}$, and outputs the new user's secret key $USK_{t+1,u}$. For the center, the algorithm takes as input the public key PK, the current time period $t < N$, and the key MSK_t, and outputs the secret key MSK_{t+1}.

Enc The encryption algorithm is a probabilistic algorithm that each content provider can use to encrypt messages. Enc takes as input the public key PK, a message M, the current time period t and a set $\mathcal{R}$ of revoked users (with $|\mathcal{R}| \leq r_{\max}$, if a threshold has been specified to the Setup algorithm), and returns the ciphertext C to be broadcast.

Dec The decryption algorithm is a deterministic algorithm run by each user to recover the content from the broadcast. Dec takes as input the public key PK, the identity u of a user, a time period $t < N$, the user's secret key $USK_{t,u}$ and a ciphertext C, and returns a message M.

An fs-BE scheme should satisfy the following correctness constraint: for any pair (PK, MSK_t) output by the algorithm $\mathsf{Setup}(k, r_{\max}, N, E)$, any $t < N$, any $\mathcal{R} \subseteq \mathcal{E}, (|\mathcal{R}| \leq r_{\max})$, any user $u \in \mathcal{E} \setminus \mathcal{R}$ with secret key $USK_{t,u}$ (properly generated for time period t) and any message M, it should hold that:

$$M = \mathsf{Dec}(PK, u, t, USK_{t,u}, \mathsf{Enc}(PK, M, t, \mathcal{R})) \ .$$

4.2. fs-BE: Security

In fs-BE scheme, if a user leaks his or her secret key and is not revoked by a content provider, the security of subsequent communications broadcasted by such provider is compromised. As a matter of fact, the forward security of broadcast encryption schemes guarantees that this is the *only* case where unauthorized access to the broadcast content may occur. The advantage of the adversary is not significantly improved even if she corrupts multiple users at different time periods. We formalize the security definition of fs-BE below.

Chosen-Ciphertext Security An fs-BE scheme is *forward-secure against chosen-ciphertext attack* if no polynomial time bounded adversary $\mathcal{A}$ has a non-negligible advantage against the challenger in the following game:

Setup The challenger takes security parameters $k, r_{\max}$, and runs the Setup algorithm, for the specified number of users E and time periods N. It gives the adversary the resulting system public key PK and keeps the initial master secret key MSK_0 secret to itself.

Phase 1 The adversary issues, in any adaptively-chosen order, queries $q_1, \ldots, q_m$, where q_i is one of the followings:

1. Key derivation query (u, t): the challenger runs algorithm $\mathsf{KeyDer}(MSK_t, u, t)$ to generate the private key $USK_{t,u}$ corresponding to user u at time t, and sends $USK_{t,u}$ to the adversary.

2. Decryption query (u, t, C): the challenger first runs the $\mathsf{KeyDer}(\mathit{MSK}_t, u, t)$ algorithm to recover private key $\mathit{USK}_{t,u}$ corresponding to user u at time t, and then runs decryption algorithm $\mathsf{Dec}(PK, u, t, \mathit{USK}_{t,u}, C)$ to decrypt C, and sends the resulting plaintext to the adversary.

Challenge Once the adversary decides that **Phase 1** is over, it outputs two equal-length plaintexts $M_0, M_1 \in \mathcal{M}$, and a time period t^* on which it wishes to be challenged. The challenger picks a random bit $b \in \{0, 1\}$, and set $C^* = \mathsf{Enc}(PK, M_b, t^*, \mathcal{R}_{t^*})$, where $\mathcal{R}_{t^*} = \{u \mid \mathcal{A}$ asked a key derivation query for (u, t), for some $t \leq t^*\}$. It sends C^* as a challenge to the adversary.

Phase 2 The adversary issues more queries $q_{m+1}, \ldots, q_n$, where q_i is one of:

1. Key derivation query (u, t): the challenger first checks that either $u \in \mathcal{R}_{t^*}$ or $t > t^*$ and if so, it responds as in **Phase 1**. Notice that if a bound r_{max} was specified in Setup, then the adversary is restricted to corrupt at most r_{max} distinct users via key derivation queries.
2. Decryption query (u, t, C): the challenger first checks that either $C \neq C^*$ or $u \in \mathcal{R}_{t^*}$ or $t \neq t^*$ and if so, it responds as in **Phase 1**.

Guess The adversary outputs a guess $b' \in \{0, 1\}$ and wins the game if $b = b'$. We define its advantage in attacking the scheme to be $|\Pr[b = b'] - \frac{1}{2}|$.

4.3. fs-BE: A Construction Based on fs-HIBE

Here, we show how our fs-HIBE scheme can be applied to the construction of the public-key broadcast encryption of [DF03a] to obtain a forward-secure public-key BE scheme. Dodis and Fazio [DF03a] provided a construction that extends the symmetric-key broadcast encryption scheme of Naor *et al.* [NNL01] to the public-key setting, based on any secure HIBE scheme. The construction of [DF03a] also applies to the scheme of Halevy and Shamir [HS02], that improves upon the work of [NNL01]. The symmetric-key BE scheme of Halevy and Shamir is an instance of the *Subset Cover Framework* [NNL01]. The main idea of the framework is to define a family $\mathcal{S}$ of subsets of the universe $\mathcal{E}$ of users in the system, and to associate each subset with a key, which is made available to all the users belonging to the given subset. To broadcast a message to all the subscribers except those in some set $\mathcal{R}$, a content provider first *covers* the set of privileged users using subsets from the family $\mathcal{S}$. This is done by identifying a partition of $\mathcal{E} \setminus \mathcal{R}$, where all the subsets are elements of $\mathcal{S}$. Then, the provider encrypts the message for all the subsets in that partition. To decrypt, a user $u \notin \mathcal{R}$ first identifies the subset in the partition of $\mathcal{E} \setminus \mathcal{R}$ to which he belongs, and then recovers the corresponding secret keys from his secret information.

In the public-key BE scheme [DF03a], the subsets containing a given user are organized into groups, and a special secret key, *protokey*, is associated with each of these groups. A user only needs to store these protokeys, from which he can derive the actual decryption keys corresponding to all the subsets in the group. Such an organization of the subsets of the family $\mathcal{S}$ produces a hierarchy, in which the leaves are elements of $\mathcal{S}$ and each internal node corresponds to a group of subsets. Using HIBE, a secret key can be associated with each internal node in the hierarchy, and constitutes the protokey for the group corresponding to that internal node.

In order to add forward secrecy in the public-key BE scheme, we essentially apply the fs-HIBE scheme to the above hierarchy. In fs-BE scheme, a protokey is associated with not only a node in the hierarchy, but also with a time period t. In fs-BE Setup, the center runs fs-HIBE Setup algorithm to compute its master secret $SK_{0,1}$. This key evolves with time, and is used by the center to compute protokeys for users. In fs-BE KeyDer, a user joins the broadcast at some time t, and the center uses its current master secret key $SK_{t,1}$ to derive protokeys for the user by running fs-HIBE KeyDer algorithm. The center and users evolve their secret keys with time autonomously by calling algorithm Upd of fs-HIBE. In fs-BE Enc, a content provider uses fs-HIBE Enc algorithm to encrypt the message not only with respect to the nodes in the hierarchy that represents the subsets in the partition of $\mathcal{E} \setminus \mathcal{R}$, but also to the current time t. In fs-BE Dec, the user first runs fs-HIBE KeyDer to derive the current secret keys from his protokey at time t. These secret keys are used for decryption by running fs-HIBE Dec algorithm. The detailed construction of our fs-BE scheme is omitted here. We analyze the complexity of fs-BE operations in Section 6.

5. Application: Multiple Hierarchical Identity-Based Encryption Scheme

ID-based cryptographic schemes have been used in complex access control scenarios [BHS04,TYW04]. We generalize the fs-HIBE into a collusion resistant multiple hierarchical ID-based encryption (MHIBE) scheme, where a message can be encrypted under multiple ID-tuples. The applications of MHIBE scheme include secure communications with users having multiple identities.

Motivations for MHIBE In role-based access control systems (RBAC) [SCFY96], individuals are assigned roles according to their qualifications, and access decisions are based on roles. The use of roles to control access is proven to be an effective means for streamlining the security management process [SCFY96]. Communications to a specific role may need to be protected so that messages can be read only by members of that role. This can be done using a shared key approach, which can be realized by an HIBE scheme. Members of a role are given a secret group key that is used for decrypting messages encrypted with the group public key of that role, which is an ID-tuple in HIBE. For example, the public key of the role *doctor* in the Emergency Room at a hospital is the ID-tuple (Hospital, ER, doctor), and members of the role *doctor* are given the corresponding private key in HIBE. The hierarchical structure of public keys in HIBE makes it particularly suitable for managing role communications in large organizations. This group key approach is efficient and scalable compared to encrypting the message with individual recipients' personal public keys, because a message is encrypted only once (under the public key of the role).

A user may have multiple roles. Some messages are intended to be read only by those who have multiple roles, and should not be recovered by collusions among role members. For example, the intended message recipients are those who must take on both role *doctor* in ER and role *research manager* at the affiliated medical school of the hospital. In health-care systems, medical data such as patient records are extremely sensitive, therefore, achieving this type of secure communications is important. However, the $\mathcal{GS}$-$\mathcal{HIBE}$ [GS02] scheme provides cryptographic operations only if the message is encrypted under one identity (ID-tuple). It cannot be used for communications to an *in-*

tersection of identities. Note that the Dual-Identity-Based Encryption scheme by Gentry and Silverberg [GS02] is different from what we want to achieve here. The word "dual" in their scheme [GS02] refers that the identities of both the sender and the recipient, rather than just the recipient, are required as input into the encryption and decryption algorithms.

To solve the problem of secure communications to members having multiple roles, we develop a multiple hierarchical identity-based encryption (MHIBE) scheme, where encryption is under multiple ID-tuples. In addition, it can be used for authenticating multiple hierarchical identities in the hidden credential protocol [BHS04], where the success of authentication of identities is implied if one can correctly decrypt the message encrypted with the required identities of the intended recipients. What makes the problem interesting is that the *intersection* of identities is different from the *union* of identities, which implies that a proper scheme should be collusion-resistant: secure even if adversaries with partial roles collude. In other words, it requires that compromising the private keys of individual identities does not compromise the messages encrypted with the intersection of identities. This property cannot be achieved by the broken Scheme III described in Section 1.3, where two separate HIBE schemes are used, as it is not collusion-resistant.

Next we use an example to describe the MHIBE scheme, including key acquisition, encryption, and the properties of MHIBE implementation generalized from our fs-HIBE scheme.

5.1. Identity-set and Joining-path-obliviousness

In MHIBE, we define an *identity-set* as the set of identities that a user has, each represented as an ID-tuple. For example, Bob's identity-set is $\{$(Hospital, ER, Doctor), (Hospital, School, Manager)$\}$. An *ancestor* E' of a node E has the same number of ID-tuples in its identity-set as that of E, and for each ID-tuple T in the identity-set of E, there is an ID-tuple in the identity-set of E' such that it is either the ancestor of T in HIBE or the same as T. In addition, the ancestor E' of the node E cannot be E. All ancestors of node E are capable of generating secret keys for E.

In an MHIBE scheme, Bob may obtain his key directly from either of the two ancestor entities. One is the entity whose identity-set is $\{$(Hospital, ER), (Hospital, School, Manager)$\}$. And the other has the identity-set $\{$(Hospital, ER, Doctor), (Hospital, School)$\}$. Bob's parents obtain their keys from their parents in the same way. The highest-level ancestor in this example is the hospital and has the identity-set $\{$Hospital, Hospital$\}$ (not $\{$Hospital$\}$). The root secret s_ϵ used for computing the private key for identity-set $\{$Hospital, Hospital$\}$ may be the same as the root secret used in regular HIBE scheme [GS02]. The private key is set to $s_\epsilon H_1(\text{Hospital} \circ \text{Hospital})$. Bob's key can be computed only by his ancestors in the MHIBE scheme. An MHIBE scheme needs to be *joining-path-oblivious*. This means that encryption should be oblivious of the path from which the receiver and his ancestors acquire their private keys. Having the receiver's identity-set is sufficient to encrypt a message. For example, the sender does not need to know whether Bob obtains his keys from entity $\{$(Hospital, ER), (Hospital, School, Manager)$\}$ or from entity $\{$(Hospital, ER, Doctor), (Hospital, School)$\}$.

5.2. Properties of our MHIBE Implementation

Our fs-HIBE scheme naturally gives rise to an MHIBE scheme. In fs-HIBE, a message is encrypted under both an ID-tuple and the current time. This can be viewed as the encryption under two tuples, one being the current time. Therefore, the identities in MHIBE scheme capture a broader sense of meaning. The MHIBE scheme generalized from our fs-HIBE scheme supports dynamic joins and joining-path-oblivious encryption. More importantly, it is collusion-resistant, which cannot be achieved by using multiple separate HIBE [GS02] schemes. In our MHIBE implementation, a message encrypted under $\{$(Hospital, ER, Doctor), (Hospital, School, Manager)$\}$ or $\{$(Hospital, School, Manager), (Hospital, ER, Doctor)$\}$ requires different decryption keys. We note that in this scheme, the fact that a user holds the private key corresponding to multiple identities does not imply that he or she has the private key to any subset of identities.

Our MHIBE scheme has similar goals as the pairing-based attribute-based encryption (ABE) schemes [OSW07]. In ABE, a user's private key for an application is constructed so that the key can encode expressive access control policies. While ABE can support expressive policies, a user may have to store several private keys, each for one application/policy. In comparison, MHIBE does not support general access control policies; the private keys in MHIBE are generated independent of applications or policies. Whether or not MHIBE can be realized by ABE is an interesting open question. We omit the details of MHIBE scheme (definition of security, description of scheme, and proof of security), as this is a direct generalization of fs-HIBE scheme. The complexities of various parameters of our MHIBE scheme are shown in Table VII.1 in Section 6.

6. Discussions

We analyze the complexity of our fs-HIBE scheme, the generalized MHIBE scheme, and the fs-BE scheme in Table VII.1 showing running time complexities and key sizes. Key generation time of fs-HIBE and MHIBE is the time to generate secret keys for a child node by the parent. Key generation time of fs-BE scheme is the running time of fs-BE KeyDer algorithm. In our fs-HIBE scheme, the time periods correspond to leaf nodes of a binary tree, and the key update time is $\mathcal{O}(h \log N)$, where N is the total number of time periods and h is the length of an ID-tuple. Because of the node arrangement, the key generation time and key update time of our fs-HIBE scheme grows logarithmically with the total number of time periods N. Faster key update time ($\mathcal{O}(h)$) can be achieved, if the time periods are associated with *all* the nodes of the tree in a pre-order traversal, as in the fs-PKE scheme by Canetti *et al.* [CHK03]. Because the realization of such a fs-HIBE scheme can be easily derived from the construction in Section 3.2, it is omitted here. We show the optimized running time in Table VII.1. Even dropping the joining-time-obliviousness requirement (as in the naive Scheme II of Section 1.3), our implementation cannot achieve a ciphertext with linear length $\mathcal{O}(h + \log N)$.

7. Summary

The Multiple Hierarchical Identity-Based Encryption scheme is an ID-Based encryption scheme for complex hierarchies. The generalization of a collusion-resistant MHIBE

Table VII.1. Dependency of parameters of our fs-HIBE, MHIBE, and fs-BE schemes on the total number N of time periods, the length h of an ID-tuple, the number m of ID-tuples in an identity-set in MHIBE, the total number E of fs-BE users and the number r of actual revoked users in fs-BE scheme. Key derivation time of fs-HIBE and MHIBE is the time to generate secret keys for a child node by the parent. Key derivation time of fs-BE scheme is the running time of fs-BE KeyDer algorithm.

Parameters	fs-HIBE	MHIBE	fs-BE
Key derivation time	$\mathcal{O}(h \log N)$	$\mathcal{O}(h^m)$	$\mathcal{O}(\log^3 E \log N)$
Encryption time	$\mathcal{O}(h \log N)$	$\mathcal{O}(h^m)$	$\mathcal{O}(r \log E \log N)$
Decryption time	$\mathcal{O}(h \log N)$	$\mathcal{O}(h^m)$	$\mathcal{O}(r + \log E \log N))$
Key update time	$\mathcal{O}(h)$	N/A	$\mathcal{O}(\log^3 E)$
Ciphertext length	$\mathcal{O}(h \log N)$	$\mathcal{O}(h^m)$	$\mathcal{O}(r \log E \log N)$
Public key size	$\mathcal{O}(h + \log N)$	$\mathcal{O}(hm)$	$\mathcal{O}(r \log E + \log N)$
Secret key size	$\mathcal{O}(h \log N)$	$\mathcal{O}(h^m)$	$\mathcal{O}(\log^3 E \log N)$

scheme from the Hierarchical Identity-Based Encryption scheme is significant, because MHIBE scheme conveniently lends itself to a wide range of applications that cannot be accomplished using HIBE schemes. To demonstrate this, we presented in details a forward-secure HIBE scheme and a forward-secure Broadcast Encryption scheme. We also described the application of MHIBE in the access control paradigm. The forward-secure applications derived from our MHIBE scheme are joining-time-oblivious and support dynamic joins, which make them scalable.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-119

Chapter VIII

Identity-Based Identification and Signature Schemes using Error Correcting Codes

Pierre-Louis CAYREL [a], Philippe GABORIT [a] and Marc GIRAULT [b]
[a] *Université de Limoges, France*
[b] *Orange Labs, France*

Abstract. In this chapter, we propose a survey of identity-based identification and signature schemes using error correcting codes. We describe coding theory, its application to cryptography and the different identification and signature schemes using error correcting codes with their new improvements. We also present the first (and unique up to now) identity-based scheme (provably secure) not based on number theory or generic constructions. The scheme combines two well known code-based schemes: the signature scheme of Courtois, Finiasz and Sendrier and the zero-knowledge identification scheme of Stern (which may also be used for signature). The scheme inherits some of the characteristics of the previous schemes: it has a large public key of order 1MB and requires a certain number of exchange rounds. The scheme can also be used as signature but leads to a very large signature of size 1.5 MB. Eventually, we present schemes obtained from the generic construction.

Keywords. Signature, identification, identity-based scheme, error-correcting codes, Stern, Niederreiter

While cryptography is trying to protect messages from attacks of potential opponents (passive or active), coding theory was designed originally to protect messages from imperfections of transmission systems (systems such as phone lines, communication with a satellite where imperfections can be: parasites, defects on the line, . . .).

Coding theory is hence a way to protect information against alterations to which it can be exposed on a channel. The idea is to send on the channel more data than the amount of information sent by adding some redundancy. If this redundancy is well structured, it is possible to correct errors introduced by the channel and to recover, despite the noise, all the information sent.

Many good families of error-correcting codes are known such as Reed-Solomon codes, Goppa codes, Reed-Muller codes, LDPC codes, BCH codes or turbo codes, which permit to achieve a good error correction rate, we refer to the book by McWilliams and Sloane [MS77] for more details.

In this chapter we first recall basic facts about error-correcting codes and the main schemes of code-based cryptography: the McEliece and the Niederreiter encryption schemes, the Courtois-Finiasz-Sendrier (CFS) signature scheme, the Stern zero-knowledge identification (and signature) scheme and the Kabiantiansky-Krouk-Smeets (KKS) signature scheme. We then show how these schemes can be combined and in particular the CFS signature scheme and the Stern identification and signature scheme to design an identity-based identification and signature scheme for coding theory. We also survey generic constructions for identity-based coding theory.

1. Error-Correcting Codes

1.1. Basic Notions of Coding Theory

A linear code of length n and dimension k is a linear subspace $\mathcal{C}$ of dimension k of the vector space $\mathbb{F}_q^n$ where $\mathbb{F}_q$ is the finite field with q elements. The elements of $\mathcal{C}$ are called codewords. The rate of an linear code of length n and dimension k is k/n. If $\mathcal{C}$ has minimum distance d, then $\mathcal{C}$ is a $[n, k, d]$ linear code over $\mathbb{F}_q$. The number $n - k$ is the redundancy of $\mathcal{C}$.

1.1.1. Definitions

Definition VIII.1 Let $\mathcal{C}$ be a $[n, k, d]$ linear code over $\mathbb{F}_q$ a generator matrix $\mathcal{G}$ is a $(k \times n)$ matrix whose rows form a basis of $\mathcal{C}$.

Let $\mathcal{G}$ be a binary matrix (k, n) (k rows and n columns) $\mathcal{G}$ is called a generator matrix of the code $\mathcal{C}$ if and only if, for any c belonging to $\mathcal{C}$, c is a linear combination of rows of $\mathcal{G}$.

If $\mathcal{G} = (I_k|A_{k\times(n-k)})$ where I_k is the identity matrix of dimension k and $A_{k\times(n-k)}$ some $(k, n-k)$ matrix, we say that $\mathcal{G}$ is in *systematic form.*

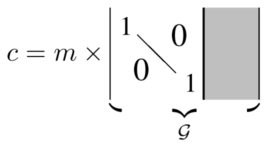


Figure VIII.1. $\mathcal{G}$: Generator matrix in systematic form.

Definition VIII.2 Let $\mathcal{C}$ be any code (not necessarily linear) in $\mathbb{F}^n$. The dual code of $\mathcal{C}$, denoted $\mathcal{C}^\perp$, is the code

$$\mathcal{C}^\perp = \{x \in \mathbb{F}^n | x * c = 0, \text{for all } c \in \mathcal{C}\},$$

where $x * c$ is the product defined by $x * c = \sum_{i=0}^{n} x_i * c_i$ where $x = (x_1, \dots, x_n)$ and $c = (c_1, \dots, c_n)$.

If $\mathcal{C}$ has a generator matrix $\mathcal{G}$ in systematic form $\mathcal{G} = (I_k | A_{k \times (n-k)})$ then a generator matrix of the dual code $\mathcal{C}^{\perp}$ is

$$\mathcal{H} = (-A^T_{(n-k) \times k} | I_{n-k}) ,$$

where A^T is the transposed matrix of A.

During the communication an error e occurs:

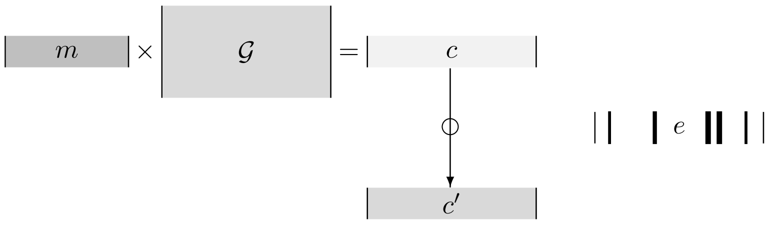


Figure VIII.2. Noise in the communication.

Definition VIII.3 We define $s = \mathcal{H}c'$ the *syndrome* of c' compared to $\mathcal{H}$. If $c' = c$ is a codeword $\mathcal{H}c = 0$.

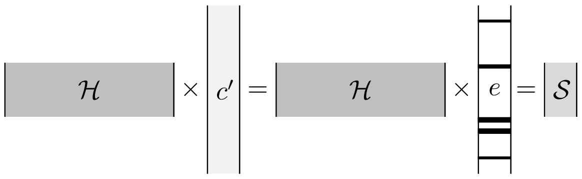


Figure VIII.3. The syndrome.

Definition VIII.4 Let $u = (u_i)_{i \in \{1,\dots,n\}}$ and $v = (v_i)_{i \in \{1,\dots,n\}}$ be two binary vectors of size n. We define:

- the *Hamming weight* of u (denoted $\mathsf{wt}(u)$): the number of ones in u. It is a norm on $\mathbb{F}_2^n$;
- the *support* of u (denoted $supp(u)$): the set of indexes i such that $u_i = 1$;
- the *Hamming distance* of u and v (denoted $d(u, v)$): the number of indexes i such that u_i is different from v_i. It is a distance on $\mathbb{F}_2^n$. Note that $d(u, v) = \mathsf{wt}(u + v)$.

The *minimum distance* (denoted by $d(\mathcal{C})$) of a code $\mathcal{C}$ is the minimum Hamming distance between any two distinct words of the code.

Definition VIII.5 Let c be a codeword, let e be an element of $\mathbb{F}_2^n$ called error, the operation which consists in recovering c and e from $c' = c + e$ is called the decoding.

Among all the codes with good decoding properties, we can cite the Goppa codes. For given integers m and t, the Goppa codes are codes of length $n = 2^m$, dimension $k \geq n - mt$ and they can correct t errors. For instance, with $m = 10, n = 1024$ and $k = 512$, it is possible to correct up to 51 errors.

1.1.2. Gilbert-Varshamov bound

The Gilbert-Varshamov bound states that there exists a binary t-error correcting code $\mathcal{C}$ of length n such that :

$$|\mathcal{C}| \geq \frac{2^n}{\sum_{i=0}^{2t} \binom{n}{i}}$$

where $|\mathcal{C}|$ represents the number of codewords in $\mathcal{C}$.

Moreover it is well known that the minimum distance of a random binary code satisfies this bound asymptotically with a very high probability. For instance if one considers a code of rate $1/2$ and length n, the Gilbert-Varshamov bound implies that the minimum distance of such a code is close to $d = 0.11n$.

1.2. Difficult Problems for Coding Theory

Definition VIII.6 (Syndrome Decoding problem (SD) [Fin04])
Input: An $(n-k) \times n$ binary matrix $\mathcal{H}$, an integer w and a syndrome $s \in {\mathbb{F}_2}^{n-k}$.
Output: An error $e \in \mathbb{F}_2^n$ such that $\mathsf{wt}(e) \leq w$ and $\mathcal{H}e^T = s$.

The safety of several cryptosystems (including McEliece [McE78] or Niederreiter [Nie86]) relies on the difficulty of the Syndrome Decoding problem. The SD problem has been proved NP-complete in [BMvT78].

The Goppa Parameterized Bounded Decoding problem (GPBD) is a special case of the SD problem. We want to fix the value of w depending on the dimensions of $\mathcal{H}$. So we are restricted to the set of the instances of SD with a special structure. This problem is also NP-complete [Fin04].

Definition VIII.7 (Goppa Parameterized Bounded Decoding problem (GPBD))
Input: An $(n-k) \times n$ binary matrix $\mathcal{H}$ (the parity check matrix of a Goppa code $\mathsf{Goppa}(n,k)$) and a syndrome $s \in {\mathbb{F}_2}^{n-k}$.
Output: An error $e \in \mathbb{F}_2^n$ such that $\mathsf{wt}(e) \leq \frac{n-k}{\log_2 n}$ and $\mathcal{H}e^T = s$.

Another problem used in code-based cryptography is the Goppa Code Distinguishing problem.

Definition VIII.8 (Goppa Code Distinguishing problem (GD) [Fin04])
Input: An $(n-k) \times n$ binary matrix $\mathcal{H}$ (the parity check matrix of a Goppa code $\mathsf{Goppa}(n,k)$) or an $(n-k) \times n$ random binary matrix $\mathcal{H}$.
Output: $b = 1$ if $\mathcal{H} \in \mathsf{Goppa}(n,k)$, $b = 0$ otherwise.

The problem GD is NP, since showing that a matrix $\mathcal{H}$ is a parity check matrix of a Goppa code can be done in polynomial time, but this problem has not been proved NP-complete.

2. Encryption Schemes Based on Error-Correcting Codes

Code-based cryptography was introduced by McEliece [McE78], one year after the introduction of public key cryptosystem RSA by Rivest Shamir and Adleman [RSA78]. In his paper, he proposed a public key encryption scheme.

The basic idea of the McEliece scheme is that Alice sends a message containing a large number of errors, that Bob can correct.

2.1. *The* McEliece *Scheme*

Setup Let $\mathcal{G}$ be a generator matrix in systematic form of a code $\mathcal{C}$ $[n, k, d]$ t-error correcting. Let S be a non-singular (i.e. invertible) binary matrix of dimension (k, k) and P be a permutation matrix of dimension (n, n). Bob's public key is:

$$\mathcal{G}' = S\mathcal{G}P \ .$$

S and P will be Bob's secret key. Bob publishes $\mathcal{G}' = S\mathcal{G}P$. Alice wants to send the message u (of k bits) to Bob.

Enc Alice computes and sends to Bob $x = u\mathcal{G}' + z$ where u is the message to send and z is a random n bits vector of weight exactly t generated by Alice.

Dec Bob receives x and computes $y = xP^{-1} = (uS)G + zP^{-1}, y$ is a word of the Goppa code of generator matrix $\mathcal{G}$ with t errors. So, Bob can determine u' by correcting the error of y. Then, he has just to compute $u = u'S^{-1}$.

The main advantage of this method is the simplicity of implementation: the only operations are bitwise operations (AND,OR,XOR,. . .). However, it requires a lot of space because the matrix is huge. For the original scheme, the author propose to use Goppa code, the parameters were $n = 1024; t = 102; k = 510$, a public key of $\approx$ 500 000 bits.

2.2. *The* Niederreiter *Scheme*

In 1986 Niederreiter proposed [Nie86] a dual public key encryption scheme.

Setup Let $\mathcal{C}$ be a linear code t-correcting of length n and of dimension k. Let $\mathcal{H}$ be a parity check matrix of $\mathcal{C}$. Let $\mathcal{H}'$ be a matrix such that:

$$\mathcal{H}' = S\mathcal{H}P \ .$$

$\mathcal{H}'$ is public and S and P are secret.

Enc For a cleartext x in the space of words of $\mathbb{F}_q^n$ with Hamming weight t, we define the ciphertext y as : $y = \mathcal{H}'x^T$.

Dec For $y = \mathcal{H}'x^T$, Bob:

1. computes $S^{-1}y \ = S^{-1}\mathcal{H}'x^T = \mathcal{H}Px^T$;
2. finds Px^T from $\mathcal{H}Px^T$ thanks to the decoding algorithm;
3. computes $P^{-1}Px^T$ and finds x by transposing the result.

2.3. Attacks

The security of the McEliece cryptosystem depends on the difficulty of the following two attacks:

(i) *Structural Attack* : recovering the secret key from the public key.
(ii) *Ciphertext-Only Attack*: recovering the plaintext from the ciphertext and the public key.

2.3.1. Structural attack

Structural attacks aim at recovering the structure of the permuted code, i.e. recovering the permutation from the code and the permuted code. The underlying problem is the equivalence of codes. This problem was considered by Sendrier who gave a nice algorithm: the Support Splitting Algorithm [Sen02]. The complexity of this algorithm is in $\mathcal{O}(2^{\text{dimension}(\mathcal{C}\cap\mathcal{C}^{\perp})})$. This means that in order to resist the attack one gets two options: either starting from a large family of codes with arbitrary small hulls (the intersection of $\mathcal{C}$ and $\mathcal{C}^{\perp}$) or starting from a small family of codes but with a large hull. For instance, the choice of Goppa codes corresponds to the first possibility.

2.3.2. Ciphertext-only attack

A first cryptanalysis using the Information-Set-Decoding was done by McEliece, then by Lee and Brickell and also by Stern and Leon and at last by Canteaut and Chabaud (see [CC98] for all references).

The Information-Set-Decoding Attack is one of the known general attacks (i.e., does not only work on particular codes) and seems to have the lowest complexity. One tries to recover the k information symbols as follows: the first step is to pick k of the n coordinates randomly, hoping that none of the k bits is erroneous. One then tries to recover the message by solving the $k \times k$ linear system (binary or over $\mathbb{F}_q$). Let $\mathcal{G}'_k$, x_k and z_k denote the k columns picked from $\mathcal{G}'$, the restriction of x and z, respectively. They have the following relationship

$$x_k = u\mathcal{G}'_k + z_k \ .$$

If $z_k = 0$ and $\mathcal{G}'_k$ is non-singular, m can be recovered by

$$m = x_k {\mathcal{G}'}_k^{-1} \ .$$

The computation cost is $\approx \frac{\binom{n-t}{k}}{\binom{n}{k}}$. The probability for an error to be decodable (see [CC98] for more details) is then $P_{dec} = \frac{\binom{n-k}{t}}{\binom{n}{t}}$, which leads with the usual binomial approximation to a probability:

$$P_{dec} = \mathcal{O}(1) \cdot 2^{-nH_2(t/n)-(1-k)H_2(t/(n-k))} \ ,$$

where $H_2(x) = -x\log_2(x) - (1-x)\log_2(1-x)$. Then the estimated work factor WF to find a word of weight t can be estimated as follows:

$$WF = \frac{P(k)}{P_{dec}},$$

where $P(k)$ corresponds to the cost of a Gaussian elimination.

All the papers which improve the complexity only impact the cost of the Gaussian elimination. In the best improvement by Canteaut and Chabaud [CC98] a good approximation of the cost besides the probability factor can be taken roughly in$(k \times \log_2 q)^2$.

Apart from these general attacks there are some attacks targeting the McEliece cryptosystem using specific codes.

3. Identification and Signature with Error-Correcting Codes

Digital signature schemes are the most important cryptographic primitive for providing message identification in a digital world. They allow a signer owning a secret key to sign a message such that anyone with access to the corresponding public key is able to verify authenticity of the message. Parallel to the efforts to build an efficient public key encryption scheme from error correcting codes, several attempts were proposed to design signature schemes based on error-correcting codes. Unfortunately, most of the proposed protocols have been proved insecure (see the survey [EOS06] and the references therein for details).

An open question was the existence of a signature scheme based on coding theory. The first code-based signature scheme is the Stern scheme in 1993. The second was proposed in 2001 by Courtois, Finiasz and Sendrier in [CFS01]. It adapts the *Full Domain Hash* approach of Bellare and Rogaway [BR93] to the Niederreiter's encryption scheme. Another approach to this problem was investigated in 1997 by Kabatianskii, Krouk and Smeets.

Compared with other public-key cryptosystems which involve modular exponentiation, these schemes have the advantage of a fast verification.

3.1. Stern *Identification and Signature Scheme*

At Crypto '93, Stern proposed a new identification and signature scheme based on coding theory [Ste94].

Stern's scheme is an interactive zero-knowledge protocol which aims at enabling any user (usually called *the prover* and denoted here by P) to be authenticated by another user (usually called *the verifier* and denoted here by S).

Let n and k be two integers such that $n \geq k$, a public $(n-k) \times n$ matrix $\mathcal{H}'$ defined over $\mathbb{F}_2$, an integer $t \leq n$. For security reasons (discussed in [Ste94]) it is recommended that t is chosen slightly below the value given by the Gilbert-Varshamov bound (see Section 1.1.2). The matrix $\mathcal{H}'$ and the weight t are protocol parameters and may be used by several (even numerous) different provers.

Each prover P receives a n-bit secret key s_P (also denoted by s if there is no ambiguity about the prover) of Hamming weight t and computes a *public key* i_P such that $i_P = \mathcal{H}' s_P^T$. This key is calculated once for all and can thus be used for several identifications. When a user P needs to prove to V that he is indeed the person associated with the public key i_P, then the two protagonists perform the following protocol where h denotes a standard hash function:

1. **Commitment Step** P randomly chooses $y \in \mathbb{F}_2^n$ and a permutation σ defined over $\mathbb{F}_2^n$. Then P sends to V the commitments c_1, c_2 and c_3 such that:

$$c_1 = h(\sigma|\mathcal{H}'y^T);\ c_2 = h(\sigma(y));\ c_3 = h(\sigma(y \oplus s))\,,$$

   where $h(a|b)$ denotes the hashed value of the concatenation of the sequences a and b.
2. **Challenge Step** V sends $b \in \{0, 1, 2\}$ to P.
3. **Answer Step** Three possibilities:
   - if $b = 0$: P reveals y and σ.
   - if $b = 1$: P reveals $(y \oplus s)$ and σ.
   - if $b = 2$: P reveals $\sigma(y)$ and $\sigma(s)$.
4. **Verification Step** Three possibilities:
   - if $b = 0$: V verifies that c_1, c_2 are correct.
   - if $b = 1$: V verifies that c_1, c_3 are correct.
   - if $b = 2$: V verifies that c_2, c_3 are correct, and that the weight of $\sigma(s)$ is t.
5. Iterate the steps 1,2,3,4 until the expected security level is reached.

Figure VIII.4. Stern's protocol.

During the fourth Step, when b equals 1, it can be noticed that $\mathcal{H}'y^T$ derives directly from $\mathcal{H}'(y \oplus s)^T$ since we have:

$$\mathcal{H}'y^T = \mathcal{H}'(y \oplus s)^T \oplus \mathcal{H}'s^T = \mathcal{H}'(y \oplus s)^T \oplus i_P\ .$$

As proved in [Ste94], the protocol is zero-knowledge and for a round iteration, the maximum probability that a dishonest person succeeds in cheating is $(2/3)$. Therefore, to get a confidence level of β, the protocol must be iterated a number of times k such that $(2/3)^k \leq 2^{-\beta}$ holds. When the number of iterations satisfies the last condition, then the security of the scheme relies on the NP-complete problem SD.

By using the so-called Fiat-Shamir paradigm [FS87], it is theoretically possible to convert Stern's protocol into a signature scheme, but the resulting signature is very large (about 140-kbit large for a security of 2^{80} binary operations). Notice however that it is large in comparison with classical signature schemes, but close to, or less than, the size of many files used in everyday life. One can also notice that the memory of common devices has considerably increased when comparing to the time (1978) when the first code-based system was proposed.

Véron's Identification Scheme In the same way the Niederreiter's scheme is the dual construction of the McEliece cryptosystem, Véron built in [Vér95] a dual construction of the Stern's scheme.

3.2. CFS *Signature Scheme*

At the difference of the RSA scheme which is naturally invertible, the McEliece or the Niederreiter schemes are not invertible, in the following sense that if one starts from a *random* element y of $\mathbb{F}_2^n$ and a code $\mathcal{C}[n,k,d]$ that we are able to decode up to $d/2$, it is almost sure that we won't be able to decode yP^{-1} into a codeword of $\mathcal{C}$. This comes from the fact that the density of the whole space which is decodable is very small. Courtois, Finiasz and Sendrier proposed in [CFS01] a practical signature scheme based on coding theory (CFS). The idea of the CFS scheme is to fix parameters $[n,k,d]$ such that the density of decodable syndromes in the Niederreiter scheme is reasonable and pick up random elements until one can be decoded.

More precisely, given M a message to sign and h a hash function onto $\{0,1\}^{n-k}$. We try to find a way to build $s \in \mathbb{F}_2^n$ of given weight t such that $h(M) = \mathcal{H}'s^T$. For $D()$ a decoding algorithm, the algorithm works as follows:

```
1. $i \leftarrow 0$
2. while $h(M|i)$ is not decodable do $i \leftarrow i+1$
3. compute $s = D(h(M|i))$
```

Figure VIII.5. The CFS signature scheme.

We get at the end an $\{s,j\}$ couple, such that $h(M \oplus j) = \mathcal{H}'s^T$. Let us notice that we can suppose that s has weight $t = [d/2]$.

The $\{s,j\}$ production process will thus be iterated, about $t!$ times before finding the correct j. But each iteration forces to compute $D(h(id_A|j))$.

The decoding of the Goppa codes consists in:

- computing a syndrome : $t^2m^2/2$ binary operations;
- computing a locator polynomial : $6t^2m$ binary operations;
- computing its roots : $2t^2m^2$ binary operations.

We thus get a total cost for the computation of Alice's private key of about: $t!t^2m^2(1/2+2+6/m)$ binary operations The cost of an attack by decoding thanks to the *split syndrome decoding* is estimated to be: $2^{tm(1/2+o(1))}$. The choice of parameters will have to be pertinent enough to conciliate cost and security. Although less crippling, some sizes have also to remain reasonable : the length of $\{s,j\}$, the cost of the verification and the size of $\mathcal{H}'$ that is for a Goppa code: $2^m tm$. The following figure sums up the different parameters:

	Operations
Comp. private key	$t!\,t^2m^2(5/2+6/m)$
Verif. cost	t^2m
Size of $\mathcal{H}'$	$2^m tm$
Attack cost	$2^{tm(1/2+o(1))}$

Following [CFS01] we can for example take $t = 9$ and $m = 16$. The cost of the signature stays then relatively reasonable for a security of about 2^{80}. The others sizes remain in that context very acceptable.

3.3. KKS *Signature Scheme*

Kabatianskii, Krouk and Smeets proposed in [KKS97] a digital signature scheme based on random linear codes. They exploited the fact that for every linear code the set of syndromes which can be decoded contains a linear subspace of relatively large dimension.

First, we suppose that $\mathcal{C}$ is defined by a random parity check $(n-k) \times n$ matrix $\mathcal{H}$. We also assume that we have a very good estimate d of its minimum distance. Next, we consider a linear code $\mathcal{U}$ of length $n' \leq n$ and dimension k defined by a $(k \times n')$ generator matrix $\mathcal{G} = [g_{i,j}]$.

We suppose that there exist two integers t_1 and t_2 such that $t_1 \leq \mathsf{wt}(u) \leq t_2$ for any non-zero codeword $u \in \mathcal{U}$.

Let J be a subset of $\{1, \ldots, n\}$ of cardinality n', $\mathcal{H}(J)$ be the sub matrix of $\mathcal{H}$ consisting of the columns h_i where $i \in J$ and define an $(n-k) \times n'$ matrix $F \overset{\text{def}}{=} \mathcal{H}(J)\mathcal{G}^T$. The application $f : \mathbb{F}_q^k \to \mathcal{M}_{n,t}$ is then defined by $f(m) = m\mathcal{G}^*$ for any $m \in GF(q)^k$ where $\mathcal{G}^* = [g^*_{i,j}]$ is the $k \times n'$ matrix with $g^*_{i,j} = g_{i,j}$ if $j \in J$ and $g^*_{i,j} = 0$ otherwise. The public application χ is $\chi(m) = Fm^T$. The scheme is described in Figure VIII.6.

Setup The signer chooses a random matrix $\mathcal{H} = [I_r|D]$ that represents the parity check matrix of a code $\mathcal{C}[n, n-r, \geq d]$. He also chooses a random generator matrix $\mathcal{G}$ that defines a code $\mathcal{U}[n', k, t_1]$ such that $\mathsf{wt}(u) \leq t_2$ for any $u \in \mathcal{U}$. He chooses a random set of size n' $J \subset \{1, \ldots, n\}$ and he forms $F = \mathcal{H}(J)\mathcal{G}^T$.

Parameters

- Private key: The set $J \subset \{1, \ldots, n\}$ and the $k \times n'$ matrix $\mathcal{G}$.
- Public key: The $r \times k$ matrix F and the $r \times n$ matrix $\mathcal{H}$.

Signature Given $m \in GF(q)^k$, the signer sends $(m, m \cdot \mathcal{G}^*)$

Verification Given (m, z), the receiver verifies that:

$$t_1 \leq \mathsf{wt}(z) \leq t_2 \text{ and } F \cdot m^T = \mathcal{H} \cdot z^T \quad .$$

Figure VIII.6. KKS signature scheme.

The main difference with the CFS signature scheme (Section 3.2) resides in the verification step where the receiver checks that:

$$t_1 \leq \mathsf{wt}(z) \leq t_2 \text{ and } F \cdot m^T = \mathcal{H} \cdot z^T \quad .$$

3.4. Recent Improvements

The three previous schemes present several drawbacks. Recent research proposed a few improvements. In this section we describe these improvements for the Stern, CFS and KKS schemes.

3.4.1. Stern's scheme: Double circulant matrices construction

Stern's scheme has rarely been used since its publication in 1993. Indeed, the scheme presents the following two drawbacks, which together makes the scheme unpractical in many applications:

1. many rounds are required (typically 28 if we want the probability of cheating less than 2^{-16}),
2. the public key element $\mathcal{H}'$ is very large (typically 150-kbit long).

The first point is inherent to interactive protocols (with small challenge length) and in some situations, it does not really constitute a drawback. For instance, if the prover entity and the verifier entity can be connected during a long period, then identification can be gradually achieved.

The second drawback has been recently considered by Gaborit and Girault in [GG07]. The idea is to replace the random matrix $\mathcal{H}'$ by the parity matrix of a particular type of random codes: *the double circulant codes.*

Let l be an integer value. A random double circulant $l \times 2l$ matrix $\mathcal{H}$ is a matrix of the form:

$$\mathcal{H} = (I|A),$$

where A is a *circulant matrix*, that is a matrix of the form:

$$A = \begin{pmatrix} a_1 & a_2 & a_3 & \cdots & a_l \\ a_l & a_1 & a_2 & \cdots & a_{l-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_2 & a_3 & a_4 & \cdots & a_1 \end{pmatrix},$$

where $(a_1, a_2, a_3, \cdots, a_l)$ is any vector in $\mathbb{F}_2^l$.

As it can be easily checked, representing $\mathcal{H}$ does not require storing all the coefficients of the matrix (as opposed to the original Stern's scheme) but requires only the l-bit vector $(a_1, a_2, a_3, \cdots, a_l)$ (which is the first row of A). Let n be equal to $2l$, the parameter sizes of the new scheme are:

- **Private Data** The secret s of bit-length n.
- **Public Data** The public syndrome i_P of size l and the first row of A of size l, which results in n bits.

It is explained in [GG07] that the behavior of this kind of matrix in terms of minimum distance is the same as that of the random matrices. Hence, it seems that the syndrome decoding Problem remains difficult in the particular case of double-circulant codes (see [GG07] for details).

As the new version of Stern's scheme involves parameters with small sizes and continues to use only elementary logical operations, it becomes highly attractive for lightweight implementations. This is especially true for environments where memory (RAM, PROM, etc.) is a rare resource [CGP08].

In order to resist decoding attacks, the authors of [GG07] suggest $l = 347$ and $t = 74$ (at least). These choices indeed lead to a security level of at least 2^{85} and to a (public and secret) key size of $n = 694$ bits.

3.4.2. CFS *Scheme: The m*CFS *signature scheme (provably secure)*

In [Dal07], a proof of security is given in the ROM. Dallot proposes to replace the counter by a random value uniformly distributed over $\{1, \ldots, 2^{n-k}\}$.

```
1. $i \xleftarrow{R} \{1, \ldots, 2^{n-k}\}$
2. while $h(M|i)$ is not decodable do $i \xleftarrow{R} \{1, \ldots, 2^{n-k}\}$
3. compute $s = D(h(M|i))$
```

Figure VIII.7. The mCFS signature scheme.

Remark VIII.9 The modified scheme uses $n - k$ bits (144 bits with the original parameters) to store the counter instead of $\log_2 t!$ bits on average in the original construction.

Dallot proved the security of the mCFS scheme. This proof relies on the security of the scheme to the *Goppa Parametrized Bounded Decoding* (GPBD) problem and the *Goppa Code Distinguishing* (GD) problem. Unfortunately the reduction is not tight.

3.4.3. KKS*: Double circulant matrices construction*

Recently in [COV07], Cayrel, Otmani and Vergnaud studied the character few-times of the KKS scheme and presented a reduction of the parameters based on a double circulant matrices construction. The new scheme proposed in [COV07] relies on the use of random double circulant matrices rather than pure random matrices.

They modified KKS schemes by replacing each random matrix by a random systematic double circulant matrix. In other words, the parity check matrix $\mathcal{H} = (I_r|D)$ is chosen such that D is double circulant. The common public key size is now $(n - r)$ bits and the personal public key still has rk bits. The random systematic matrix $\mathcal{G}$ can also be a systematic double circulant matrix. In that case the private key has $nh_2(\frac{n'}{n}) + (n' - k)$. This methods gives 176 000 bits for the personal public key, 900 bits for common public key and only 2726 bits for the private key. The signature length is about $\log_2 \binom{n}{t_2} = \log_2 \binom{2\,000}{110} = nh_2(\frac{11}{200}) = 615$ bits.

Table VIII.1. KKS parameters.

Scheme	k	n'	t_1	t_2	r	n
KKS	160	1000	90	110	1100	2000

Public key size	**176900**
Private key size	**2726**
Signature size	**615**

3.5. Summary

Stern Scheme The operations used are really simple, the identification is possible, the signature is done quickly and in a zero-knowledge context, using double circulant matrices construction by the public key is small but the signature produced remains very large.

CFS Scheme On one hand, we have seen that both key size and signing cost remain high. On the other hand the signature length and verification cost are always extremely small.

KKS Scheme The signature is quite small and the system is fast using double circulant matrices construction, the public key is small but the fact that we can just make a few times signature is a real drawback.

Table VIII.2. Parameters of code based signature schemes.

Scheme	n	Signature size	Public key
mCFS	2^{16}	144 bits	1 Mo
Stern	2^{9}	120 Kbits	347 bits
KKS	2^{11}	615 bits	22 Ko

4. Identity-Based Construction from Code-Based Signature Scheme

4.1. *The* CGG *Construction*

In [CGG07], Cayrel, Gaborit and Girault propose a new identity-based identification (and signature) scheme based on error-correcting codes. This scheme is to date the first non-generic identity-based scheme not based on number theory. The general idea of the scheme is to use the CFS scheme to solve a particular syndrome decoding problem given a value related to the identity of a person, and then use the zero-knowledge Stern identification scheme with the low weight vector associated to the given syndrome.

Here is this scheme.

Let $\mathcal{C}$ be a q-ary linear code of length n and of dimension k. Put $\mathcal{H}$ be a parity check matrix of $\mathcal{C}$. Given $\mathcal{H}' = V\mathcal{H}P$ with V invertible and P a permutation matrix. Let h a hash function with values in $\{0,1\}^{n-k}$. Let id_A be Alice's public identity. Similarly, $\mathcal{H}'$ is public. The decomposition of $\mathcal{H}'$ is, on the contrary, a secret of the authority and not of Alice.

We shall describe an identity-based identification method: Alice the prover is authenticated by Bob the verifier.

4.1.1. *Preliminary: Key setup*

Alice first has to get the private key which will then allow her to be authenticated by Bob. For that purpose, we search a way to find $s \in E_{q,n,t}$ such that $h(id_A) = \mathcal{H}'s^T$.

We use the CFS signature scheme (figure VIII.8). We get at the end a couple $\{s, j\}$,

```
1. $i \leftarrow 0$
2. while $h(id_A|i)$ is not decodable do $i \leftarrow i+1$
3. compute $s = D(h(id_A|i))$
```

Figure VIII.8. Key delivery.

such that $h(id_A|j) = \mathcal{H}'s^T$. We can note that we have s of weight t or less.

4.1.2. Identification by Bob

We use a slight derivation of Stern's protocol (Section 3.1). We suppose that A obtained a couple $\{s, j\}$ verifying $h(id_A|j) = \mathcal{H}'s^T$. (id_A, j) is A's public key. The new protocol is based on Stern's protocol but with two changes (steps 1 and 4, see figure VIII.9, we keep the same notations)First A sends j to B at the step one and second, during the fourth step, when b equals 1, it can be noticed that $\mathcal{H}'y^T$ derives directly from $\mathcal{H}'(y \oplus s)^T$ since we have:

$$\mathcal{H}'y^T = \mathcal{H}'(y \oplus s)^T \oplus \mathcal{H}'s^T = \mathcal{H}'(y \oplus s)^T \oplus h(id_A|j) \ .$$

Here is the new scheme:

1. **Commitment Step** P randomly chooses $y \in \mathbb{F}^n$ and a permutation σ defined over $\mathbb{F}_2^n$. Then P sends to V the commitments c_1, c_2, c_3 and **j** such that:

$$c_1 = h(\sigma|\mathcal{H}'y^T);\ c_2 = h(\sigma(y));\ c_3 = h(\sigma(y \oplus s))\,,$$

   where $h(a|b)$ denotes the hash of the concatenation of the sequences a and b.
2. **Challenge Step** V sends $b \in \{0, 1, 2\}$ to P.
3. **Answer Step** Three possibilities:
   - if $b = 0$: P reveals y and σ.
   - if $b = 1$: P reveals $(y \oplus s)$ and σ.
   - if $b = 2$: P reveals $\sigma(y)$ and $\sigma(s)$.
4. **Verification Step** Three possibilities:
   - if $b = 0$: V verifies that c_1, c_2 are correct.
   - if $b = 1$: V verifies that c_1, c_3 are correct.
   - if $b = 2$: V verifies that c_2, c_3 are correct, and that the weight of $\sigma(s)$ is t.
5. Iterate the steps 1,2,3,4 until the expected security level is reached.

Figure VIII.9. Stern-Niederreiter's protocol.

The knowledge of j doesn't permit to find s such that $h(id_A|j) = \mathcal{H}'s^T$. The security of this system is the same as the security of the Stern's scheme.

4.2. Parameters and Security of the Scheme

The scheme has to satisfy the following conditions:

1. the computation of $\{s, j\}$ has to be difficult without the knowledge of the description of $\mathcal{H}$,
2. the number of trials to determine the correct j must not be too important in order to reduce the cost of the computation of s.

Following [CFS01] the Goppa $[2^m, 2^m - tm, t]$ codes are a large class of codes which are compatible with condition 2. Indeed, for such a code, the proportion of the decodable syndromes is about $1/t!$ (which is a relatively good proportion). We also have to choose a relatively small t.

4.3. Practical Values

The big difference when using the parameters associated to the CFS scheme is that the code used is very long, 2^{16} against 2^9 for the basic Stern scheme, so it dramatically increases communication costs.

In the next tables we sum up for the parameters $m = 16, t = 9$ the general parameters of the $\mathcal{IBI}$ and $\mathcal{IBS}$ schemes.

Table VIII.3. Practical values for the $\mathcal{IBI}$ scheme: $m = 16, t = 9$.

Public key	Private key	Matrix size	Communication cost	Key generation
tm	tm	$2^m tm$	$\approx 2^m \times \#rounds$	
144	144	1 MB	500 Ko (58 rounds)	1 s

Table VIII.4. Practical values for the $\mathcal{IBS}$ scheme: $m = 16, t = 9$.

Public key	Private key	Matrix size	Signature length	Key generation
tm	tm	$2^m tm$	$\approx 2^m \times 150$	
144	144	1 MB	1.5 MB	1 s

Reduction of the Size of the Public Matrix Conversely to a pure signature scheme in which one wants to be able to sign quickly, in this scheme the signature is only computed once for sending it to the prover, hence the time for signing may be judged less crucial and a longer time of signature may be accepted at the cost of reducing (a little bit) the parameters of the public matrix.

In [CGGG08], Cayrel *et al.* give a proof of security in the ROM of a slightly modified version of this scheme: the mCFS-Stern scheme.

5. Generic Construction

To build an identity-based signature scheme there exists a *generic* construction due to Bellare, Namprempre and Neven.

In [BNN04], Bellare, Namprempre and Neven present a construction of an identity-based signature scheme from two standard (possibly equal) signature schemes. See Chapter III.

To apply this construction to the code-based identification scheme we can check that the code-based identification and signature schemes are convertible.

Their construction requires two signature schemes, we use:

1. mCFS-mCFS,
2. Stern-Stern,
3. KKS-KKS.

Here are the results:

	mCFS-Stern	(1)	(2)	(3)
Signature length	1,5 MB	$\approx$ 1 MB	48 Ko	22 Ko
Signature cost	1.4 $\times 10^9$ op.	2.2 $\times 10^{10}$ op.	2.2 $\times 10^7$ op.	1.3 $\times 10^9$ op.

6. Summary

In this chapter we present a survey of identity-based identification and signature schemes for coding theory. We showed that besides standard generic construction with certificates there exist non-generic constructions without certificates which combine known coding protocols. Eventually the results obtained show that in term of performance, code-based schemes have larger size of key, or signature lengths but that in term of number operations needed they could outperform number theory based schemes (especially when combining two Stern schemes). The large size of the signature make these schemes difficult to use in practice but still they represent an alternative to number theory based schemes in case a quantum computer may exist. Another question is the possible existence of an identity-based encryption scheme based on coding theory which remains unknown.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The author and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-135

Chapter IX

Certificateless Encryption

Sherman S.M. CHOW
New York University, USA

Abstract. Certificateless cryptography combines the best of traditional public-key cryptography and the ID-based paradigm. There is no need to get and verify certificates, and the secret keys are not under the key generation center (KGC)'s escrow.

This chapter focuses on giving a complete survey of secure certificateless encryption (CLE) schemes in the literature. In particular, we discuss the techniques behind addressing special security concerns in certificateless paradigm, which include key replacement attack, decryption capabilities for arbitrary public keys (as considered in strongly-secure CLE), partial decryption capabilities (as considered in security-mediated certificateless encryption, or SMCLE) and malicious KGC attack. We also examine some less obvious efficiency issues, in which we compare specific CLE constructions with the normal public key encryption approach.

1. Public-Key, Identity-Based and Certificateless Encryption

Encryption in identity-based (ID-based) cryptography avoids the tiresome certificates but makes the private key generation a privileged operation which only a trusted authority can perform. Such key escrow is desirable in some applications, but in many contexts, it is completely unacceptable. While escrow seems to be unavoidable in ID-based cryptography, it does not rule out the possibility of having a public key cryptosystem which is free from escrow *and* the hassle of public key infrastructure (PKI). In a public key cryptosystem, only the authenticated one is able to complete the cryptographic task. For example, only the intended recipient of a ciphertext can decrypt and read the plaintext message. The escrow problem of ID-based paradigm is caused by the fact that the private key generator can always generate the private key corresponding to a particular identity string, which is *sufficient* to do the decryption. On the other hand, authentication is done before one can obtain such a private key. As long as such a private key is *necessary* in decryption, authentication has been implicitly enforced and there is no need to resort to special means like a certificate to prove that it has occurred. These observations lead to a possible approach to circumvent the escrow problem without using certificate, which is introducing an extra component making the cryptographic task requires not only the private key from the ID-based cryptosystem, but also something out of the knowledge of the private key generator. This is the idea of *certificateless (public key) cryptography*.

1.1. High-Level Description of Certificateless Encryption

The focus of this chapter is on certificateless encryption. Certificateless encryption (CLE) combines ID-based encryption (IBE) with public-key encryption (PKE). Since CLE provides the functionalities of both IBE and PKE, it requires the existence of a trusted third party, called key generation center (KGC), to generate master secret and the corresponding master public key, and make this public key along with other system parameters publicly available (which is similar to IBE). A user in CLE is required to generate a public key (which is not necessary in IBE), which means the encryption cannot be done with respect to the identity alone (i.e. the encryption is not ID-based). As a result, a sender needs to fetch a public key of the recipient. However, unlike PKE, no certificates are needed.

To recapitulate, the encryption algorithm requires the recipient's identity and the public key generated by the recipient as inputs to produce a ciphertext. Correspondingly, to decrypt a ciphertext, the recipient requires both the *partial private key* (given by the KGC) corresponding to his/her identity and the *secret value* corresponding to the public key distributed to the sender.

1.2. Complexities of Certificate

Generally, the complications involved with certificates include the following.

1. Issuing: A certificate contains a digital signature created by the certificate authority (CA) certifying the binding between the identity and the public key. The CA needs to authenticate the user before issuing the certificate.
2. Storage: A repository of certificates is usually maintained by the CA.
3. Transfer: In addition to the public key, the corresponding certificate should be retrieved by the message sender for encryption.
4. Verification: The validity of the retrieved certificate should be checked, i.e. the signature by CA should be verified.
5. Revocation: If the user's private key is compromised, the corresponding certificate should be revoked.

In certificateless cryptography, the KGC still needs to authenticate and issue partial private keys. Possibly, the KGC maintains a repository of public keys for the convenience of retrieval. The real benefits it brought lie in *transfer* and *verification*. Message senders are *not required to get the certificates*; all they need is the authenticated KGC public key. Verification of certificate is not needed since it does not exist at all. Finally, certificateless cryptography by itself has not solved the problem of revocation. A more general notion, security-mediated certificateless cryptography [CBG06], is proposed to address this concern.

1.3. Development of Certificateless Encryption Schemes

In 2003, Al-Riyami and Paterson [ARP03] proposed the notion of certificateless cryptography, with constructions of certificateless encryption, its hierarchical variant, signature and key agreement protocol. A security model for the encryption case is developed and their encryption scheme is proven secure in the random oracle model (ROM), which models the hash function by the random oracle (RO).

Since this proposal, a large body of work has been devoted to creating better CLE schemes in terms of efficiency, formal security assurance and ability to withstand special attack. These extensions can be partitioned in two types: (1) specific constructions, which often borrow ideas from existing IBE schemes, and (2) generic constructions from weaker primitives, again often including IBE schemes. This section gives an overview of development of CLE schemes in these five years.

1.3.1. Concrete constructions

This first CLE scheme by Al-Riyami and Paterson ($\mathcal{AP}03$) [ARP03] is secure against a strong adversary, which can replace the user's public key (since there is no certificate to show which is the "authentic" one), and ask for the decryption of the ciphertext of a user even the user's public key is maliciously replaced. CLE schemes secure in this sense are called *strongly-secure*. A drawback of their scheme is that it requires the sender to check if the two elements in a public key possess a special relationship with reference to the master public key of the KGC. Two years later, they proposed a more efficient scheme [ARP05] using a single element public key, and the only validity check is a simple group membership test. Besides, it is possible for a single user public key to be used across different CLE systems.

Unfortunately, [ARP05] was broken by Libert and Quisquater [LQ06]. In 2006, Chow, Boyd and González Nieto [CBG06] proposed a scheme ($\mathcal{CBG}06c$) which also has the simpler key construction, but not vulnerable to a similar attack.

The constructions of $\mathcal{AP}03$ and $\mathcal{CBG}06c$ stemmed from the IBE scheme by Boneh and Franklin ($\mathcal{BF}$-$\mathcal{IBE}$) [BF03] (the full-domain hash approach). Encryption requires computation of pairing and a special hash function mapping strings onto a cyclic group, which are rather computationally expensive (but they can be pre-computed for each identity). Based on the IBE scheme by Sakai and Kasahara ($\mathcal{SK}$-$\mathcal{IBE}$) [SK03] (the exponent-inversion approach), Libert and Quisquater [LQ06] proposed a CLE scheme ($\mathcal{LQ}06c$) in 2006. Inherited from $\mathcal{SK}$-$\mathcal{IBE}$, encryption is free from special hash and pairing. Decryption still requires a single pairing. Cheng *et al.* [CCLC07] adopted another IBE by Boneh and Boyen [BB04a] (the commutative-blinding approach) to give an efficient CLE scheme.

These CLE schemes require the use of pairing since they have an "ID-based flavor" – a public key can be defined before knowing the partial private key. This is a distinctive feature that makes CLE a possible candidate to support interesting applications like timed-released encryption (e.g. see [CRR08]) and cryptographic workflow (e.g. see [AR04]). In an attempt to construct a pairing-free CLE scheme, instead of building an IBE scheme without pairing, Baek, Safavi-Naini and Susilo [BSNS05] chose to relax the CLE model. Their formulation requires a user to obtain a partial private key from the KGC before a public key can be computed (and published). Even without predefined key, "certificateless" property is not affected, but the restricted notion is less distinct from the self-certified key idea of Girault back in 1991 [Gir91]. However, their pairing-free CLE scheme ($\mathcal{BSS}05$) comes with formal analysis in a slightly weaker security model than [ARP03], which the previous schemes using self-certified key are lacking.

Unfortunately, the security argument in $\mathcal{BSS}05$ was later found to be flawed. In 2007, Lai and Kou [LK07] gave another scheme which is basically the same as $\mathcal{BSS}05$ except key-binding in [ARP03] is used, where the KGC applies the master secret key on both the identity and the public key to generate a partial private key. They assert the security of their schemes with theorems but not complete proofs. It is unclear how their scheme can avoid committing the same flaw as the proof in [BSNS05]. Fortunately, Sun, Zhang and Baek [SZB07] extended $\mathcal{BSS05}$ to give a new scheme ($\mathcal{SZB07}$) and fixed this problem.

The security proofs of all the above schemes are in the ROM. Various schemes without random oracles are proposed but they assume different adversary models (to be discussed in Section 2). Liu, Au and Susilo [LAS07] gave a CLE scheme by extending an IBE scheme in the standard model by Waters [Wat05] ($\mathcal{Waters\text{-}IBE}$). However, the scheme is not strongly-secure and its efficiency is comparable to a prior generic construction [CBG06] in the standard model. In response to these shortcomings, Park *et al.* proposed a strongly-secure CLE scheme [PCHL07] based on an IBE scheme in the standard model proposed by Gentry [Gen06]. Yet, their scheme is only proven in a weaker selective-ID model. Dent, Libert and Paterson [DLP08] proposed a concrete strongly-secure CLE scheme ($\mathcal{DLP08c}$) in the usual adaptive-ID model, based on $\mathcal{Waters\text{-}IBE}$ and a technique to obtain chosen-ciphertext security by Boyen, Mei and Waters [BMW05]. Their scheme has been extended by Chow, Roth, and Rieffel [CRR08] into a hierarchical CLE scheme ($\mathcal{CRR08}$) with immediate revocation and application of timed-release encryption in mind. It is also the first formal study of hierarchical CLE. Another extension is made by Hwang, Liu and Chow [HLC08] for security against malicious KGC attack.

1.3.2. Generic constructions

Following the intuition of how key escrow in IBE can be avoided, various generic CLE constructions combining IBE and PKE in a black-box manner have been proposed as early as in 2004. Three ways of composition are given in Al-Riyami's thesis [AR04]. In parallel composition, the message M is first split into two shares s_1 and s_2 such that $s_1 \oplus s_2 = M$; then s_1 is encrypted by IBE and s_2 is encrypted by PKE. For sequential compositions, the message is first encrypted by PKE and the resulting ciphertext is further encrypted by IBE, or in a reversed order.

Naïve compositions of multiple encryptions are insecure against chosen-ciphertext attack (CCA) when one of the secret keys involved is leaked [DK05]. Along this line, Libert and Quisquater [LQ06] showed all these compositions are CCA-insecure. Nevertheless, these compositions are secure against chosen-plaintext attack (CPA) if the underlying schemes are also CPA-secure. Libert and Quisquater [LQ06] thus brought the concept of Fujisaki-Okamoto transformation (FO transform) [FO00], which builds CCA-secure PKE out of a CPA-secure PKE, into CLE paradigm. They showed that a certificateless version of FO transform ($\mathcal{LQ06g}$, 'g' for generic) can convert a CPA-secure CLE into a CCA-secure CLE which is strongly-secure. Cheng *et al.* [CCLC07] adopted another transformation, also due to Fujisaki and Okamoto [FO99], to give a similar result based on one-way secure CLE. Consequently, all these compositions can be extended to achieve CCA security, albeit under the ROM. We remark that these studies generalize the result in [ARP03,CBG06] where FO transform [FO99,FO00] is applied.

Another approach to strengthen these compositions is to make them a secure multiple encryption. In 2005, Dodis and Katz [DK05] proposed two ways to combine multiple CCA-secure encryption instances into a CCA-secure multiple encryption. The first way utilizes a one-time signature and the second way uses a message authentication code (MAC) to replace one-time signature. The latter is more efficient yet the security guarantee is weaker. The first paper that suggested using Dodis-Katz technique to build generic CLE scheme in the standard model is by Chow, Boyd and González Nieto [CBG06] in 2006.[1] The major objective of their scheme ($\mathcal{CBG}06g$) is to solve the revocation problem in the CLE paradigm. A security-mediator (SEM) is introduced such that it partially decrypts ciphertexts for non-revoked users. Libert and Quisquater [LQ06] independently pointed out the possibility of using Dodis-Katz technique to build CLE (without concerning partial decryption in [CBG06]) but remarked that it seems difficult to get a strongly-secure CLE out of it. In the next year, Huang and Wong [HW07a] presented a sequential variant of $\mathcal{CBG}06g$. They also briefly discussed a variant using MAC instead of one-time signature, as suggested by Dodis and Katz [DK05]. Finally, Dent provided a detailed proof of Dodis-Katz approach of building CLE, with stronger security guarantees compared with [HW07a].

Theories developed about PKE also help the design of generic CLE schemes. Cramer and Shoup [CS03] showed a hybrid approach of constructing CCA-secure PKE by composing a key encapsulation mechanism (KEM) with a data encapsulation mechanism (DEM). Then, a generic certificateless KEM based on the parallel composition is proposed and proved secure in the ROM by Bentahar *et al.* [BFMLS08]. Furthermore, they showed that combining certificateless KEM and a standard DEM results in secure CLE, yet not strongly-secure. Abe *et al.* [AGKS05] introduced tag-KEM, presented several constructions of CCA-secure tag-KEM, and a generic construction of hybrid PKE. Subsequently, Huang and Wong [HW07b] mirrored their work. They extended certificateless KEM to certificateless tag-KEM, and proposed constructions for certificateless tag-KEM.

Finally, extending the CCA technique of Sahai [Sah99], Dent, Libert and Paterson [DLP08] proposed a generic strongly-secure CLE scheme from a CPA-secure PKE, a CPA-secure CLE and non-interactive zero-knowledge proof system.

2. Preliminaries

2.1. Framework

Al-Riyami and Paterson [ARP03] define a CLE as a septuple of algorithms:

- Setup (run by the KGC) is a probabilistic algorithm which takes a security parameter 1^k, outputs a master secret key msk, the corresponding master public key mpk, and the global parameters pub, which includes k, a secret value space and a message space in particular. For brevity, we assume all other algorithms take mpk and pub implicitly as an input.
- KeyDer (run by the KGC) is a possibly probabilistic algorithm which takes msk and an identity $id \in \{0,1\}^*$, outputs a partial private key D_{id}, which is distributed to the user of identity id in a secure channel.
- Set-Secret-Value (run by a user) is a probabilistic algorithm which takes an identity id, outputs a secret value sk for that identity.

[1]Dodis and Katz suggested that their technique can combine IBE and PKE into a certificate-based encryption (CBE) without random oracles. CBE, proposed by Gentry [Gen03], is similar to CLE. In general, CLE is more difficult to design due to its flexibilities in partial private key generation and public key replacement [ARP05].

- Set-Private-Key (by a user) is a possibly probabilistic algorithm taking a partial private key D_{id} and a secret value sk, outputs a full private key S_{id}.
- Set-Public-Key (by a user) is a (usually) deterministic algorithm which takes sk and outputs the corresponding public key pk to be distributed.
- Enc (by a sender) is a probabilistic algorithm which takes a message M, the receiver's identity id and public key pk; returns a ciphertext C or $\perp$.
- Dec is a (usually) deterministic algorithm which takes a ciphertext C and the full private key S_{id} as input, returns a message M or $\perp$.

The symbol $\perp$ denotes invalid input, e.g. invalid public key or ciphertext.

The above formulation can be simplified, where the Set-Secret-Value and the Set-Public-Key algorithms is replaced with a single SetUK algorithm:

- SetUK is a probabilistic algorithm which takes an identity id, outputs a secret value sk and the corresponding public key pk.

This is shown to be an equivalent method for constructing public keys [Den08, HWZD07], and has been used subsequently [ACL$^+$07,CRR08,HW07a,HW07b,LAS07]. The input id is even removed in [CRR08].

It has also been suggested in [HWZD07] that the introduction of SetUK algorithm removes the need for Set-Private-Key as well if Dec is defined as taking both the partial private key D_{id} and the secret value sk as input. This modification preserves the functionalities provided by the original framework, but the associated security model should be changed [Den08] (see Section 2.2). In a general model formulated by Chow, Roth, and Rieffel [CRR08], this modification is shown to fit with other applications of CLE. Here we adopt these simplifications.

2.2. Security Model

The confidentiality of CLE, indistinguishability against adaptive chosen-ciphertext attack, is defined based on two similar games, with the following framework.

Setup The challenger C executes Setup to get $\langle msk, mpk, pub \rangle$.

Phase 1 The adversary A is given mpk, pub and possibly some extra information aux. A is allowed to adaptively make queries to a number of oracles.

Challenge Phase A terminates Phase 1 by returning an identity id^* and two messages M_0^*, M_1^*. C picks a random bit b, gives A the challenge ciphertext $C^* = \mathsf{Enc}(M_b^*, id^*, pk^*)$, where pk^* is the public key associated with id^*.

Phase 2 A can continue to make oracle queries, subjected to restrictions.

Guess When A has finished with Phase 2, A must output a guess bit b'. A wins the game if $b' = b$ and A's advantage is defined as $2 \times |\Pr[b' = b] - 1/2|$.

The available oracles to A, with expected input and output, are as follows.

- **Request Public Key** On input an identity id, A obtains the public key pk currently associated with id. If there is no such key, C executes SetUK to generate the public key first and associate it to id before returns it to A.
- **Replace Public Key** On input an identity id and a valid public key pk, the public key associated with id is replaced by this new one.

- **Extract Partial Private Key** On input an identity id, A gets $\mathsf{KeyDer}(msk, id)$.
- **Extract Full Private Key** On input an identity id, A obtains the private key S_{id}. This query is disallowed if the public key has been replaced by A.
- **Weak SV Decrypt** On input a ciphertext C, an identity id and a secret value sk, C retrieves D_{id} and returns A with $\mathsf{Dec}(C, D_{id}, sk)$.

Replace Public Key is something special in CLE. This models the situation that nothing is assuring the binding between an identity and a public key, so a sender may be fooled to use a false public key given by an adversary.

It is possible to consider a stronger adversary with the following oracles.

- **Extract Secret Value** On input an identity id, A obtains the secret value sk associated with id, if the public key has not been replaced by A.
- **Strong Decrypt** On input a ciphertext C and an identity id, A obtains a correct decryption of C, even the public key of id has already been replaced.

Extract Secret Value oracle models a stronger adversary, since one can derive the full private key from the secret value and the partial private key, but not necessarily vice versa. For a simplified framework which has no concept of full private key, the absence of this oracle means the compromise of the user generated secret value is not modeled. It is noted in [Den08] that only schemes with deterministic Set-Public-Key (i.e. only one public key for a given secret value) can be secure if **Extract Secret Value** and **Strong Decrypt** are provided simultaneously.

Strong Decrypt is a distinctive and controversial feature of CLE. Many CLE schemes (including *AP03*, *CBG06c*, *CRR08*, *DLP08c*, *DLP08g*, *LQ06c*, *LQ06g*) are strongly-secure. While many people question whether this models any realistic attack, it is prudent to use a strongly-secure scheme when a high level of security is required as it is hard to ensure that there is absolutely no weird but real scenarios that a weaker model may fail to cover. Moreover, there may be some innovative applications of CLE in future which makes this oracle modeling a realistic attack. On the other hand, weakly-secure schemes are often more efficient in general.

Now we are ready to see the two types of adversaries in CLE paradigm.

1. **Type-I adversaries** model the collusion of insiders to the system (except the legitimate receiver) which do not have access to the master secret, but are allowed to choose any public key to be used for the challenge ciphertext.
2. **Type-II adversaries** model an honest-but-curious KGC who has access to the master secret, but can only use a registered public key for the challenge.

Below definitions are given for the simplified model without full private key.

Definition IX.1 A certificateless encryption scheme is said (t, q_E, q_D, ϵ) ind-id-cca secure against a Strong Type-I adversary if A's advantage is less than ϵ for all t-time adversaries A making at most q_E extraction queries and q_D decryption queries, subjects to the following constraints:

1. $aux = \emptyset$, i.e. no auxiliary information is given to the adversary.
2. No **Extract(id^*)** query throughout the game, where id^* is the target identity chosen by A.
3. No **Strong Decrypt(C^*, id^*)** query after C^* is defined.

The above definition for Type-I adversary, which is being generalized in [CRR08] recently, is weaker than the original definition given by Al-Riyami and Paterson [ARP03], where **Extract**$(\boldsymbol{id}^*)$ is allowed if **Replace Public Key**$(\boldsymbol{id}^*)$ query has not been made in Phase 1 [Den08, Section 2.3.5]. However, the "missing part" will not be omitted from the security requirement since it is unreasonable to define a CLE without considering Type-II adversary. To see how a Type-II adversary can do the same thing, i.e. **Extract**$(\boldsymbol{id}^*)$ first and **Replace Public Key**$(\boldsymbol{id}^*)$ in Phase 2, note that the only effect of making **Replace Public Key**$(\boldsymbol{id}^*)$ query in Phase 2 is to decrypt ciphertexts corresponding to the new public key. For weak security, this replacement is of no use (since no decryption related to the new public key of id^* can be done). For strong security, this adversarial behavior can be simulated by a Type-II adversary which computes the partial private key of id^* itself and issues queries to **Strong Decryption**. Indeed, this simplification has already been justified and adopted [HWZD07, Section 2.3].

For weakly-secure schemes (e.g. [LAS07]), only **Weak SV Decrypt** oracle is available but not **Strong Decrypt** oracle. The restriction 3 should be changed to:

3. *No* **Weak SV Decrypt**$(\boldsymbol{C}^*, \boldsymbol{id}^*, \boldsymbol{sk}^*)$ *query if* sk^* *is the response of the* **Extract Secret Value**$(\boldsymbol{id}^*)$ *query issued by* A.

The above definition for Weak Type-I adversary is similar to "Weak Type-Ia" in [Den08]. Type-Ib and Type-Ic in [Den08] are too weak and thus omitted.

Definition IX.2 A certificateless encryption scheme is said (t, q_E, q_D, ϵ) ind-id-cca secure against a Strong Type-II adversary if A's advantage is less than ϵ for all t-time adversary A making at most q_K public key queries, q_E extraction queries and q_D decryption queries, subjects to the following conditions:

1. $aux = msk$, i.e. the master secret key is given to the adversary.
2. No **Extract Secret Value**$(\boldsymbol{id}^*)$ query throughout the game.
3. No **Replace Public Key**$(\boldsymbol{id}^*, \boldsymbol{pk})$ before C^* is defined, i.e. the public key associated with the challenge identity cannot be replaced until Phase 2.
4. No **Strong Decrypt**$(\boldsymbol{C}^*, \boldsymbol{id}^*)$ until **Replace Public Key**$(\boldsymbol{id}^*, \boldsymbol{pk})$ has been made where $pk \neq pk^*$.

The definition we give here for Type-II adversary is stronger than the original one by Al-Riyami and Paterson [ARP03], where neither **Strong Decrypt** nor **Weak SV Decrypt** is allowed (the adversary can only query a decryption oracle which always uses the original secret value/private key). It has been shown [Den08] that their definition of Weak Type-II security [ARP03] can be achieved by PKE alone. Strong Type-II adversary has been considered by both [CRR08] and [Den08].

Finally, we remark that the above CLE definition is not general enough to cover applications of CLE in which an identity corresponds to a policy governing the decryption, which is "shared" among multiple users and is never bound to a single public key. The general CLE model recently proposed by Chow, Roth, and Rieffel [CRR08] removes **Replace Public Key** oracles and the concept of "user = identity". Nevertheless, implicit public key replacement can still be made via the interfaces of the other oracles. Detailed justifications can be found in [CRR08].

This concludes our discussion of the basic CLE security model. We will cover security-mediated CLE, malicious KGC attack and denial-of-decryption attack in later sections. For more discussion on the differences among different variations of models and their practical significances, we refer the reader to [Den08].

3. Strongly-Secure Certificateless Encryption

We start our review with the techniques for constructing strongly-secure CLE. The first scheme is $\mathcal{AP}03$ [ARP03], which is based on $\mathcal{BF}$-$\mathcal{IBE}$. We focus on the design of the encryption mechanism. A detailed description on the ID-based component can be found in other chapters. The definitions of the hash functions should be clear from the context.

- Setup: $msk = s \in_R \mathbb{Z}_p$, $mpk = P_0 = P^s$, $pub = \{P, \mathrm{H}_1(\cdot), \mathrm{H}_2(\cdot), \mathrm{H}_3(\cdot,\cdot), \mathrm{H}_4(\cdot)\}$.
- KeyDer(s, id): $D_{id} = \mathrm{H}_1(id)^s \in \mathbb{G}$.
- SetUK: $sk \in_R \mathbb{Z}_p$, $pk = \langle X, Y\rangle = \langle P^{sk}, {P_0}^{sk}\rangle$.
- Enc$(M, id, \langle X, Y\rangle)$:
  1. The validity of the public key pk is firstly checked. It is considered as valid if $\hat{e}(P, Y) = \hat{e}(P_0, X)$; otherwise, $\perp$ is returned.
  2. Randomly pick $\rho \in_R \mathbb{Z}_p$ and compute $r = \mathrm{H}_3(\rho, M) \in \mathbb{Z}_p^*$.
  3. Output $C = \langle C_1, C_2, C_3\rangle = \langle P^r, \rho \oplus \mathrm{H}_2(\hat{e}(Y, \mathrm{H}_1(id))^r), M \oplus \mathrm{H}_4(\rho)\rangle$.
- Dec$(\langle C_1, C_2, C_3\rangle, D_{id}, sk)$:
  1. Recover $Z = \hat{e}(C_1, D_{id}^{sk})$, retrieve $\rho' = C_2 \oplus \mathrm{H}_2(Z)$.
  2. Recover $M' = C_3 \oplus \mathrm{H}_4(\rho')$, return M' if $C_1 = P^{\mathrm{H}_3(\rho', M')}$, $\perp$ otherwise.

Enc is almost identical as that of $\mathcal{BF}$-$\mathcal{IBE}$ except $\hat{e}(P_0, \mathrm{H}_1(id))^r$ is now changed to $\hat{e}(Y, \mathrm{H}_1(id))^r$. With only P^r in the ciphertext, one needs to recover this term by $\hat{e}(P^r, \mathrm{H}_1(id)^s)^{sk}$, which is not computable if either sk or $\mathrm{H}_1(id)^s$ is unknown. The validity checking in Enc ensures that Y is in a "correct" form such that both s (the master secret) and sk (the user secret value) are included.

The crux to get strong security is the use of H_3 and H_4 in FO transformation [FO99] employed by the scheme. In the ROM, the adversary must have queried (either implicitly or explicitly) H_3 to learn the value of $r = \mathrm{H}_3(\rho, M)$ with non-negligible probability (and a similar argument holds for H_4). If the simulator cannot find a pair of (ρ', M') in the H_3 input list such that all of $C_1' \overset{?}{=} P^{\mathrm{H}_3(\rho', M')}$, $C_2' \overset{?}{=} \rho' \oplus \mathrm{H}_2(\hat{e}(Y, \mathrm{H}_1(id))^{\mathrm{H}_3(\rho', M')})$ and $C_3' \overset{?}{=} M' \oplus \mathrm{H}_4(\rho')$ hold. The challenger can be pretty sure that decryption query of ciphertext $\langle C_1', C_2', C_3'\rangle$ under id and Y will return $\perp$, as Dec returns $\perp$ if the value r in the ciphertext does not match with (ρ', M'). In other words, the simulator either can find M' encrypted inside the given ciphertext by finding the ROs' input lists, or can reject the given ciphertext with negligible probability of error (rejecting a valid ciphertext). The above procedure is known as knowledge extractor in [FO99].

Removing Full-Domain Hash $\mathcal{LQ}06c$ [LQ06] uses $\mathcal{SK}$-$\mathcal{IBE}$ instead of $\mathcal{BF}$-$\mathcal{IBE}$.

- Setup: Same as $\mathcal{AP}03$, but include $\hat{g} = \hat{e}(P, P)$ and new $\mathrm{H}_1, \mathrm{H}_2, \mathrm{H}_3$.
- KeyDer(s, id): $D_{id} = P^{1/(s + \mathrm{H}_1(id))} \in \mathbb{G}$, i.e. $\mathrm{H}_1 : \{0,1\}^* \to \mathbb{Z}_p$.
- SetUK: $sk \in_R \mathbb{Z}_p$, $pk = X = \hat{g}^{sk}$.
- Enc(M, id, X): Pick $\rho \in_R \mathbb{Z}_p$, compute $r = \mathrm{H}_3(\rho, M, X, id) \in \mathbb{Z}_p^*$, and output $C = \langle C_1, C_2\rangle = \langle (P^{\mathrm{H}_1(id)} \cdot P_0)^r, (M||\rho) \oplus \mathrm{H}_2(\hat{g}^r, X^r)\rangle$.

- $\mathsf{Dec}(\langle C_1, C_2\rangle, D_{id}, sk)$:
  1. Recover $Z = \hat{e}(C_1, D_{id})$, retrieve $(M' || \rho') = C_2 \oplus \mathrm{H}_2(Z, Z^{sk})$.
  2. Return M' if $C_1 = (P^{\mathrm{H}_1(id)} \cdot P_0)^{\mathrm{H}_3(\rho, M, X, id)}$, $\perp$ otherwise.

LQ06c is obtained by instantiating *LQ06g* with *SK-IBE* and ElGamal in $\mathbb{G}_\mathrm{T}$.

Same as *AP03*, decryption recovers the input of H_2 used by the sender. Recall that the public key in *AP03* is in the form of $\langle X = P^{sk}, Y = {P_0}^{sk} = P^{s \cdot sk}\rangle$. Note that X is not involved in *AP03*'s message hiding at all. This "special" structure is for ensuring the input of H_2, $\hat{e}(Y, \mathrm{H}_1(id))^r$), involves both the secret value sk (of the user) and the master secret key s (of the KGC), but it does not mean that they must be glued together as a single element. With the property of RO, $\mathrm{H}_2(b_1 b_2 \cdots)$ is unknown even a single bit out of b_i's is unknown. This property is utilized in *LQ06c* (Z here can only be computed from the partial private key, and Z^{sk} can only be computed from the user secret value). It also suggests how [ARP05] uses $X = P^{sk}$ only as the "main" public key without $Y = {P_0}^{sk}$ in *AP03*.

Standard Model Too much FO transformation? Let us make a detour to escape from ROM. *DLP08c* [DLP08], based on *Waters-IBE* [Wat05], uses a probabilistic partial key generation instead of a deterministic one in *AP03* or *LQ06c*.

- Setup:
  * **Encryption key**: choose two generators $g, g_2 \in_R \mathbb{G}$.
  * **Master public key**: choose an exponent $\alpha \in_R \mathbb{Z}_p$ and set $g_1 = g^\alpha$.
  * **Hash key for identity-based public key derivation**: pick $(\ell + 1)$ $\mathbb{G}$ elements $\vec{U} = (u', u_1, \cdots, u_\ell)$. Let $id = \iota_1 \cdots \iota_\ell$. Define $F_u(id) = u' \prod_{j=1}^{\ell} u_j^{\iota_j}$.
  * **Hash key for ciphertext validity**: pick $\vec{V} = (v', v_1, \cdots, v_\ell) \in_R \mathbb{G}^{\ell+1}$ This vector defines $F_v(w) = v' \prod_{j=1}^{\ell} {v_j}^{b_j}$ where w is an ℓ-bit string $b_1 b_2 \cdots b_\ell$.

$$msk = g_2^\alpha,\ mpk = \langle g, g_1, g_2, \vec{U}, \vec{V}\rangle,\ pub = \langle \lambda, q, \mathbb{G}, \mathbb{G}_\mathrm{T}, \hat{e}(\cdot,\cdot), \ell, \mathrm{H}(\cdot)\rangle\ .$$

- $\mathsf{KeyDer}(g_2^\alpha, id)$: Pick $\tau \in_R \mathbb{Z}_p^*$, return $D_{id} = \langle d_1, d_2\rangle = \langle {g_2}^\alpha \cdot F_u(id)^\tau, g^\tau\rangle$.
- SetUK: $sk \in_R \mathbb{Z}_p^*$ and $pk = \langle X, Y\rangle = \langle g^{sk}, {g_1}^{sk}\rangle$.
- $\mathsf{Enc}(M, id, \langle X, Y\rangle)$:
  1. Check if pk is valid by $\hat{e}(X, g_1) = \hat{e}(g, Y)$, return $\perp$ if not.
  2. Pick $r \in_R \mathbb{Z}_p^*$, return $\langle C_0, C_1, C_2, C_3\rangle = \langle M\hat{e}(Y, g_2)^r, g^r, F_u(id)^r, F_v(w)^r\rangle$ where $w = \mathrm{H}(C_0 || C_1 || C_2 || id || pk)$, H is a collision-resistant hash function.
- $\mathsf{Dec}(\langle C_0, C_1, C_2, C_3\rangle, \langle d_1 d_2\rangle, sk)$:
  1. Compute $w' = \mathrm{H}(C_0 || C_1 || C_2 || id || pk)$.
  2. Check if $\hat{e}(C_1, F_u(id) F_v(w')) = \hat{e}(g, C_2 C_3)$.
  3. Return $\perp$ if it does not hold; otherwise, return $M = C_0 \cdot \{\frac{\hat{e}(C_2, d_2)}{\hat{e}(C_1, d_1)}\}^{sk}$.

The random oracle approach to realize strong decryption oracle is based on FO transformation which is originally proposed to achieve CCA security for PKE. Interestingly, the strong decryption oracle of *DLP08c* is made possible by a direct CCA transformation technique without random oracles [BMW05]. It is clear that the random factor r "hides" the message M. From a high-level point of view, C_3 is "another" ciphertext produced from the same r as that in $\langle C_0, C_1, C_2\rangle$ (note that C_3 is never used in the actual message recovery step in Dec). Thus, the simulator is able to extract useful information about r from C_3, irrespective of the public key. More concretely, the mapping F_v is "loaded" such that $F_v = g_2^{J_v(w)} g^{K_v(w)}$ for certain functions $J_v(\cdot)$ and $K_v(\cdot)$ in the environment simulated by the challenger. The validity checking ensures $C_1 = g^r$ and $C_3 = F_v(w)^r$ for some $r \in \mathbb{Z}_p^*$, and hence the simulator can extract $g_2^r = (C_3/C_1^{K_v(w)})^{1/J_v(w)}$ if $J_v(w) \neq 0 \bmod q$. The padding $\hat{e}(Y, g_2)^r$ can thus be easily recovered for whatever Y.

4. Malicious KGC Attack

The Type-II model in Section 2.2 considers an *honest-but-curious* KGC. We know that if a KGC is *actively-malicious*, it can always fool a message sender to use an adversarially generated public key, which makes decryption a trivial task. But how about something in between, like *malicious-but-passive*? Apart from distributing false public keys, there are other things that a KGC can do. Recall that the KGC is responsible for generating system parameters, so it may not perform in complete accordance with the Setup algorithm. In particular, it may keep the intermediate data used that is not supposed to be part of the master secret key. In other words, a KGC may compromise the confidentiality of the messages by embedding extra "trapdoors" in the system parameters. The KGC can actively generate the parameters maliciously once but remains passive later on.

Take $\mathcal{DLP}08c$ as an example; a KGC who picks[2] $\beta \in_R \mathbb{Z}_p^*$ and computes g_2 as g^β, can recover the padding $\hat{e}(Y, g_2)^s$ in every ciphertexts by $\hat{e}(Y, C_1)^\beta$. This kind of attack can be generally avoided by requiring the KGC to demonstrate the randomness in parameter generation. Suppose h is a cryptographic hash function, the KGC can publish a witness S such that $g_2 = \mathrm{h}(S)$ (and any other parameters which need the same treatment). The existence of a common reference string is thus assumed. IEEE P1363 and ANSI X9.62 standards have shown examples of methods to generate verifiably random parameters. This has been considered as a common assumption in other cryptographic primitives (e.g. [ACJT06]).

In 2007, Au *et al.* [ACL$^+$07] brought to light this seemingly neglected concern in certificateless paradigm. They demonstrated how the confidentiality of a single pre-selected user can be compromised in $\mathcal{AP}03$, proposed an appropriate security model for malicious KGC attack, and gave a proof showing a sequential composition of PKE and IBE is secure in their model, assuming random oracle. Later, [CCLC07] questioned whether the challenger or the adversary is controlling the random oracle. If the adversary can control a random oracle in addition to the parameters generation, it might be difficult to get security even in the ROM.

Since the master secret key is generated by the adversary and is never handed in to the challenger, Au *et al.* chose not to model any form of decryption request in their proof; but they claimed that certificateless FO transformation in [LQ06] can be applied to achieve full security (which suggests that the random oracle is probably on the challenger side). Unfortunately, Dent [Den08] demonstrated an attack against a scheme obtained from applying certificateless FO transformation on a contrived scheme, where the

[2]It has been specified in [DLP08] that such an element β should be securely erased.

structure of a partial private key (PPK) is changed so that it can even embed a ciphertext[3]. In response, a new **Weak PPK Decryption** oracle is introduced, which can been seen providing an interface for replacing (i.e. maliciously generating) the partial private key of a user.

4.1. Malicious KGC Model

We present a security model proposed by Dent [Den08] for Malicious Weak Type-II adversary, which is extended from the three-phased game in [ACL$^+$07].

Phase 0 On input pub, A outputs mpk. No oracle is available to A in Phase 0.

Phase 1 A can query **Request Public Key** and **Weak PPK Decryption** oracles.

Challenge Phase A terminates Phase 1 by outputting an identity id^* and two messages M_0^*, M_1^*. C picks a random bit b, returns the challenge ciphertext $C^* = \mathsf{Enc}(M_b^*, id^*, pk^*)$ to A, where pk^* is the public key of user id^*.

Phase 2 A can continue to make oracle queries, subjected to restrictions.

Guess When A has finished with Phase 2, A must output a guess bit b'. A wins the game if $b' = b$ and A's advantage is defined as $2 \times |\Pr[b' = b] - 1/2|$.

Request Public Key has been defined. Now we define the remaining one.

- **Weak PPK Decrypt** On input a ciphertext C, an identity id and an arbitrary partial private key D_{id}, return $\mathsf{Dec}(C, D_{id}, sk)$.

Definition IX.3 A certificateless encryption scheme is said (t, q_D, ϵ) $\mathrm{ind\text{-}id\text{-}cca}$ secure against a Malicious Weak Type-II adversary if A's advantage defined above is less than ϵ for all t-time adversary A making at most q_D decryption queries, subjects to the constraint that A may not query **Weak PPK Decrypt** oracle with the challenge ciphertext C^* on the identity id^*, irrespective of any partial private key value, after C^* is defined.

It is possible to relax the restriction such that the adversary can put the challenge ciphertext to the **Weak PPK Decrypt** oracle as long as the partial private key is not for id^*. Dent [Den08] chose not to include this in the model since the ability to distinguish between a valid and an invalid partial private key for a particular id is not guaranteed. This is true but in almost all existing CLE schemes, the partial private key is verifiable. Even so, the stricter restriction still gives a strong enough adversary model for schemes with verifiability since a user can easily check that a given partial private key is invalid and refuse to use it.

[3]The attack is as follow. If a decryption oracle “blindly” takes a PPK supplied by the adversary (i.e. it does not care what the PPK is and it is fine even a ciphertext is embedded) and decrypts using this PPK and the real secret value, then a challenge ciphertext can be decrypted without violating the restriction, via the specially design decryption algorithm of a contrived scheme.

4.2. Generic Constructions

Dent [Den08] proved that the CLE scheme constructed from Dodis-Katz technique [DK05] (which is also how $\mathcal{CBG}06g$ in [CBG06] is constructed) remains secure against malicious KGC attack with **Weak PPK Decrypt**, which is stronger than the proofs in [ACL$^+$07,HW07a,HW07b]. Here, we review this generic construction without random oracles, which we call $\mathcal{DDK}08$. It makes use of a PKE scheme $\mathcal{PKE}$ and an IBE scheme $\mathcal{IBE}$ with CCA-security adapted to deal with labels [Sho01], and a one-time signature scheme $\{\mathsf{SGen}, \mathsf{Sign}, \mathsf{Vf}\}$.

- $\mathsf{Setup} \equiv \mathcal{IBE}.\mathsf{Setup}$, $\mathsf{KeyDer} \equiv \mathcal{IBE}.\mathsf{KeyDer}$, $\mathsf{SetUK} \equiv \mathcal{PKE}.\mathsf{KeyGen}$
- $\mathsf{Enc}(M, id, pk)$:
  1. Execute SGen to get a verification/signing key pair (ovk, osk).
  2. Randomly pick s_1 from the message space and compute $s_2 = M \oplus s_1$.
  3. Compute $C_1 = \mathcal{IBE}.\mathsf{Enc}^{ovk}(s_1, id)$ where ovk is the label.
  4. Compute $C_2 = \mathcal{PKE}.\mathsf{Enc}^{ovk}(s_2, pk)$ where ovk is the label.
  5. Sign $(C_1||C_2)$ by the one-time signature scheme: $\sigma = \mathsf{Sign}(C_1||C_2, osk)$.
  6. Output $C = \langle C_1, C_2, ovk, \sigma\rangle$.
- $\mathsf{Dec}(\langle C_1, C_2, ovk, \sigma\rangle, D_{id}, sk)$:
  1. Verify $(C_1||C_2)$ by the one-time signature scheme: $\mathsf{Vf}(\sigma, (C_1||C_2), ovk)$.
  2. Compute $s_1' = \mathcal{IBE}.\mathsf{Dec}^{ovk}(C_1, D_{id})$ where ovk is the label.
  3. Compute $s_2' = \mathcal{PKE}.\mathsf{Dec}^{ovk}(C_2, sk)$ where ovk is the label.
  4. Return $M' = s_1' \oplus s_2'$ if Vf passes, otherwise return $\perp$.

Since $\mathcal{PKE}$ and $\mathcal{IBE}$ are used as black-boxes, and the system parameters only consist of those from $\mathcal{IBE}$, it is counterintuitive that $\mathcal{IBE}$ component will contain some “trapdoors” for the decryption of $\mathcal{PKE}$. This gives a feeling that generic constructions should be easier to achieve security against malicious KGC attack. On the other hand, a similar argument cannot be applied in concrete constructions since the PKE and the IBE components are often “twisted” together.

Intuitively, Dodis-Katz technique in using one-time signature to sign each ciphertext and binding the corresponding verification key to each component of the ciphertext (via the use of labels) makes the adversary cannot “tweak” the challenge ciphertext and put another version of it to the decryption oracle. While for the challenge ciphertext itself, the security model does not allow it to be put to the **Weak PPK Decrypt** oracle, no matter what the PPK is; all these taken together makes querying **Weak PPK Decrypt** practically useless for the adversary.

In [HW07a], a sequential variant of the above idea is presented, which tried to simplify the design by not explicitly using label in both IBE and the PKE component. However, their IBE component encrypts the verification key ovk (or a hash of it) along with the message, which is just a ciphertext-expanding way to realize the non-malleable label. Actually, various submissions for an ISO standard for PKE [Sho01] support label pretty efficiently. Finally, note that sequential encryption means the message space required by the outer layers of encryption should be large enough to cater for the ciphertext expansion of the inner encryptions.

5. Certificateless Encryption without Pairing

5.1. Simplified Framework

Currently, all concrete constructions [BSNS05,LK07,SZB07] of certificateless encryption without pairing follow the Baek, Safavi-Naini and Susilo's relaxation of the original Al-Riyami and Paterson model, where the construction of public key requires not only a secret value but also a partial private key (issued by the KGC to a user). Similar to the simplification brought by the inclusion of SetUK, [Den08] formulated a very simple framework of CLE as a quintuple of algorithms consisting of the original KeyGen, KeyDer, Enc, and Dec, with SetUK$'$ defined below.

- SetUK$'$ is a probabilistic algorithm which takes an identity id and a partial private key D_{id}, outputs a public/private key pair (pk, sk) or error $(\perp)$.

The secret value only appears as an internal variable in SetUK$'$. Assuming it is securely erased and **Extract Full Private Key** oracle is available, **Extract Secret Value** is no longer necessary in this formulation.

Consequently, **Weak SV Decrypt** should not be available. On the other hand, **Strong Decrypt** may be too strong. If an arbitrary public key is presented, the simulator is expected to generate the corresponding PPK for the decryption. This is the opposite of the reality where PPK is generated first and then determines part of the public key. This subtlety has been discussed in the context of certificateless signature in [YCHG07]. The Type-I adversary model for this formulation in [Den08] has a **Decrypt** oracle which only decrypts under the *original* private key, but neither **Strong Decrypt** nor **Weak SV Decrypt** is available.

For Malicious Type-II adversary, the master secret key msk is generated by the adversary but not the challenger. Note that msk is required to compute a PPK, which in turns determines part of the public key. Does it mean that the public key is now chosen by the adversary? A careful definition is thus needed.

5.2. Concrete Constructions

We review $\mathcal{SZB}07$ [SZB07] and explain how it fixes $\mathcal{BSS}05$ [BSNS05].

- Setup: $msk = s \in_R \mathbb{Z}_q^*$, $mpk = y = g^s \in \mathbb{Z}_p^*$, $(q|(p-1)$, so $\mathbb{Z}_p^*$ is of order $q)$, $pub = \{g, p, q, \mathrm{H}_0(\cdot), \mathrm{H}_1(\cdot), \mathrm{H}_2(\cdot,\cdot), \mathrm{H}_3(\cdot)\}$.
- KeyDer(s, id):
  1. Pick $\tau_0, \tau_1 \in_R \mathbb{Z}_q^*$, compute $w_0 = g^{\tau_0}, w_1 = g^{\tau_1}$
  2. Compute $d_0 = \tau_0 + s\mathrm{H}_0(id||w_0)$ and $d_1 = \tau_1 + s\mathrm{H}_1(id||w_0||w_1)$
  3. Output $D_{id} = \langle d_0, P_{id}\rangle$ where $P_{id} = \langle w_0, w_1, d_1\rangle$.
     (P_{id} will be made public, which can be considered as partial public key.)
- SetUK$'$: $sk = \langle x \in_R \mathbb{Z}_q, d_0\rangle$, $pk = \langle u = g^x, w_0, w_1, d_1\rangle$.
- Enc(M, id, pk):
  1. Check if $g^{d_1} = w_1 y^{\mathrm{H}_1(id||w_0||w_1)}$; otherwise, $\perp$ is returned.
  2. Pick $\rho \in_R \mathbb{Z}_q$, compute $r = \mathrm{H}_3(\rho||M||u||id) \in \mathbb{Z}_q^*$,
  3. Output $C = \langle C_1, C_2\rangle = \langle g^r, (M||\rho) \oplus \mathrm{H}_2(w_0^r y^{\mathrm{H}_0(id||w_0)r}, u^r)\rangle$.

- $\mathsf{Dec}(\langle C_1, C_2\rangle, \langle x, d_0\rangle)$:
  1. Recover $Z_1 = C_1^{d_0}$ and $Z_2 = C_1^x$, retrieve $(M'||\rho') = C_2 \oplus \mathrm{H}_2(Z_1, Z_2)$.
  2. Return M' if $C_1 = g^{\mathrm{H}_3(\rho'||M'||u||id)}$, $\perp$ otherwise.

Enc and Dec are obtained by following *LQ06g*, i.e. certificateless FO transformation in [LQ06], with Z_1 and Z_2 are the paddings from ElGamal encryption under the "partial public key" $w_0 y^{\mathrm{H}_0(id||w_0)}$ and the public key u.

The d_0 element in the partial private key is basically a Schnorr signature on the message id signed by s. As a result, there is an element w_0 which is originally part of the Schnorr signature embedding the randomness used in signing, which is fine to be public. Actually, it *must* be made public. Otherwise, considering how one can decrypt the ciphertext with d_0, applying d_0 on C_1 results in $C_1^{d_0} = (g^{r\tau_0})y^{\mathrm{H}_0(id||w_0)r}$. Computing $(g^{r\tau_0})$ from C_1 requires knowing τ_0, which cannot be leaked to the user (or s can be recovered from τ_0 and d_0), hence the term w_0^r should somehow also appear in the ciphertext, which means w_0 should be available to the message sender. This explains why in this kind of schemes the public key can only be distributed after a partial private key has been used.

So far, all the discussions also apply on *BSS05*. It remains to explain the significance of d_1 from the partial private key, which is eventually distributed in the public key but not used in the decryption at all. The public key in *BSS05* only contains $\langle u, w_0\rangle$. Their proof cannot go through if w_0 is replaced by a Type-I adversary. To fix this problem, *SZB07* uses another Schnorr signature $\langle w_1, d_1\rangle$ to certify on $(id||w_0)$. The security of the signature scheme makes the partial private key of id cannot be computed (c.f. unforgeable) by the adversary even it can get the partial private keys for other identities (c.f. chosen message attack). With the verification of (w_1, d_1) in Enc, w_0 cannot be replaced. This makes the simulation against a Strong Type-I adversary easier as the adversarially-generated public keys will not be "completely arbitrary". Signature-like techniques have been used to ensure the non-malleability of the ciphertext (e.g. the $F_v(w)^s$ term in *DLP08c*). Here the signature is utilized to protect the partial private key as well.

6. Security-Mediated Certificateless Encryption

6.1. Revocation Problem

Certificateless cryptography solves the escrow problem of ID-based cryptography, but still inherits some of its problems. One of them which should be solved before practical deployment is revocation. A private key may become compromised and hence the public key should be revoked. Solutions to the revocation problem often require online checking, such as the certificate revocation lists approach.

A pragmatic way has been suggested to deal with this problem in ID-based cryptography, which is appending a validity period to the end of an identity. To encrypt, the sender needs to retrieve the parameters for the current period (frequent sender may just need to check a clock which is loosely synchronized with the system though). The receiver then retrieves a new private key which is only given by the KGC at the beginning of validity period. This solution requires the action of all the involved parties, in particular, the KGC needs to be virtually permanently online. Considering the KGC is holding the master secret which affects all the system users, it is undesirable.

An instant revocation mechanism is suggested by Boneh, Ding and Tsudik [BDT04]. There is an online mediator for every cryptographic operation, which is referred to as a SEM (SEcurity Mediator) since it provides a control of security capabilities. If the SEM does not cooperate, the cryptographic task cannot be completed. Revocation is made possible by notifying the SEM to stop answering any requests for the revoked user. This is particularly useful in government, corporate or military environments, where there are immediate needs for revocation when a user suspects key compromise, or when a user is removed from a position of authority; but it may be an overkill for "casual" applications.

In 2006, Chow, Boyd and González Nieto [CBG06] brought this concept to certificateless cryptography by formalizing the notion of security-mediated certificateless (SMC) cryptography and proposing two constructions for security-mediated certificateless encryption (SMCLE) schemes. The KGC delegates the partial private key (termed as "the SEM key" in SMC paradigm) to a SEM (possibly, more than one SEM serve a particular user and a SEM serve more than one user) which helps the users for decryption. In contrast with previous solutions, the KGC is not expected to be online indefinitely. Compared with decryption in a normal CLE, it is true that the user can no longer decrypt locally. However, considering the ciphertext should be originated from somewhere in the network (better ways exist for secure storage of messages from previous decryptions), the user is online before the decryption anyway, hence contacting a SEM is not unrealistic, especially when modern networks become more widely available and reliable.

6.2. Model

In SMCLE, a ciphertext is first decrypted by the SEM to give a partially decrypted ciphertext, which will be decrypted by the user, via these decryption algorithms:

- SEM-Dec is a probably probabilistic algorithm which takes a ciphertext C, an identity id, a public key X and the SEM-key D_{id} as input, it first checks if the user with identity id is a legitimate one whose key is X and is not revoked; if the checks pass, it returns a partial decryption result C' or $\perp$.
- User-Dec is a (usually) deterministic algorithm which takes a partial decryption result C' and a secret value sk, returns a message M or $\perp$.

The security model introduced in [CBG06] is a natural generalization of CLE to cater for SEM partial decryption, in the flavor of **Strong Decrypt**.

- **SEM Decrypt** On input a ciphertext C and an identity id, A obtains a partial decryption of C using the SEM-key D_{id}.
- **User Decrypt** On input a partial decryption C' and an identity id, A obtains a correct decryption of C' with respect to the (probably replaced) public key of id.

There are two pieces of secret of a user which can be compromised – the SEM key and the secret value. There is no concept of *full private key* in SMC paradigm.

The constraints which prevent the adversary from winning trivially are:

1. A cannot ask both of **Extract SEM Key** and **Extract Secret Value** (which is implicitly issued if A has replaced the public key).
2. A cannot ask a **SEM Decrypt** query for the challenge ciphertext C^* under the challenge identity id^* if A has queried **Extract Secret Value** for id^*.

3. A cannot ask a **User Decrypt** query for C' where C' is the result of SEM-Dec *algorithm*, if A has requested the correct SEM-key (or msk).

The constraint on **User Decrypt** requires the ability to check if a partial decryption result is valid with respect to a ciphertext and an identity, which is reasonable in SMCLE since the user should be able to distinguish between the cases an invalid ciphertext is sent by the sender at the first place or the partial decryption result from the SEM has something wrong (e.g. maliciously replaced by another party over an unsecured network).

Previous models for mediated cryptography fall short in many aspects. For example, Libert and Quisquater's model [LQ03] allows the leakage of SEM keys, but does not allow the adversary to get the secret value of the target user or a decryption oracle using this secret value. Indeed, while it may look easy to separate the decryption algorithm of a plain CLE into SEM decryption and user decryption, CCA security would be probably lost due to the leakage of secret information via partial decryption. For example, for all RO-based schemes we reviewed so far, checking the ciphertext validity requires the message to be recovered, which is not possible for the SEM decryption since it is not supposed to recover the original message. Conversely, there is no security problem for combining these two partial decryption algorithms together to get back the original CLE. Thus, the first CLE scheme without random oracles has been proposed as early as in 2006 (*CBG06g*), but this seems to be neglected in some literature [LAS07,LK07,PCHL07].

6.3. Random Oracle Approach

We review *CBG06c* [CBG06], which is the first concrete SMCLE in the ROM.

- Setup, KeyDer: Same as *AP03*, but five hash functions are needed $(\mathrm{H}_1, \cdots, \mathrm{H}_5)$.
- SetUK: $sk \in_R \mathbb{Z}_p$, $pk = X = P^{sk}$.
- Enc(M, id, X):

  1. Pick $\rho \in_R \mathbb{Z}_p$, compute $r = \mathrm{H}_3(\rho, M) \in \mathbb{Z}_p^*$ and $C_1 = P^r$
  2. Compute $C_2 = (M||\rho) \oplus \mathrm{H}_2(\hat{e}(P_0, \mathrm{H}_1(id))^r) \oplus \mathrm{H}_4(X^r)$.
  3. Compute $C_3 = \mathrm{H}_5(X||C_1|\ C_2)^r \in \mathbb{G}$.
  4. Output $C = \langle C_1, C_2, C_3\rangle$.

- SEM-Dec$(\langle C_1, C_2, C_3\rangle, id, X, D_{id})$:

  1. Check that the user is not revoked, abort otherwise.
  2. Proceed if $\hat{e}(P, C_3) = \hat{e}(C_1, \mathrm{H}_5(X||C_1||C_2))$, return $\perp$ otherwise.
  3. Recover $Z_1 = \hat{e}(C_1, D_{id})$ and return $\langle C_1, C_2' = C_2 \oplus \mathrm{H}_2(Z_1)\rangle$.

- User-Dec$(\langle C_1, C_2'\rangle, sk)$:

  1. Recover $Z_2 = C_1^{sk}$ and retrieve $(M'||\rho') = C_2' \oplus \mathrm{H}_4(Z_2)$.
  2. Return M' if $C_1 = P^{\mathrm{H}_3(\rho', M')}$, $\perp$ otherwise.

The ciphertext of *CBG06c* looks similar to that in [ARP05], in which the messages are hidden by an exclusive-OR of separate paddings from IBE component and PKE component. However, [ARP05] is vulnerable to the following attack [LQ06]. A Type-I adversary first unwraps the PKE padding from the challenge ciphertext and wraps it again with another padding for a new public key, which can be done with the knowledge of the respective secret value. The message can be obtained by a strong decryption query of this transformed ciphertext under a replaced public key. This also demonstrates the power of public key replacement attack. In $\mathcal{LQ}06c$, this vulnerability is removed by feeding Z_1 and Z_2 into a single hash instead of hashing them separately. But its security proof is for plain CLE and cannot be used to guarantee the CCA-security for SMCLE. To see this, partial decryption requires leaking Z_1, which is never returned to the adversary.

$\mathcal{CBG}06c$ took another approach. Both C_1 and C_2 are "signed" by using the random factor in C_1 as a signing key following the short signature scheme in [BLS04]. It is thus not possible to change C_2 without being detected, as the adversary needs to "forge" the term C_3 on a new C_2. This public ciphertext validity check (i.e. without using any secret of the intended recipient or the knowledge of the message) also made the partial decryption by SEM immune to CCA-attack.

6.4. Standard Model Approach

The above technique in supporting partial decryption is essentially a non-interactive zero knowledge (NIZK) proof of the randomness in the ciphertext, which requires H_5 to be a random oracle. But an arbitrary NIZK proof in the standard model is inefficient. Is it possible to have efficient SMCLE scheme in the standard model? We first study a generic construction ($\mathcal{CBG}06g$) [CBG06].[4]

- $\mathsf{Setup} \equiv \mathcal{IBE}.\mathsf{Setup}$, $\mathsf{KeyDer} \equiv \mathcal{IBE}.\mathsf{KeyDer}$, $\mathsf{SetUK} \equiv \mathcal{PKE}.\mathsf{KeyGen}$
- $\mathsf{Enc}(M, id, pk)$:

  1. Execute SGen to get a verification/signing key pair (ovk, osk).
  2. Randomly pick s_1 from the message space and compute $s_2 = M \oplus s_1$.
  3. Compute $C_1 = \mathcal{IBE}.\mathsf{Enc}^{ovk}(s_1, id)$ and $C_2 = \mathcal{PKE}.\mathsf{Enc}^{ovk}(s_2, pk)$.
  4. Compute $\alpha = \mathrm{H}(C_1, C_2, s_1)$ and sign it by $\sigma = \mathsf{Sign}(\alpha, osk)$.
  5. Output $C = \langle C_1, C_2, ovk, \sigma\rangle$.

- $\mathsf{SEM\text{-}Dec}(\langle C_1, C_2, ovk, \sigma\rangle, id, X, D_{id})$:

  1. If $\langle id, X\rangle$ is not revoked, compute $s'_1 = \mathcal{IBE}.\mathsf{Dec}^{ovk}(C_1, D_{id})$.
  2. Recover $\alpha' = \mathrm{H}(C_1, C_2, s'_1)$ and verify σ by $\mathsf{Vf}(\sigma, \alpha, ovk)$.
  3. If the verification passes, return $\langle C_1, C_2, ovk, \sigma, s'_1\rangle$, $\perp$ otherwise.

- $\mathsf{User\text{-}Dec}(\langle C_1, C_2, ovk, \sigma, s'_1\rangle, sk)$:

  1. Compute α and check σ similar to $\mathsf{SEM\text{-}Dec}$, output $\perp$ if it fails.
  2. Otherwise, compute $s'_2 = \mathcal{PKE}.\mathsf{Dec}^{ovk}(C_2, sk)$ and return $M' = s'_1 \oplus s'_2$.

It is easy to see $\mathcal{CBG}06g$ is similar to $\mathcal{DDK}08$. Indeed, they are both originated from Dodis-Katz's multiple encryption technique – one for security against SEM decryption and another for security against malicious KGC attack. Due to the latter property, it is unclear how the above scheme can support **Strong Decrypt** oracle. Weak Type-I security and Malicious Type-II security come from the strong-multiple chosen-ciphertext security [DK05] of the underlying multiple encryption, which allows the adversary to ask arbitrary queries to the individual decryption oracles, i.e. **SEM Decrypt** and **User Decrypt** oracles.

[4] with corrections to the errata described in [YCHG07].

Recently, Chow, Roth and Rieffel [CRR08] proposed a concrete SMCLE scheme (*CRR08*) without random oracles. It follows a new CLE model which is general enough to support other applications of CLE like timed-released encryption and cryptographic workflow. It is also the first CLE scheme supporting hierarchical identities with security proof. Due to space limitation we only highlight their technique in supporting partial decryption, which can be seen as an augmented algorithm for *DLP08c*.

- SEM-Dec($\langle C_0, C_1, C_2, C_3\rangle, \langle d_1, d_2\rangle$):
  1. Compute $w' = \mathrm{H}(C_0||C_1||C_2|\ id||pk)$.
  2. Return $\perp$ if $\hat{e}(C_1, F_u(id)F_v(w')) \neq \hat{e}(g, C_2C_3)$.
  3. If equality holds, randomly pick $t \in \mathbb{Z}_p^*$, return $U = \langle u_1, u_2, u_3\rangle$, where $u_1 = d_1 \cdot F_v(w')^t$, $u_2 = d_2$, $u_3 = g^t$.
- User-Dec($\langle C_0, C_1, C_2, C_3\rangle, \langle u_1, u_2, u_3\rangle, sk$):
  1. Compute $w' = \mathrm{H}(C_0||C_1||C_2||id||pk)$.
  2. Check if $\hat{e}(C_1, F_u(id)F_v(w')) = \hat{e}(g, C_2C_3)$.
  3. Return $\perp$ if it does not hold; otherwise, return $M = C_0 \cdot \{\frac{\hat{e}(C_2,u_2)\hat{e}(C_3,u_3)}{\hat{e}(C_1,u_1)}\}^{sk}$.

While *DLP08c* uses the CCA technique in [BMW05] to support strong decryption oracle, *CRR08* uses the same technique to support partial decryption oracle with CCA security. Intuitively, $\langle u_1, u_2, u_3\rangle$ is a decryption key only for the ciphertext "associated" with $\langle id, w\rangle$ but nothing else. It is also easy to check the validity of a partial decryption (by $\hat{e}(g, u_1) \overset{?}{=} \hat{e}(g_1, g_2)\hat{e}(u_2, F_u(id))\hat{e}(u_3, F_v(w'))$), thus the **User Decryption** oracle can detect when the challenge ciphertext is presented with its valid partial decryption.

7. Efficiency Issues

Recall that the benefits of CLE are the removal of *transfer* and *verification*, now we look into some CLE schemes where they are still "necessary".

7.1. Key Verification versus Certificate Verification

In PKE approach, the message sender needs to retrieve the authenticated parameters from the CA, the user-generated public key, and the certificate signed by the CA. In CLE, the message sender also needs to retrieve the authenticated parameters from the KGC, the user-generated public key, but not any certificate.

There exists discrete-logarithm based PKE schemes (e.g. [CHK04]) in which the public key includes only a group generator g and another group element. This means the public key consists of only one element if the group generator is included in the CA parameters and shared by all users. In *AP03*, the public key consists of two elements, X and Y, which means its size is larger by one element when compared with some PKE schemes. Now the problem is, can a certificate be represented in one group element? The answer is yes, we have *BLS-Sig*, a signature scheme which offer this competitive signature size [BLS04]. Hence, *AP03* offers no advantage in this regard. Moreover, the message sender also needs to ensure that $\hat{e}(P, Y) = \hat{e}(P_0, X)$ holds. This is nearly the same as the verification algorithm of *BLS-Sig*. In other words, treating Y as a certificate (certifying X is well-formed), the sender still needs to do "certificate verification". This is more obvious in $\mathcal{SZB}07$, which simply includes a Schnorr signature in the public key.

It is possible to integrate the public key verification with the encryption by computing $\mathrm{H}_2(\{\hat{e}(Y, \mathrm{H}_1(id) \cdot P^{\beta})/\hat{e}(X, P_0^{\beta})\}^r)$ instead of $\mathrm{H}_2(\hat{e}(Y, \mathrm{H}_1(id))^r)$ for a random $\beta \in_R \mathbb{Z}_p^*$, so only one extra pairing is required. However, it makes future encryption inefficient since $\hat{e}(Y, \mathrm{H}_1(id))$ is not pre-computed in this case; also note that the certificate verification is required for once only in PKE.

The situation seems to be better for the scheme in standard model. $\mathcal{CRR}08$, $\mathcal{DLP}08c$ consist of two-element public key and take two pairing operations for verification. Certificate based on short signatures without random oracles does not offer a more compact size. For example, [BB04c] (see also [BB08]) still has a signature length of two group elements; but it requires only one pairing operation in verification.

We make an important remark that the above discussion assumes all the schemes are using groups of the same size. Different schemes have different tightness of security reduction and rely on different assumptions, it is thus unfair to compare them merely by "counting". Nevertheless, we aim to raise that it is not sufficient to provide only a "functional" CLE, without addressing these concerns.

7.2. Self-Generated-Certificate Encryption versus Public-Key Encryption

In CLE, there is no certificate helping the identification of the true public key. The sender may make a wrong choice when faced with many public keys, or even tricked to use one which is never owned by the intended recipient. This is termed as "denial-of-decryption" (DoD) attack by Liu, Au and Susilo [LAS07], since this possibly denies the recipient's opportunity to get a correct decryption result. Based on the notion of CLE, they propose the idea of self-generated-certificate public key encryption (SGC-PKE) to address this problem.

A Type-II adversary can always actively certify a false public key, it is thus reasonable to consider only Type-I adversary in modeling of DoD attack. The model in [LAS07] is simple. The adversarial goal is to output an identity id^*, a valid public key pk^* and a message M^* such that a ciphertext given by an honest encryption $\mathsf{Enc}(M^*, id^*, pk^*)$ when decrypted by (D_{id^*}, sk^*) results in a message $M' \neq M^*$. The adversary A has usual accesses of oracles, but A cannot get the partial private key of id^*. **Weak Decrypt** oracle is used instead of **Strong Decrypt** in their model, possibly due to the feature provided by their concrete scheme.

Recall how implicit authentication is done in CLE – only the legitimate "owner" of an identity is entitled with the corresponding partial private key. Same as CLE, a KGC in SGC-PKE is *trusted* to only issue a partial private key after authentication. To get a correct copy of the public key, we can just ask the recipient to use this partial private key[5] to certify the public key. This is the underlying idea for their generic construction of of SGC-PKE. It takes a secure CLE and a secure certificateless signature (CLS). CLE and CLS both use a single user public key (and a single secret value). There are two partial private keys, one for CLE and the other for CLS, as a result there are two full private keys as well. A user first *signs* his/her public key pk using CLS private key to obtain

[5]The use of CLS (signed by a full private key) in [LAS07] may be an overkill, it suffices to use a partial private key, and signs by a security-mediated certificateless signature [YCHG07] which supports such "partial" signing with existential unforgeability under chosen-message attack.

σ. User's final public key is $\langle pk, \sigma\rangle$. A sender first *verifies* if σ is a valid CLS on pk before encrypting using CLE. The recipient just uses the CLE private key to decrypt as usual. The whole concept (and the name of the notion) is somewhat analogous to self-signed certificates in PKI (e.g. [Gir91]). We remark that [LK07] essentially instantiates this generic construction with a Schnorr-based CLS.

Let us compare SGC-PKE with PKE approach. In PKI, a user presents a public key pk and possibly a proof-of-knowledge (PoK) of the corresponding secret key to the CA. CA then supplies a certificate to the user, which is distributed with pk to the message sender, who checks the signature inside the certificate to see if pk is certified. If a PoK is also required in SGC-PKE, the only difference from the PKE approach seems to be the generation of the certificate. In SGC-PKE, the certificate is self-generated; while in the PKE, it is generated by the CA. However, note that the KGC is required to issue two partial private keys, but the CA is only required to issue one signature. Does SGC-PKE offer any advantage compared with PKE? It depends on whether two partial private key generations is more efficient than one message signing. Even the *certification process* looks implicit, the message senders still need to *retrieve* and *verify* the self-generated-certificates.

8. Summary

Certificateless encryption combines the best of traditional public key encryption and ID-based encryption. We have studied a few representative constructions which cover the techniques behind a large number of different schemes. We have also presented the corresponding security models and discussed the practical significances and subtleties of these formal definitions. We have studied the techniques behind addressing special security concerns in certificateless paradigm, which include key replacement attack, decryption capabilities for arbitrary public keys (in strongly-secure schemes), partial decryption capabilities (in security-mediated schemes) and malicious KGC attack. We have also examined some less obvious efficiency issues to obtain the real benefits brought by certificateless encryption. We conclude this chapter by a few possible research directions.

1. What theory about PKE is worthy of investigation to see if it helps CLE?
2. Can we construct CLE schemes related to factoring or lattices?
3. Can we find an application of security against strong decryption oracle?
4. Can we have an efficient solution to the DoD problem which does not involve the complexities of PKI? Or it is the price we must pay for CLE?

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-156

Chapter X

Attribute-Based Encryption

Amit SAHAI [a], Brent WATERS [b] and Steve LU [a]
[a] *UCLA, USA*
[b] *University of Texas at Austin, USA*

Abstract. As more sensitive data is shared and stored by third-party sites on the Internet, there will be a need to encrypt data stored at these sites. One drawback of encrypting data is that it can be selectively shared only at a coarse-grained level (i.e., giving another party your private key). We describe a new concept for fine-grained sharing of encrypted data that we call Attribute-Based Encryption (ABE). In ABE, ciphertexts are labeled with sets of attributes and private keys are associated with access structures that control which ciphertexts a user is able to decrypt. We demonstrate the applicability of ABE to sharing of audit-log information and broadcast encryption. We describe a construction that supports delegation of private keys, which subsumes Hierarchical Identity-Based Encryption (HIBE).

Keywords. Encryption, public key, attribute-based encryption

There is a trend for sensitive user data to be stored by third parties on the Internet. For example, personal email, data, and personal preferences are stored on web portal sites such as Google and Yahoo. The attack correlation center, `dshield.org`, presents aggregated views of attacks on the Internet, but stores intrusion reports individually submitted by users. Given the variety, amount, and importance of information stored at these sites, there is cause for concern that personal data will be compromised. This worry is escalated by the surge in recent attacks and legal pressure faced by such services.

One method for alleviating some of these problems is to store data in encrypted form. Thus, if the storage is compromised the amount of information loss will be limited. One disadvantage of encrypting data is that it severely limits the ability of users to selectively share their encrypted data at a fine-grained level. Suppose a particular user wants to grant decryption access to a party to all of its Internet traffic logs for all entries on a particular range of dates that had a source IP address from a particular subnet. The user either needs to act as an intermediary and decrypt all relevant entries for the party or must give the party its private decryption key, and thus let it have access to *all* entries. Neither one of these options is particularly appealing. An important setting where these issues give rise to serious problems is *audit logs* (discussed in more detail in Section 4).

In this chapter, we survey a new concept for solving this problem that we call Attribute-Based Encryption (ABE), introduced by Sahai and Waters [SW05]. In an ABE system, a user's keys and ciphertexts are labeled with sets of descriptive attributes and a particular key can decrypt a particular ciphertext only if there is a match between the attributes of the ciphertext and the user's key. The cryptosystem of Sahai and Waters allowed for decryption when at least k attributes overlapped between a ciphertext and a private key. While this primitive was shown to be useful for error-tolerant encryption with biometrics, the lack of expressibility seems to limit its applicability to larger systems.

Here, we describe a much richer type of attribute-based encryption cryptosystem due to Goyal et al. [GPSW06] and demonstrate its applications. In our system each ciphertext is labeled by the encryptor with a set of descriptive attributes. Each private key is associated with an access structure that specifies which type of ciphertexts the key can decrypt. We call such a scheme a Key-Policy Attribute-Based Encryption (KP-ABE) since the access structure is specified in the private key, while the ciphertexts are simply labeled with a set of descriptive attributes.[1]

We note that this setting is reminiscent of secret sharing schemes (see, e.g., [Bei96]). Using known techniques one can build a secret sharing scheme that specifies that a set of parties must cooperate in order to reconstruct a secret. For example, one can specify a tree access structure where the interior nodes consist of **AND** and **OR** gates and the leaves consist of different parties. Any set of parties that satisfy the tree can reconstruct the secret.

In our construction each user's key is associated with a tree-access structure where the leaves are associated with attributes.[2] A user is able to decrypt a ciphertext if the attributes associated with a ciphertext satisfy the key's access structure. The primary difference between our setting and secret sharing schemes is that *while secret sharing schemes allow for cooperation between different parties, in our setting this is expressly forbidden*. For instance, if Alice has the key associated with the access structure "X **AND** Y" and Bob has the key associated with the access structure "Y **AND** Z", we would not want them to be able to decrypt a ciphertext whose only attribute is "Y" by colluding. To do this, we adapt and generalize the techniques introduced by [SW05] to deal with more complex settings. We will show that this cryptosystem gives us a powerful tool for encryption with fine-grained access control for applications such as sharing audit log information.

In addition, we provide a delegation mechanism for our construction. Roughly, this allows any user that has a key for access structure $\mathbb{A}$ to derive a key for access structure $\mathbb{A}'$ if and only if $\mathbb{A}'$ is more restrictive than $\mathbb{A}$. Somewhat surprisingly, we observe that our construction with the delegation property subsumes Hierarchical Identity-Based Encryption [HL02,GS02] and its derivatives [ACD$^+$06].

1. Related Work

Fine-grained Access Control Fine-grained access control systems facilitate granting differential access rights to a set of users and allow flexibility in specifying the access

[1]This contrasts with what we call Ciphertext-Policy Attribute-Based Encryption (CP-ABE), where an access structure (i.e. policy) would be associated to each ciphertext, while a user's private key would be associated with a set of attributes. KP-ABE and CP-ABE systems are useful in different contexts.

[2]In fact, we can extend our scheme to work for any access structure realizable by a Linear Secret Sharing Scheme. We briefly discuss this notion in Section 3.

rights of individual users. Several techniques are known for implementing fine-grained access control.

Common to the existing techniques (see, e.g., [KPF01,GNO$^+$04,YW03,LLW05, HCM01,MP02] and the references therein) is the employment of a trusted server that stores the data in the clear. Access control relies on software checks to ensure that a user can access a piece of data only if he is authorized to do so. This situation is not particularly appealing from a security standpoint. For example, in the event of server compromise due to a software vulnerability exploit, the potential for information theft is immense. Furthermore, there is always a danger of "insider attacks" wherein a person having access to the server steals and leaks the information, for example for economic gains. Some techniques (see, e.g., [AT83]) create user hierarchies and require the users to share a common secret key if they are in a common set in the hierarchy. The data is then classified according to the hierarchy and encrypted under the public key of the set it is meant for. Clearly, such methods have several limitations. If a third party must access the data for a set, a user of that set either needs to act as an intermediary and decrypt all relevant entries for the party or must give the party its private decryption key, and thus let it have access to all entries. In many cases, by using the user hierarchies it is not even possible to realize an access control equivalent to monotone access trees.

In this chapter, we introduce new techniques to implement fine-grained access control. Under these new techniques, the data is stored on the server in an encrypted form while different users are still allowed to decrypt different pieces of data per the security policy. This effectively eliminates the need to rely on the storage server for preventing unauthorized data access.

Secret Sharing Schemes secret sharing schemes (SSS) are used to divide a secret among a number of parties. The information given to a party is called the share (of the secret) for that party. Every SSS realizes some access structure that defines the sets of parties who should be able to reconstruct the secret by using their shares.

Shamir [Sha79] and Blakley [Bla79] were the first to propose a construction for secret sharing schemes where the access structure is a threshold gate. That is, if any t or more parties come together, they can reconstruct the secret by using their shares; however, any smaller group of parties do not get any information about the secret. Benaloh and Leichter [BL90] extended Shamir's idea to realize any access structure that can be represented as a tree consisting of threshold gates. Other notable secret sharing schemes are [ISN87,Bri89].

In SSS, one can specify a tree-access structure where the interior nodes consist of AND and OR gates and the leaves consist of different parties. Any set of parties that satisfy the tree can come together and reconstruct the secret. Therefore in SSS, collusion among different users (or parties) is not only allowed but required.

In our construction, each user's key is associated with a tree-access structure in which the leaves are associated with attributes. A user is able to decrypt a ciphertext if the attributes associated with a ciphertext satisfy the key's access structure. In our scheme, contrary to SSS, users should be *unable* to collude in any meaningful way.

Identity-Based Encryption and Extensions The concept of Attribute-Based Encryption was introduced by Sahai and Waters [SW05], who also presented a particular scheme that they called Fuzzy Identity-Based Encryption (FIBE). The Fuzzy-IBE scheme builds upon several ideas from Identity-Based Encryption [BF03,Sha85,Coc01]. In FIBE, an identity is viewed as a set of attributes. FIBE allows for a private key for an identity ω to decrypt to a ciphertext encrypted with an identity ω' if and only if the identities ω and ω' are close to each other as measured by the "set intersection" distance metric. In other words, if the message is encrypted with a set of attributes ω', a private key for a set of attributes ω enables decrypting that message if and only if $|\omega \cap \omega'| \geq d$, where d is fixed during the setup time. Thus, FIBE achieves error tolerance, making it suitable for use with biometric identities. However, it has limited applicability to access control of data, our primary motivation here. Since the main goal in FIBE is error tolerance, the only access structure supported is a threshold gate whose threshold is fixed at the setup time.

We develop a much richer type of attribute-based encryption. The private keys of different users might be associated with different access structures. Our constructions support a wide variety of access structures[3], including a tree of threshold gates.

Yao et al. [YFDL04] (see Chapter VII) show how an IBE system that encrypts to multiple hierarchical identities in a collusion-resistant manner implies a forward secure Hierarchical IBE scheme. They also note how their techniques for resisting collusion attacks are useful in attribute-based encryption. However, the cost of their scheme in terms of computation, private key size, and ciphertext size increases exponentially with the number of attributes. We also note that there has been other work that applied IBE techniques to access control, but did not address our central concern of resisting attacks from colluding users [Sma03,BHS04].

2. Background

We first give formal definitions for the security of Key-Policy Attribute-Based Encryption.

2.1. Definitions

Definition X.1 (Access Structure [Bei96]) Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,P_2,\ldots,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An *access structure* (respectively, *monotone access structure*) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,P_2,\ldots,P_n\}} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are called the *authorized sets*, and the sets not in $\mathbb{A}$ are called the *unauthorized sets*.

In our context, the role of the parties is taken by the attributes. Thus, the access structure $\mathbb{A}$ will contain the authorized sets of attributes. We restrict our attention to monotone access structures. However, it is also possible to (inefficiently) realize general access structures using our techniques by having the **NOT** of an attribute as a separate attribute altogether. Thus, the number of attributes in the system will be doubled. From now on, unless stated otherwise, by an access structure we mean a monotone access structure.

A (Key-Policy) Attribute-Based Encryption scheme consists of four algorithms.

[3]Indeed, in its most general form, every LSSS realizable access structure. We briefly discuss this notion in Section 3.

Setup This is a randomized algorithm that takes no input other than the implicit security parameter. It outputs the public parameters mpk and a master key msk.

Encryption This is a randomized algorithm that takes as input – a message M, a set of attributes γ, and the public parameters mpk. It outputs the ciphertext E.

Key Derivation This is a randomized algorithm that takes as input – an access structure $\mathbb{A}$, the master key msk, and the public parameters mpk. It outputs a decryption key usk.

Decryption This algorithm takes as input – the ciphertext E that was encrypted under the set γ of attributes, the decryption key usk for access structure $\mathbb{A}$, and the public parameters mpk. It outputs the message M if $\gamma \in \mathbb{A}$.

We now discuss the security of an ABE scheme. We define a Selective-Set model for proving the security of an ABE scheme under chosen-plaintext attack. This model can be seen as analogous to the Selective-ID model [CHK03,BCHK06,BB04a] used in IBE schemes [Sha85,BF03,Coc01].

The Selective-Set Game for ABE

INIT The adversary declares the set of attributes, γ, that he wishes to be challenged upon.

SETUP The challenger runs the setup algorithm of the ABE scheme and gives the public parameters to the adversary.

PHASE 1 The adversary is allowed to adaptively issue queries for private keys for many access structures $\mathbb{A}_j$, where $\gamma \notin \mathbb{A}_j$ for all j.

CHALLENGE The adversary submits two equal length messages M_0 and M_1. The challenger flips a random coin b, and encrypts M_b with γ. The ciphertext is passed to the adversary.

PHASE 2 Phase 1 is repeated.

GUESS The adversary outputs a guess b' of b.

Given an ABE scheme $\mathcal{ABE}$ and an adversary A, the *advantage* of A in this game is defined as $\mathbf{Adv}^{\text{Selective-Set}}_{\mathcal{ABE}}(\mathsf{A}) = |\Pr[b' = b] - \frac{1}{2}|$.

We note that the model can easily be extended to handle chosen-ciphertext attacks by allowing for decryption queries in Phase 1 and Phase 2.

Definition X.2 An attribute-based encryption scheme is *secure* in the Selective-Set model of security if all polynomial time adversaries have at most a negligible advantage in the Selective-Set game.

On the other hand, we say that an adversary *breaks* an ABE scheme if its advantage in the Selective-Set game is non-negligible.

3. Construction for Access Trees

We now present the main construction of this chapter. We follow the construction that was first given by Goyal, Pandey, Sahai, and Waters [GPSW06], which we call the $\mathcal{GPSW}$-$\mathcal{ABE}$ scheme. In this construction ciphertexts will be labeled with a set of descriptive attributes. Private keys are identified by a tree access structure in which each interior node of the tree is a threshold gate and the leaves are associated with attributes. (We note that this setting is very expressive. For example, we can represent a tree of **AND** and **OR** gates by using respectively 2-out-of-2 and 1-out-of-2 threshold gates.) A user will be able to decrypt a ciphertext with a given key if and only if there is an assignment of attributes from the ciphertexts to nodes of the tree such that the tree is satisfied.

3.1. Access Trees

Access Tree $\mathcal{T}$ Let $\mathcal{T}$ be a tree representing an access structure. Each non-leaf node of the tree represents a threshold gate, described by its children and a threshold value. If num_x is the number of children of a node x and k_x is its threshold value, then $0 < k_x \leq num_x$. When $k_x = 1$, the threshold gate is an **OR** gate and when $k_x = num_x$, it is an **AND** gate. Each leaf node of the tree simply represents an attribute.

To facilitate working with the access trees, we define a few functions. We denote the parent of a node x by $\text{parent}(x)$. The function $\text{att}(x)$ is defined only if x is a leaf node and denotes the attribute associated with x. The access tree $\mathcal{T}$ also defines an ordering between the children of every node, i.e., the children of a node x are numbered from 1 to num_x. The function $\text{index}(x)$ returns such a number associated with the node x.

Satisfying an Access Tree Let $\mathcal{T}$ be an access tree with root r. Denote by $\mathcal{T}_x$ the subtree of $\mathcal{T}$ rooted at the node x. Hence $\mathcal{T}$ is the same as $\mathcal{T}_r$. If a set of attributes γ satisfies the access tree $\mathcal{T}_x$, we denote it as $\mathcal{T}_x(\gamma) = 1$. We compute $\mathcal{T}_x(\gamma)$ recursively as follows. If x is a non-leaf node, evaluate $\mathcal{T}_{x'}(\gamma)$ for all children x' of node x. $\mathcal{T}_x(\gamma)$ returns 1 if and only if at least k_x children return 1. If x is a leaf node, then $\mathcal{T}_x(\gamma)$ returns 1 if and only if $\text{att}(x) \in \gamma$.

3.2. Main Construction

Let $\mathbb{G}$ be a bilinear group of prime order p and let P be a generator of $\mathbb{G}$. Additionally, let $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_\mathrm{T}$ denote the bilinear map. A security parameter k will determine the size of the groups. We also define the Lagrange coefficient $\Delta_{i,S}$ for $i \in \mathbb{Z}_p$ and a set S of elements in $\mathbb{Z}_p$: $\Delta_{i,S}(x) = \prod_{j \in S, j \neq i} \frac{x-j}{i-j}$. We will associate each attribute with a unique element in $\mathbb{Z}_p^*$. For notational convenience, we assume that all the leaves of the access tree are at the same level ℓ. *We stress that this is only for notational convenience, and that our construction applies to any arbitrary tree structure, which can be chosen differently for each private key.* The construction follows.

Setup Define the universe of attributes $\mathcal{U} = \{1, 2, \ldots, n\}$. Now, for each attribute $i \in \mathcal{U}$, choose a number t_i uniformly at random from $\mathbb{Z}_p$. Finally, choose y uniformly at random in $\mathbb{Z}_p$. Publish the public parameters mpk as: $(T_1 = t_1 P, \ldots, T_{|\mathcal{U}|} = t_{|\mathcal{U}|} P, Y = \hat{e}(P, P)^y)$. The master key msk is: $(t_1, \ldots, t_{|\mathcal{U}|}, y)$.

Encryption(M, γ, mpk) To encrypt a message $M \in \mathbb{G}_\mathrm{T}$ under a set of attributes γ, choose a random value $s \in \mathbb{Z}_p$ and publish the ciphertext as: $E = (\gamma, E' = MY^s, \{E_i = sT_i\}_{i \in \gamma})$.

Key Derivation($\mathcal{T}, msk, mpk$) The algorithm outputs a key that enables the user to decrypt a message encrypted under a set of attributes γ if and only if $\mathcal{T}(\gamma) = 1$. The algorithm proceeds as follows. First choose a polynomial q_x for each non-leaf node x in the tree $\mathcal{T}$. These polynomials are chosen in the following way in a top down manner, starting from the root node r.

For each non-leaf node x in the tree, set the degree d_x of the polynomial q_x to be one less than the threshold of the gate at node x, i.e., $d_x = k_x - 1$. Now for the root node r, set $q_r(0) = y$ and d_r other points of the polynomial q_r randomly to define it completely. For any other node x, set $q_x(0) = q_{\text{parent}(x)}(\text{index}(x))$ and choose d_x other points randomly to define q_x.

Once the polynomials have been decided, for each leaf node x, we give the following secret value to the user: $D_x = \frac{q_{\text{parent}(x)}(i)}{t_i} P$ where $i = \text{att}(x)$. The decryption key $usk = \{D_x\}$ is the set of these secret values.

Decryption($\boldsymbol{E}, \boldsymbol{D}, \boldsymbol{mpk}$) To decrypt the ciphertext $E = (\gamma, E', \{E_i\}_{i\in\gamma})$ with the decryption key usk, proceed as follows. For every satisfied non-leaf node x in the access tree, define S_x to be a set of any $k_x = d_x + 1$ satisfied children of x. Now, if x is a node at level[4] $(\ell - 1)$, define:

$$\begin{aligned} F_x &= \prod_{x'\in S_x} \hat{e}(D_{x'}, E_i)^{\Delta_{i,S'_x}(0)} \quad \text{where } {}_{S'_x=\{\text{att}(x'):x'\in S_x\}}^{i=\text{att}(x')} \\ &= \prod_{x'\in S_x} \hat{e}\left(\frac{q_x(i)}{t_i} P, s\cdot t_i P\right)^{\Delta_{i,S'_x}(0)} = \prod_{x'\in S_x} \hat{e}(P,P)^{s\cdot q_x(i)\cdot\Delta_{i,S'_x}(0)} \\ &= \hat{e}(P,P)^{s\cdot q_x(0)} \quad \text{(using polynomial interpolation)}\ . \end{aligned}$$

Else, for higher levels:

$$\begin{aligned} F_x &= \prod_{x'\in S_x} F_{x'}^{\Delta_{i,S'_x}(0)}, \quad \text{where } {}_{S'_x=\{\text{index}(x'):x'\in S_x\}}^{i=\text{index}(x')} \\ &= \prod_{x'\in S_x} \left(\hat{e}(P,P)^{s\cdot q_{x'}(0)}\right)^{\Delta_{i,S'_x}(0)} \\ &= \prod_{x'\in S_x} \left(\hat{e}(P,P)^{s\cdot q_{\text{parent}(x')}(\text{index}(x'))}\right)^{\Delta_{i,S'_x}(0)} \quad \text{(by construction)} \\ &= \prod_{x'\in S_x} \hat{e}(P,P)^{s\cdot q_x(i)\cdot\Delta_{i,S'_x}(0)} \\ &= \hat{e}(P,P)^{s\cdot q_x(0)} \quad \text{(using polynomial interpolation)}\ . \end{aligned}$$

If $\mathcal{T}(\gamma) = 1$, then using above computations, the user will always be able to compute F_r. Now,

$$E'/F_r = MY^s/\hat{e}(P,P)^{s\cdot q_r(0)} = M\hat{e}(P,P)^{sy}/\hat{e}(P,P)^{sy} = M\ .$$

[4]Note that more generally, if all leaves of the tree are not at level ℓ, this can just be seen as a recursive procedure, with a "base case" for dealing with leaves, and a general case for internal nodes. Again, the simplifying assumption that all leaves are at level ℓ is made here only to simplify notation.

Performance The number of group elements that compose a user's private key grows linearly with the number of leaf nodes in the access tree. The number of group elements in a ciphertext grows linearly with the size of the set we are encrypting under. Finally, the number of group elements in the public parameters grows linearly with the number of attributes in the defined universe.

The number of scalar multiplications in the group $\mathbb{G}$ to encrypt a message under a set γ grows linearly with the size of γ. Let the number of satisfied *leaf* nodes *used* during the decryption procedure be N_l. Similarly, let N_s denote the number of all satisfied nodes, except the root node, used during the decryption procedure. Then the number of pairing computations grows linearly with N_l and the number of exponentiations in $\mathbb{G}_\mathrm{T}$ grows linearly with N_s. The cost of multiplications in $\mathbb{G}_\mathrm{T}$ is negligible compared to exponentiations and pairing computations.

Improvements The following observation shows how to modify our decryption method so that it minimizes the number of pairing computations and exponentiations. First, we find out which leaf nodes we should use in order to minimize the number of pairing computations as follows. For each node x, we define a set $\mathbb{S}_x$ as follows. If x is a leaf node, then $\mathbb{S}_x = \{x\}$. Otherwise, let k be the threshold value of x. From amongst the child nodes of x, choose k nodes $x_1, x_2, \ldots, x_k$ such that $\mathbb{S}_{x_i}$ (for $i = 1, 2, \ldots, k$) are first k sets of the smallest size. Then for each non-leaf node x, $\mathbb{S}_x = \{x' : x' \in \mathbb{S}_{x_i}, i = 1, 2, \ldots, k\}$. The set $\mathbb{S}_r$ corresponding to the root node r contains the leaf nodes that should be used to minimize the number of pairing computations. Next, we notice that in the given decryption algorithm, Lagrange coefficients (Δ_{i,S'_x}) from various levels get multiplied in the exponent in a certain way in $\mathbb{Z}_p$. Thus, instead of exponentiating at each level, for each leaf node $x \in \mathbb{S}_r$, we can keep track of which Lagrange coefficients get multiplied with each other. Using this we can compute the final exponent f_x for each leaf node $x \in \mathbb{S}_r$ by doing multiplication in $\mathbb{Z}_p$. Now F_r is simply $\prod_{x \in \mathbb{S}_r} \hat{e}(D_x, E_{\mathrm{att}(x)})^{f_x}$. This reduces the number of pairing computations and exponentiations to $|\mathbb{S}_r|$. Thus, decryption is dominated by $|\mathbb{S}_r|$ pairing computations.

Large Universes In the construction provided, a group element must be given in the mpk for every attribute in the system. As discussed in [SW05] one can also consider systems in which any string can be used as an attribute.

Linear Secret Sharing Schemes In a linear secret sharing scheme (LSSS) [Bei96] realizing an access structure $\mathbb{A}$, a third party called the dealer holds a secret y and distributes the shares of y to parties such that y can be reconstructed by a linear combination of the shares of any authorized set. Further, an unauthorized set has no information about the secret y. Our construction in this chapter can be generalized to any access structure realizable by a LSSS (see [GPSW06]).

3.3. Proof of Security

We prove that the security of our scheme in the attribute-based Selective-Set model reduces to the hardness of the Decisional BDH assumption.

Theorem X.3 *If an adversary can break the $\mathcal{GPSW}$-$\mathcal{ABE}$ scheme in the* Selective-Set *model, then a simulator can be constructed to succeed in the* Decisional BDH *game with a non-negligible advantage.*

More precisely, given an adversary that has an ϵ advantage in the Selective-Set *game, a simulator can be constructed to have an $\epsilon/2$ advantage in the* Decisional BDH *game.*

PROOF: Suppose that there exists a polynomial-time adversary A that can attack the $\mathcal{GPSW}$-$\mathcal{ABE}$ scheme in the Selective-Set model with advantage ϵ. We show how to build a simulator B that can play the Decisional BDH game with advantage $\epsilon/2$. The simulation proceeds as follows:

We first let the challenger set the groups $\mathbb{G}$ and $\mathbb{G}_{\mathrm{T}}$ with an efficient bilinear map $\hat{e}$ and a generator P for $\mathbb{G}$. The challenger flips a fair binary coin μ outside of B's view. If $\mu = 0$, the challenger sets $(A, B, C, Z) = (aP, bP, cP, \hat{e}(P, P)^{abc})$; otherwise it sets $(A, B, C, Z) = (aP, bP, cP, \hat{e}(P, P)^{z})$ for random a, b, c, z. We assume the universe $\mathcal{U}$ is defined.

INIT The simulator B runs A. A chooses the set of attributes γ it wishes to be challenged upon.

SETUP The simulator sets the parameter $Y = \hat{e}(A, B) = \hat{e}(P, P)^{ab}$. For all $i \in \mathcal{U}$, it sets T_i as follows: if $i \in \gamma$, it chooses a random r_i and sets $T_i = r_i P$ (thus, implicitly $t_i = r_i$); otherwise it chooses a random β_i and sets $T_i = \beta_i B = b\beta_i P$ (thus, implicitly $t_i = b\beta_i$). It then gives the public parameters to A.

PHASE 1 A adaptively makes requests for decryption keys. The access structure associated with each requested key must not be satisfied by γ.

Suppose A makes a request for the secret key for an access structure $\mathcal{T}$ where $\mathcal{T}(\gamma) = 0$. To generate the secret key, B has to fix a polynomial q_x of degree d_x for every non-leaf node in the access tree. Additionally, we define a new scaled down polynomial q'_x for each non-leaf node x such that $q'_x = \frac{1}{b} \cdot q_x$. Now B fixes the polynomials q_x by choosing q'_x as follows.

Set $y = q_r(0) = ab$. Hence, $q'_r(0) = a$. By the definition of $\mathcal{T}$, no more than d_r children of the root will be satisfied. For every satisfied child x of the root, choose a random λ_x and set $q'_r(\mathrm{index}(x)) = \lambda_x$ (or equivalently $q'_x(0) = \lambda_x$). If needed, choose the rest of the points randomly to completely fix the polynomial q'_r. Now to fix the polynomials for a subtree $\mathcal{T}_x$ defined by a child x of the root, proceed as follows:

CASE-I The node x is satisfied.
In this case, $q'_x(0)$ is known (as λ_x). Hence, it is easy to fix the polynomials for the nodes in this subtree as in the ordinary key derivation process. That is, for a node x' in this subtree, $q'_{x'}(0)$ is known (as $q'_{\mathrm{parent}(x')}(\mathrm{index}(x'))$. The rest of the points are chosen randomly if needed.
Following this procedure, we can fix the polynomials for the nodes of this subtree up to the level $\ell - 1$.

CASE-II The node x is not satisfied.
In this case, first compute $q'_x(0)P$ (as $q'_{\mathrm{parent}(x)}(\mathrm{index}(x))P$) by interpolating the parent polynomial in the multiplicative coefficient of P.[5] If x is a node at level $\ell - 1$, do not fix any further points on q'_x; else recursively apply the same strategy as we did for the main tree $\mathcal{T}$ to fix polynomials for the subtree $\mathcal{T}_x$.

[5]Regular interpolation is not possible as only $q'_r(0)P = aP = A$ is known rather than $q'_r(0)$ itself.

Now, for every node x at level $\ell - 1$, we have the following two cases:

CASE-I Node x is satisfied.
In this case, the polynomial q'_x is fixed and completely known. For every child node x' of x, give the decryption key as

$$D_{x'} = \frac{q_x(i)}{t_i} P = \begin{cases} \frac{q'_x(i)}{r_i} B \text{ if } i \in \gamma \\ \frac{q'_x(i)}{\beta_i} P \text{ otherwise} \end{cases} \quad \text{where } i = \mathrm{att}(x') \ .$$

CASE-II Node x is not satisfied.
In this case, only $q'_x(0)$ is fixed with just $q'_x(0)P$ known rather than $q'_x(0)$ itself. Since x is not satisfied, the number of child nodes x' of x such that $\mathrm{att}(x') \in \gamma$ is no more than d_x. For all such nodes x', set $q'_x(\mathrm{att}(x')) = \lambda_{x'}$ where $\lambda_{x'}$ is chosen randomly. If required, choose the rest of the points randomly to completely fix q'_x.

Now the secret key for a node x', with $i = \mathrm{att}(x')$, is given as follows:

(a) If $i \in \gamma$, $D_{x'} = \frac{q_x(i)}{t_i} P = \frac{\lambda_{x'}}{r_i} bP$.
(b) If $i \notin \gamma$, compute $q'_x(i)P$ by interpolating q'_x in the multiplicative coefficient of P. Now, $D_{x'} = \frac{q_x(i)}{t_i} P = \frac{q'_x(i)}{\beta_i} P$.

Therefore, the simulator is able to construct a private key for the access structure $\mathcal{T}$. Furthermore, the distribution of the private key for $\mathcal{T}$ is identical to that of the original scheme.

CHALLENGE The adversary A will submit two challenge messages M_0 and M_1 to the simulator. The simulator flips a fair binary coin ν and returns an encryption of M_ν. The ciphertext is output as:

$$E = (\gamma, E' = M_\nu Z, \{E_i = r_i C\}_{i \in \gamma}) \ .$$

If $\mu = 0$ then $Z = \hat{e}(P, P)^{abc}$. If we let $s = c$, then we have $Y^s = (\hat{e}(P, P)^{ab})^c = \hat{e}(P, P)^{abc}$, and $E_i = c(r_i P) = r_i C$. Therefore, the ciphertext is a valid random encryption of message M_ν.

Otherwise, if $\mu = 1$, then $Z = \hat{e}(P, P)^z$. We then have $E' = M_\nu \hat{e}(P, P)^z$. Since z is random, E' will be a random element of $\mathbb{G}_T$ from the adversary's view and the message contains no information about M_ν.

PHASE 2 The simulator acts exactly as it did in Phase 1.

GUESS A will submit a guess ν' of ν. If $\nu' = \nu$ the simulator will output $\mu' = 0$ to indicate that it was given a valid BDH-tuple; otherwise it will output $\mu' = 1$ to indicate that it was given a random 4-tuple.

As shown in the construction the simulator's generation of public parameters and private keys is identical to that of the actual scheme.

In the case where $\mu = 1$ the adversary gains no information about ν. Therefore, $\Pr[\nu \neq \nu' : \mu = 1] = \frac{1}{2}$. Since the simulator guesses $\mu' = 1$ when $\nu \neq \nu'$, $\Pr[\mu' = \mu : \mu = 1] = \frac{1}{2}$.

If $\mu = 0$ then the adversary sees an encryption of M_ν. The adversary's advantage in this situation is ϵ by definition. Therefore, $\Pr[\nu = \nu' : \mu = 0] = \frac{1}{2} + \epsilon$. Since the simulator guesses $\mu' = 0$ when $\nu = \nu'$, $\Pr[\mu' = \mu : \mu = 0] = \frac{1}{2} + \epsilon$.

The overall advantage of the simulator in the Decisional BDH game is

$$\frac{1}{2}\Pr\left[\mu' = \mu : \mu = 0\right] + \frac{1}{2}\Pr\left[\mu' = \mu : \mu = 1\right] - \frac{1}{2} = \frac{1}{2}\left(\frac{1}{2} + \epsilon\right) + \frac{1}{2}\left(\frac{1}{2}\right) - \frac{1}{2} = \frac{\epsilon}{2} \ .$$

□

Chosen-Ciphertext Security Our security definitions and proofs have been in the chosen-plaintext model. Similar to [SW05], we notice that the construction can be extended to the chosen-ciphertext model by applying the technique of using simulation-sound NIZK proofs to achieve chosen-ciphertext security [Sah99].

4. Audit Log Application

An important application of KP-ABE deals with secure forensic analysis: one of the most important needs for electronic forensic analysis is an "audit log" containing a detailed account of all activity on the system or network to be protected. Such audit logs, however, raise significant security concerns: a comprehensive audit log would become a prized target for enemy capture. Merely encrypting the audit log is not sufficient, since then any party who needs to legitimately access the audit log contents (for instance a forensic analyst) would require the secret key—thereby giving this single analyst access to essentially all secret information on the network. Such problematic security issues arise in nearly every secure system, and particularly in large-scale networked systems such as the Global Information Grid, where diverse secret, top secret, and highly classified information will need to appear intermingled in distributed audit logs.

A KP-ABE system provides an attractive solution to the audit log problem. Audit log entries could be annotated with attributes such as the name of the user, the date and time of the user action, and the type of data modified or accessed by the user action. Then, a forensic analyst charged with some investigation would be issued a secret key associated with a particular "access structure", which would correspond to the key allowing for a particular kind of encrypted search. Such a key, for example, would only open audit log records whose attributes satisfied the condition that "the user name is Bob **OR** (the date is between October 4, 2005 and October 7, 2005 **AND** the data accessed pertained to naval operations off the coast of North Korea)". The system would provide the guarantee that even if multiple rogue analysts collude to try to extract unauthorized information from the audit log, they will fail.

A more concrete example would be the hypothetical application of an ABE system to the existing ArmyCERT program, which uses NetFlow logs [Net]. Basically, an entry is created for every flow (e.g. TCP connection), indexed by seven attributes: source IP address, destination IP address, L3 protocol type, source port, destination port, ToS byte (DSCP), and input logical interface (ifIndex). These aspects of every flow are in the clear, and the payload can be encrypted using an ABE system with these fields as attributes.

Note that in the scheme we presented, we would need to assume that the attributes associated with audit log entries would be available to all analysts.[6] This may present a problem in highly secret environments where even attributes themselves would need to be kept hidden from analysts. In Section 6, we discuss the important open problem of constructing KP-ABE systems where attributes associated with ciphertexts remain secret.

5. Application to Broadcast Encryption: Targeted Broadcast

We describe a new broadcast scenario that we call *targeted broadcast*. Consider the following setting.

- A broadcaster broadcasts a sequence of different items, each one labeled with a set of attributes describing the item. For instance, a television broadcaster might broadcast an episode of the show "ER", and label this item with attributes such as the name of the program ("ER"), the genre ("DRAMA"), the season, the episode number, the year, month, and date of original broadcast, the current year, month, and date, the name of the director, and the name of the producing company.
- Each user is subscribed to a different "package". The user package describes an access policy, which along with the set of attributes describing any particular item being broadcast, determine whether or not the user should be able to access the item. For example, a television user may want to subscribe to a package that allows him view episodes of "ER" from either the final season or the first two seasons. This policy could be encoded as ("ER" **AND** ("SEASON:15" **OR** "SEASON:1" **OR** "SEASON:2")).

The essential idea of Targeted Broadcast is to enjoy the economies-of-scale offered by a broadcast channel, while still being able to deliver programming targeted at the needs or wishes of individual users. The growing acceptability of such a model can be seen by the rising popularity of DVR systems such as TiVo, which allow users to easily record only the programming they want in order to watch it later. In the case of television, taking the approach that we envision here would allow for much more flexibility than just allowing users to select what channels they like.

A KP-ABE system naturally offers a targeted broadcast system. A new symmetric key would be chosen and used to encrypt each item being broadcast, and then the KP-ABE system would be used to encrypt the symmetric key with the attributes associated with the item being broadcast. The KP-ABE system would precisely allow the flexibility we envision in issuing private keys for the unique needs of each user.

It is worth mentioning that handling such a situation with the best known broadcast encryption schemes [BGW05,HS02] (which allow encrypting to an arbitrary subset of users) is quite inefficient in comparison. The efficiency of such systems is dependent on the size of the authorized user set or the number of users in the system [BGW05,HS02], and would also require the broadcaster to refer to its database of user authorizations each time a different item is to be encrypted for broadcast. In our scheme, the encryption of an item would depend only on the properties of that item. The broadcaster could in principle

[6]We observe that this does not mean that the attributes need be "public." The system's ciphertexts could be re-encrypted with a key that corresponds to the general clearance level of all analysts.

even forget about the levels of access granted to each user after preparing a private key for the user.

6. Discussion and Extensions

We discuss various extensions to the ABE system and open problems.

Ciphertext-Policy Attribute-Based Encryption In this chapter, we considered the setting where ciphertexts are associated with sets of attributes, whereas user secret keys are associated with policies. As we have discussed, this setting has a number of natural applications. Another possibility is to have the reverse situation: user keys are associated with sets of attributes, whereas ciphertexts are associated with policies. We call such systems Ciphertext-Policy Attribute-Based Encryption (CP-ABE) systems. We note that the construction of Sahai and Waters [SW05] was most naturally considered in this framework. CP-ABE systems that allow for complex policies (like those considered here) have a number of applications. An important example is a kind of sophisticated Broadcast Encryption, where users are described by (and therefore associated with) various attributes. Then, one could create a ciphertext that can be opened only if the attributes of a user match a policy. For instance, in a military setting, one could broadcast a message that is meant to be read only by users who have a rank of Lieutenant or higher, and who were deployed in South Korea in the year 2005. Several works (see [BSW07,CN07,GJPS08,Wat08]) explore the problem of constructing these CP-ABE systems.

Non-monotonic Access Structures Another extension one might consider is the ability to create policies that *prohibit* a user from decrypting certain ciphertexts. Situations where conflicts of interest occur would be a natural example that necessitates restrictive policies. Take as an example an annual university-wide peer-review process where each department will be critiqued by a panel of faculty from other departments. In a KP-ABE system, the reviews could be encrypted and labeled with attributes such as "YEAR=2008", "DEPT-REVIEW", and "DEPT=HISTORY". We want to ensure that a panelist, Bob, from the biology department will not have the access rights to comments written about his own department, so the correct policy to ascribe to Bob's key is "YEAR=2008" **AND** "DEPT-REVIEW" **AND** (**NOT** "DEPT=BIOLOGY"). The construction of such a scheme is given by Ostrovsky, Sahai, and Waters [OSW07], which allows for policies that can represent any *non-monotonic* access formula over the attributes.

Attribute Hiding Our current constructions do not hide the set of attributes under which the data is encrypted. However, if it were possible to hide the attributes, then viewing attributes as keywords in such a system would lead to the first general keyword-based search on encrypted data [BDOP04]. A search query could potentially be any monotone boolean formula of any number of keywords. Some progress towards this goal has been made in [BW07,KSW08], in a notion generalizing ABE called Functional Encryption (or Predicate Encryption). In Functional Encryption, secret keys correspond to functions that can be used to decrypt a ciphertext (with attributes) if and only if the attributes satisfy the function. The works [BW07,KSW08] construct schemes that answer this problem for certain classes of functions. We leave the problem of hiding the set of attributes in general as open.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-169

Chapter XI

On Generic Groups and Related Bilinear Problems

David LUBICZ and Thomas SIRVENT

DGA-CÉLAR and Université de Rennes 1, France

Abstract. Groups with pairing are now considered as standard building blocks for cryptographic primitives. The security of schemes based on such groups relies on hypotheses related to the discrete logarithm problem. As these hypotheses are not proved, one would like to have some positive security argument for them. It is usual to assess their security in the so called generic group model introduced by Nechaev and Shoup. Over the time, this model has been extended in different directions to cover new features.

The relevance of this model is nevertheless subject to criticisms: in particular, the fact that the answer to any fresh query is a random bit string is not what one expects from a usual group law.

In this chapter, we first present the original model of Nechaev and Shoup as well as some classical extensions, with a focus on ideas rather than formal correctness. Then, we develop rigorously a generic group model with pairing which generalizes all models seen so far in the literature. We provide a general framework in order to prove difficulty assumptions in this setting. In order to improve the realism of this model, we introduce the notion of pseudo-random families of groups. We show how to reduce the security of a problem in such a family to the security of the same problem in the generic group model and to the security of an underlying strong pseudo-random family of permutations.

Keywords. Generic groups, bilinear Diffie-Hellman assumptions, pairings, discrete logarithm, pseudo-random permutations

The discrete logarithm problem is a general way to build trapdoor functions for asymmetric cryptographic protocols. It can be used as long as one can find a family of cyclic groups satisfying certain properties:

- one would like to be able to compute quickly the group law and, as a consequence, the exponential map using for instance the square and multiply algorithm,
- it is also necessary for the discrete logarithm problem to be intractable in the considered family of groups.

There is no family of groups for which this last property is proved. Nevertheless, on the practical side, there is no known algorithm which performs better than in exponential time in the family of rational points of elliptic curves defined over a finite field.

Some widely used cryptosystems, for instance Diffie-Hellman key agreement and its derivative [DH76,MTI86], El Gamal encryption [ElG85] or DSA and related signature schemes [DSS94,ElG85,Sch91], actually implement trapdoor functions such that their security relies on the intractability of a discrete logarithm related problem. This situation is not really satisfying, and one would expect at least some positive argument for the discrete logarithm problem to be intractable.

Actually there is a class of algorithms for solving discrete logarithm and related problems which do not rely on any particular property of the group used. The classical Pohlig-Hellman algorithm [PH78] and the Pollard's Rho algorithm [Pol78] are for instance in this class. In [Sho97], Shoup calls this class of algorithms "generic algorithms" because they work with any family of groups. Following Nechaev [Nec94], Shoup introduces moreover a systematic way to study the computational power of these algorithms. Of course, not all algorithms solving a discrete logarithm related problem fall in the class of generic algorithms. For instance, the index-calculus method in $\mathbb{Z}_p^*$, where p is a prime number, is not a generic algorithm.

Generic Algorithms The generic algorithms are roughly speaking automatons with memory which can only perform group operations. Nechaev and Shoup proved that the fastest generic algorithms for solving the discrete logarithm problem or the Diffie-Hellman problem are in exponential time complexity with respect to the group size. In a modern formulation, a generic algorithm is a Turing machine dealing with bit strings representing the group elements by the way of an encoding function. A generic algorithm is unable to compute group operations. These group operations are provided by oracles defined in the following way:

- all queries are stored in a list;
- when a new query is similar to a former one, the oracles return the same answer;
- when a new query is fresh, the oracles return a randomly chosen bit string.

At the end of the game, the challenger verifies that the simulation provided by these oracles is coherent with discrete logarithm values drawn randomly for each initial bit string submitted to the generic algorithm and by checking that no collision occurs.

This general framework has subsequently been improved in order to take into account new group properties used in protocols such as pairings [BB04c,YW05]. As a matter of fact, the generic group model is now a standard tool for the proof of cryptographic protocols. It is used either directly to compute running time lower bounds for breaking a given protocol [Bro02,BBG05], or more generally in order to assess the security of a new computational or decisional hypothesis upon which is based the security of a protocol [Jou04,BB04c].

Limitations and Criticisms Because of the widespread use of the generic group model, a natural and important question is the relevance of this model. As pointed out in [AF07], the original model presented by Shoup does not take into account all possible computations of a full-fledged Turing Machine with access to group oracles. For instance, when receiving a bit string from an oracle, an algorithm may flip a certain bit and use the resulting bit string to submit a new query. Intuitively, this type of query will not improve the probability of success of the algorithm but still should be taken into account, in an extension of Shoup's model.

Moreover, the generic group model has been the target of numerous criticisms [Den02,SPMLS02,NSS04] some of which have been turned down [KM07]. For instance, because the oracles answer with random bit strings to fresh queries, the generic group model is often compared with the random oracle model. Nonetheless, such random behavior of the generic group model is not what one expects to model real groups, like it was pointed out in [KM07].

The aim of this chapter is to present an overview of the generic group model following [Nec94,Sho97] as well as some classical and less classical extensions.

Our approach, is slightly different from the original one of [Sho97]. In the usual generic group model, the answer of the group law oracles is just a random sequence of bits for any fresh query and it does not take into account any underlying algebraic structure. The group structure is only used at the end of the game in order to check the presence of a collision. In contrast, in our model, the group law is randomly chosen at the beginning of the game. We study the set of group laws that remain indistinguishable from this specific law during the whole game, and we compute a probability based on this set. Using this formalism, we prove that any problem for which there does not exist a general formula to compute the solution from the input has exponential time complexity in the generic group model (see Theorem XI.27). This result shows that the generic group model is a crude but nonetheless important evaluation of the security of a problem.

In order to improve the realism of the model, we introduce the notion of pseudo-random groups: a pseudo-random group is a group where the group laws are provided by the way of a strong pseudo-random permutation. As the family of random permutations is a particular case of strong pseudo-random permutations, it is easy to see that our model is a generalization of the generic group model. The main result of this chapter shows that every result that can be proved in the generic group model can also be proved in the pseudo-random group model. In other words, we can consider families of groups where the group laws, provided by oracles, are drawn following a pseudo-random distribution. The security of a hypothesis over such a family can be reduced to the robustness of this family as a pseudo-random family of groups, through the security of this same hypothesis in the generic model. As a matter of fact, because of the reduction that we prove, the pseudo-random group model does not bring new security insight. Still it constitutes an improvement because it is more realistic than the usual generic group model.

1. Generic Algorithms: Shoup's Model

Let $B(n)$ be the set of binary representation of integers in $\{0, \ldots, n-1\}$. An encoding function, ξ is a bijective map from the additive group $\mathbb{Z}_n$ into $B(n)$. A generic algorithm A is an algorithm that behaves as follows. It takes as input an encoding list $(\xi(x_1), \ldots, \xi(x_k))$, for $k \in \mathbb{N}^*$ and $x_i \in \mathbb{Z}_n$. From time to time A can submit a query to an oracle by specifying two indices i, j of the encoding list and a bit sign. The oracle computes $\xi(x_i \pm x_j)$ according to the bit sign and this bit string is appended to the encoding list. The output of A is a bit string which is denoted by $\mathsf{A}(\xi; x_1, \ldots, x_k)$. We assess the running time of a generic algorithm by the number of bit operations and the number of calls to the oracle. The Pohlig-Hellman algorithm [PH78] is an example of a generic algorithm to solve the discrete logarithm problem in a group of prime order p in time $O(\ln(p)\sqrt{p})$. The Pollard's discrete logarithm algorithm [Pol78] is also a generic algorithm with the same expected running time as the Pohlig-Hellman algorithm.

In this model it is possible to show that the discrete logarithm problem is difficult:

Theorem XI.1 ([Sho97], Theorem 1) *Let n be a positive integer whose largest prime divisor is p. Let A be a generic algorithm for $\mathbb{Z}_n$ on $B(n)$ that makes at most m queries. If $x \in \mathbb{Z}_n$ and an encoding function ξ are chosen at random, then the probability that $A(\xi; 1, x) = x$ is $O(m^2/p)$.*

We sketch the proof of Theorem XI.1. For the sake of simplicity, we suppose that n is prime. The idea of the proof is to consider a modified game which almost always looks like the real game from the point of view of the generic algorithm. In this modified game, the group oracle always answers by random bit strings to fresh queries: the group oracle gives then no information to the algorithm.

We describe the modified game. Let X be an indeterminate. In order to keep track of the queries of the algorithm we build a list of affine polynomials $F_1, \dots, F_k$, together with a list $\xi_1, \dots, \xi_k$ of elements of $B(n)$. At the beginning of the game $k = 2$, $F_1 = 1$, $F_2 = X$, ξ_1 and ξ_2 are chosen at random, such that $\xi_1 \neq \xi_2$. When the oracle is given two indices i and j and a sign, we append to the list $F_{k+1} = F_i \pm F_j$. If $F_{k+1} = F_l$ for some $l \leqslant k$, we define $\xi_{k+1} = \xi_l$ otherwise we draw ξ_{k+1} at random in $B(n) \setminus \{\xi_1, \dots, \xi_k\}$.

When the algorithm outputs some $x' \in \mathbb{Z}_n$, we choose a random $x \in \mathbb{Z}_n$ and say that the algorithm wins the game if $x = x'$ or if there exist $i \neq j$ such that $F_i(x) = F_j(x)$. It is easy to see that the probability that the generic algorithm wins the game is the probability that there exists two indices i and j such that $(F_i - F_j)(x) = 0$ which is bounded using [Sch80, Lemma 1] by $O(m^2/p)$.

In order to finish the proof, we only have to remark that the modified game differ from the real game only when the generic algorithm wins. As a consequence, the probability that the algorithm gives the correct answer is bounded by the probability that the algorithm wins the modified game and we are done.

Using the same technique, the Diffie-Hellman problem (and its decisional version as well) can be proved to be difficult:

Theorem XI.2 ([Sho97], Theorem 3) *Let n be a positive integer whose largest prime divisor is p. Let A be a generic algorithm for $\mathbb{Z}_n$ on $B(n)$ that makes at most m oracle queries. If $x, y \in \mathbb{Z}_n$ and an encoding function ξ are chosen at random, then the probability that $A(\xi; 1, x, y) = \xi(x\,y)$ is $O(m^2/p)$.*

This treatment of the generic group model has been extended in [BB04c] to take pairings into account. This can be done in the following way: consider two encoding functions ξ and ξ' from $\mathbb{Z}_n$ into $B(n)$. A generic algorithm for group with pairing A is an algorithm the behaves as follows. It takes as input two encoding lists $(\xi(x_1), \dots, \xi(x_{k_1}))$ and $(\xi'(y_1), \dots, \xi'(y_{k_2}))$. The algorithm has access to two oracles which receives two indices i and j, a bit sign, and reply respectively $\xi(x_i \pm x_j)$ and $\xi'(y_i \pm y_j)$ according to the bit sign. A third oracle, receives two indices and reply $\xi'(x_i\,x_j)$. The output of A is a bit string denoted by $\mathsf{A}(\xi, \xi'; x_1, \dots, x_{k_1}; y_1, \dots, y_{k_2})$. Using some adaptation of the arguments of Theorem XI.1, the authors of [BB04c], prove that the better generic algorithm solving the q-$\mathcal{SDH}$ problem has an exponential-time complexity.

2. Full Formalization of Generic Families of Groups

2.1. Representations of Cyclic Groups

To define and study a group-based problem, we need a representation of a group as an object understandable by a Turing machine, not as a mathematical object. The study of such problem is moreover based on a family of groups (with different properties or different sizes): we define therefore a family of representations of cyclic groups.

All elements of groups of a family are represented by bit-strings. Each group of the family is defined by a parameter α. The set L_α contains the bit-strings corresponding to all elements of this group. Its size is given by $c(\alpha)$. The laws are defined by $+_\alpha$ and $-_\alpha$. The zero element 0_α and a generator g_α are moreover given. In the following definition, we require that $(c(\alpha))_{\alpha\in\Omega}$ is not upper-bounded. This means that the family contains groups of arbitrarily large orders so that we can obtain an asymptotic complexity for an algorithm using this family. We require moreover that the elements of a group in the family have short binary representations.

Definition XI.3 Let L be a subset of $\{0,1\}^*$. A family of representations of cyclic groups over this language L is the data of:

- a set Ω of parameters (bit-strings), together with a function $c : \Omega \rightarrow \mathbb{N}^*$ computable in polynomial time, such that $\forall N \in \mathbb{N}, \exists\, \alpha \in \Omega \;:\; c(\alpha) \geqslant N$,
- for each $\alpha \in \Omega$, a finite subset L_α of L, together with two laws, $+_\alpha$ and $-_\alpha$, computable in polynomial time, and two elements in L_α: 0_α and g_α. We require L_α to be a cyclic group of order $c(\alpha)$ with group law $+_\alpha$, inverse law $-_\alpha$, and such that 0_α and g_α are respectively the zero element and a generator of this group. We suppose moreover that $\max\{|x|, x \in L_\alpha\} = O(\log_2(c(\alpha)))$.

Example XI.4 Consider the family of all elliptic curves of prime order over fields $\mathbb{F}_p$ where p is a prime number larger than 3. An elliptic curve E in this family can be represented by a parameter $\alpha = (p, a_1, a_2, a_3, a_4, a_6, P)$ defining its Weierstrass equation $y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$ over $\mathbb{F}_p$ and a generator P of $E(\mathbb{F}_p)$. The group laws and the zero element are directly deduced from such parameter. The function c is based on the polynomial-time algorithm due to Schoof [Sch98].

2.2. Generic Family of Cyclic Groups

The cyclic groups are classified up to isomorphisms by their order, i.e. a cyclic group $\mathbb{G}$ is of order $n \in \mathbb{N}^*$ if and only if it is isomorphic to the additive group $\mathbb{Z}_n$. A generator g of $\mathbb{G}$ defines an unique isomorphism such that its image in $\mathbb{Z}_n$ is 1. This isomorphism is called the discrete logarithm map of $\mathbb{G}$ in base g, and noted $\log_g$.

We consider now a set A of n elements. Let $\mathcal{F}(A)$ be the set of all bijective applications from $\mathbb{Z}_n$ into A. Any bijective application $f \in \mathcal{F}$ provides A with a structure of group, where the zero element is $f(0)$, a generator is $f(1)$ and the group and inverse laws ($+_f$ and $-_f$) are given by:

$$\forall (x,y) \in A^2,\ x +_f y = f(f^{-1}(x) + f^{-1}(y)) \text{ and } -_f x = f(-f^{-1}(x)). \qquad (1)$$

We denote by A_f the set A together with the group structure given by f. For any subset $S \subset \mathcal{F}(A)$, the family $\{A_f, f \in S\}$ is denoted by A_S. The family $A_{\mathcal{F}(A)}$ contains all possible group structures over A.

Definition XI.5 Let $B(n)$ be the set of binary representations of integers in $\{0, \ldots, n-1\}$. The union over $n \in \mathbb{N}^*$ of the families $B(n)_{\mathcal{F}(B(n))}$ is called the generic family of cyclic groups.

To consider this generic family of cyclic groups as a family of representations of cyclic groups over $\{0,1\}^*$, we use the following description:

- a parameter is a 3-uple $(n, 0_f, g_f) \in \mathbb{N} \times B(n)^2$ such that $0_f \neq g_f$,
- the function c is given by: $(n, 0_f, g_f) \mapsto n$,
- elements of a group of order n are represented in $B(n)$.

The group and inverse laws, $+_f$ and $-_f$, are built using rules (1) from a function f randomly chosen in $\mathcal{F}(B(n))$ such that $f(0) = 0_f$ and $f(1) = g_f$. The only difference with the definition of a family of representations of cyclic groups is that these laws cannot be deduced from the parameter: these group laws are given through oracles.

Remark XI.6 Using the square and multiply algorithm, the application $f : \mathbb{Z}_n \to B(n)_f$ can be computed in at most $2\log(n)$ group operations $+_f$ from the knowledge of $f(1)$. In the opposite direction, the application f^{-1} is exactly $\log_{f(1)}$. Thus, when the computation of the discrete logarithm is supposed to be hard, an algorithm cannot efficiently compute the group law $+_f$ using rules (1).

2.3. Representations of Cyclic Groups with Pairing

We introduce now non-degenerate bilinear applications between families of representations of cyclic groups, to define a family of representations of cyclic groups with pairing. We consider here only symmetric pairings, but all results obtained later are still valid for different types of pairings.

Definition XI.7 Let L and M be two subsets of $\{0,1\}^*$. A family of representations of cyclic groups with pairing, over L and M, is the data of:

- two families of representations of cyclic groups, $(\Gamma, (L_\gamma)_{\gamma\in\Gamma})$ and $(\Delta, (M_\delta)_{\delta\in\Delta})$,
- a parameter space $\Omega \subset \Gamma \times \Delta$,
- for all $\alpha = (\gamma, \delta) \in \Omega$, a bilinear map $\hat{e}_\alpha$ from $L_\gamma \times L_\gamma$ into M_δ, computable in polynomial time, such that $\hat{e}_\alpha(g_\gamma, g_\gamma) = g_\delta$.

Such family is denoted by: $(\Omega, (L_\gamma)_{\gamma\in\Gamma}, (M_\delta)_{\delta\in\Delta}, (\hat{e}_\alpha)_{\alpha\in\Omega})$.

Example XI.8 Following Example XI.4, we consider now an elliptic curve E defined over a finite field $\mathbb{F}_q$ of characteristic p. Let $l \geqslant 2$ be an integer prime to p, let k be the smallest integer such that l divides $q^k - 1$. $\mathbb{F}_{q^k}$ is the smallest extension of $\mathbb{F}_q$ containing the l^{th} roots of unity. The Weil pairing $e_W : E[l] \times E[l] \to \mathbb{F}_{q^k}$ (see [Sil92]) is a skew symmetric form on $E[l]$. Considering a maximal isotropic subgroup $G(E[l])$ of $E[l]$ and an isomorphism with its dual, one obtains a modified bilinear non trivial pairing $\tilde{e} : G(E[l]) \times G(E[l]) \to \mathbb{F}_{q^k}$, such that $\tilde{e}(P, P)$ is a generator of the group of l^{th} roots of unity in $\mathbb{F}_{q^k}$, when P is a generator of $G(E[l])$. This pairing can be computed in probabilistic polynomial time using Miller's algorithm [CF05]. A family of representations of cyclic groups with pairing can be deduced from this construction.

2.4. Generic Family of Cyclic Groups with Pairing

Let $\mathbb{G}$ be a cyclic group of order n. We denote by $\hat{\mathbb{G}}$ the dual of $\mathbb{G}$, i.e. the group of $\mathbb{Z}_n$-linear maps from $\mathbb{G}$ into $\mathbb{Z}_n$. The bilinear map $\hat{e}_c : (x, v) \in \mathbb{G} \times \hat{\mathbb{G}} \mapsto v(x) \in \mathbb{Z}_n$ is called the canonical pairing of $\mathbb{G}$.

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_\mathrm{T}$ be three cyclic groups of order n. A $\mathbb{Z}_n$-bilinear map $\hat{e} : G_1 \times G_2 \to \mathbb{G}_\mathrm{T}$ is a perfect pairing if it is isomorphic to the canonical pairing of $\mathbb{G}_1$, i.e. there exist three isomorphisms $m_1 : \mathbb{G}_1 \to \mathbb{G}_1$, $m_2 : \mathbb{G}_2 \to \hat{\mathbb{G}}_1$ and $m_T : \mathbb{Z}_n \to \mathbb{G}_\mathrm{T}$ such that $\forall (x, y) \in \mathbb{G}_1 \times \mathbb{G}_2$, $\hat{e}(x, y) = m_T(\hat{e}_c(m_1(x), m_2(y)))$.

Let A be a set of n elements. Let $(f, h) \in \mathcal{F}(A)^2$, we would like to describe all possible perfect pairings from $A_f \times A_f$ into A_h. By definition, such a pairing is deduced from three isomorphisms: m_1, m_2 and m_T.

$$\begin{tikzcd}
A_f \times A_f \arrow[r, "\hat{e}"] \arrow[d, "m_1"', shift right=3ex] \arrow[d, "m_2", shift left=3ex] & A_h \\
A_f \times \hat{A}_f \arrow[r, "\hat{e}_c"'] & \mathbb{Z}_n \arrow[u, "m_T"']
\end{tikzcd}$$

By bilinearity, the perfect pairing $\hat{e}$ is in fact uniquely determined by the value of $\hat{e}(f(1), f(1))$. For all $(f, h) \in \mathcal{F}(A)^2$, there exists thus a unique perfect pairing $\hat{e}$ such that $\hat{e}(f(1), f(1)) = h(1)$. This perfect pairing is in fact defined by:

$$\forall (x, y) \in A^2,\ \hat{e}_{f,h}(x, y) = h(f^{-1}(x) \,.\, f^{-1}(y)) \ . \tag{2}$$

For any subset S of $\mathcal{F}(A)$, let $\mathcal{P}(A, S)$ be a family of two representations of groups with pairings parametrized by the set $\{(f, h) \in S^2\}$. From a pair $(f, h) \in S^2$, we deduce the group structures A_f and A_h following rules (1). Moreover, we consider the perfect pairing $\hat{e}_{f,h}$ from $A_f \times A_f$ into A_h defined by rule (2). The family $\mathcal{P}(A, \mathcal{F}(A))$ contains all pairing structures from $A \times A$ into A.

Definition XI.9 Let $B(n)$ be the set of binary representations of integers in $\{0, \dots, n - 1\}$. The union over $n \in \mathbb{N}^*$ of the families $\mathcal{P}(B(n), \mathcal{F}(B(n)))$ is called the generic family of cyclic groups with pairing.

To consider this generic family of cyclic groups with pairing as a family of representations of cyclic groups with pairing over $\{0,1\}^*$ and $\{0,1\}^*$, we use the following description:

- the generic family of cyclic groups is used twice as a family of representations of cyclic groups (see Section 2.2),
- the set Ω is $\{((n, 0_f, g_f), (n', 0_h, g_h)) \;:\; n = n'\}$.

All group laws ($+_f$, $-_f$, $+_h$ and $-_h$) are built using rules (1) from two functions f and h randomly chosen in $\mathcal{F}(B(n))$ such that $f(0) = 0_f$, $f(1) = g_f$, $h(0) = 0_h$ and $h(1) = g_h$. The pairing law is deduced from these functions using rule (2). Like previously (Section 2.2) ,the only difference with the definition of a family of representations of cyclic groups with pairing is that all laws cannot be deduced from the parameter: they are given through oracles.

Remark XI.10 Like previously (Remark XI.6), when the discrete logarithm problem is assumed to be hard in the groups A_f and A_h, the group laws and the pairing law cannot be efficiently computable using rules (1) and (2).

2.5. Standard Problems

We state now some standard problems for families of representations of cyclic groups, and families of representations of cyclic groups with pairing:

Definition XI.11 Let $(\Omega, (L_\alpha)_{\alpha\in\Omega})$ be a family of representations of cyclic groups. In this family,

- an algorithm solving the discrete logarithm ($\mathcal{DL}$) problem computes $\log_{g_\alpha}(x)$ in the group L_α from the inputs $\alpha \in \Omega$, $x \in L_\alpha$;
- an algorithm solving the Diffie-Hellman ($\mathcal{DH}$) problem computes $\log_{g_\alpha}(x) \,.\, y$ in the group L_α from the inputs $\alpha \in \Omega$, $(x, y) \in (L_\alpha)^2$;
- an algorithm solving the decisional Diffie-Hellman ($\mathcal{DDH}$) problem decides if z is $\log_{g_\alpha}(x) \,.\, y$ in the group L_α from the inputs $\alpha \in \Omega$, $(x, y, z) \in (L_\alpha)^3$.

Definition XI.12 Let $(\Omega, (L_\gamma)_{\gamma\in\Gamma}, (M_\delta)_{\delta\in\Delta}, (\hat{e}_\alpha)_{\alpha\in\Omega})$ be a family of representations of cyclic groups with pairing. In this family,

- an algorithm solving the bilinear Diffie-Hellman ($\mathcal{BDH}$) problem computes the element $\log_{g_\gamma}(w) \,.\, \hat{e}_{(\gamma,\delta)}(x, y)$ in the group M_δ from the inputs $(\gamma, \delta) \in \Omega$, $(w, x, y) \in (L_\gamma)^3$;
- an algorithm solving the decisional bilinear Diffie-Hellman ($\mathcal{DBDH}$) problem decides if z is $\log_{g_\gamma}(w) \,.\, \hat{e}_{(\gamma,\delta)}(x, y)$ in the group M_δ from the inputs $(\gamma, \delta) \in \Omega$, $(w, x, y) \in (L_\gamma)^3$, $z \in M_\delta$.

We can extend these definitions to the generic family of cyclic groups (Definition XI.11) and to the generic family of cyclic groups with pairing (Definition XI.12). An algorithm solving one of these problems in a generic family has the same input and output. The only restriction is that it must use oracles to access the laws.

3. Complexity Analysis

From now on, we focus on the generic family of cyclic groups with pairing. In order to express time and space complexity, we choose the computational model of Turing Machines that have access to some oracles (see [Pap94] pp. 36). We define the time cost of a call to an oracle as one unit of time.

3.1. A General Framework

In this section, we present a general framework in order to assess the difficulty of a problem in the generic family of cyclic groups with pairing. An algorithm A solving a problem in this family is provided at the beginning with the following inputs:

- $n \in \mathbb{N}^*$, $(0_f, g_f, 0_h, g_h) \in (B(n))^4$,
- a r_0-uple $(x_1, \ldots, x_{r_0}) \in (B(n))^{r_0}$ corresponding to parameters in $B(n)_f$,
- a s_0-uple $(y_1, \ldots, y_{s_0}) \in (B(n))^{s_0}$ corresponding to parameters in $B(n)_h$.

Moreover, A has access to oracles computing the group laws $+_f$ and $+_h$, the inverse laws $-_f$ and $-_h$, and the pairing $\hat{e}_{f,h}$. All these oracles are built using two bijections f and h randomly chosen in $\mathcal{F}(B(n))$, but not given to A. The bijections f and h must be compatible with $(0_f, g_f, 0_h, g_h)$: $f(0) = 0_f$, $f(1) = g_f$, $g(0) = 0_h$, $g(1) = g_h$.

We suppose that the algorithm A is a probabilistic Turing machine. We assess its running time by the number of calls to the group and pairing oracles. We want to measure the asymptotic behavior of its average success probability as n goes to infinity, the probability being taken for a fixed n over the set of pairs $(f, h) \in \mathcal{F}(B(n))^2$.

In order to analyze the algorithm A, we maintain two series of lists, R and S, with values in $B(n) \times \mathbb{Z}_n(X_1, \ldots, X_n, Y_1, \ldots, Y_n)$ where $\mathbb{Z}_n(X_1, \ldots, Y_n)$ is the field of rational functions in the variables $X_1, \ldots, X_n, Y_1, \ldots, Y_n$. The lists R_k and S_k represent the "knowledge" of A after k queries to the oracles. The integers r_k and s_k index the number of variables used in the lists R_k and S_k. Let ρ_k and σ_k be the sizes of R_k and S_k. When A makes a new call to an oracle, r_{k+1}, s_{k+1}, ρ_{k+1}, σ_{k+1}, R_{k+1} and S_{k+1} are initialized with the values of r_k, s_k, ρ_k, σ_k, R_k and S_k and updated as follows:

- in the case of a call $a +_f b$, $-_f a$ or $\hat{e}_{f,h}(a, b)$, if the element a (resp. b) is not the first member of a pair in R_k, we increase r_{k+1} and ρ_{k+1} by one, we define $x_{r_{k+1}} = a$ (resp. b) and add $(x_{r_{k+1}}, X_{r_{k+1}})$ to R_{k+1},
- in the case of a call $a +_h b$ or $-_h a$, if the element a (resp. b) is not the first member of a pair in S_k, we increase s_{k+1} and σ_{k+1} by one, we define $y_{s_{k+1}} = a$ (resp. b) and add $(y_{s_{k+1}}, Y_{s_{k+1}})$ to S_{k+1},
- when c is a fresh answer to the call $a +_f b$ (resp. $-_f a$), the pair $(c, P_a + P_b)$ (resp. the pair $(c, -P_a)$) is added in R_{k+1}, where (a, P_a) and (b, P_b) are in R_k, and we increase ρ_{k+1} by one,
- when c is a fresh answer to the call $a +_h b$ (resp. $-_h a$), the pair $(c, P_a + P_b)$ (resp. the pair $(c, -P_a)$) is added in S_{k+1}, where (a, P_a) and (b, P_b) are in S_k, and we increase σ_{k+1} by one,
- when c is a fresh answer to the call $\hat{e}_{f,h}(a, b)$, the pair $(c, P_a.P_b)$ is added in S_{k+1}, where (a, P_a) and (b, P_b) are in R_k, and we increase σ_{k+1} by one.

We remark that the previous rules imply that $\rho_{k+1} + \sigma_{k+1} \leqslant \rho_k + \sigma_k + 3$: each query corresponds to at most three new pairs in the lists.

Definition XI.13 A pair of bijections $(f, h) \in \mathcal{F}(B(n))^2$ is said to be compatible with $(\mathrm{R}_k, \mathrm{S}_k)$ if:

1. $\forall (v_R, P_R) \in \mathrm{R}_k$, $P_R(f^{-1}(x_1), \ldots, f^{-1}(x_{r_k}), h^{-1}(y_1), \ldots, h^{-1}(y_{s_k}))$ is defined and equal to $f^{-1}(v_R)$,
2. $\forall (v_S, P_S) \in \mathrm{S}_k$, $P_S(f^{-1}(x_1), \ldots, f^{-1}(x_{r_k}), h^{-1}(y_1), \ldots, h^{-1}(y_{s_k}))$ is defined and equal to $h^{-1}(v_S)$.

We consider the algorithm A after k calls to oracles. By definition XI.13 and by construction of the oracles, the pair $(f, h) \in \mathcal{F}(B(n))^2$ used in the oracles is compatible with $(\mathrm{R}_k, \mathrm{S}_k)$. But any pair $(f, h) \in \mathcal{F}(B(n))^2$ compatible with $(\mathrm{R}_k, \mathrm{S}_k)$ would have produced the same answers to the calls chosen by the algorithm A. There is then no way for the algorithm A to distinguish the pair of bijection used in the oracles from any other pair of bijections compatible with $(\mathrm{R}_k, \mathrm{S}_k)$.

In order to state and to prove our main lemma, we need to define the notions of collision and of coherence:

Definition XI.14 We say that there is a collision in $(\mathrm{R}_k, \mathrm{S}_k)$ if there exists some pair $((v_1, P_1), (v_2, P_2))$ in $(\mathrm{R}_k)^2$ or in $(\mathrm{S}_k)^2$ such that: $v_1 = v_2$ and $P_1 \neq P_2$.

Definition XI.15 A $(r + s)$-uple $(\alpha_1, \ldots, \alpha_r, \beta_1, \ldots, \beta_s) \in (\mathbb{Z}_n)^{r+s}$ is said to be coherent with a set Π of rational functions in $\mathbb{Z}_n(X_1, \ldots, X_r, Y_1, \ldots, Y_s)$ if: $\forall P \in \Pi$, $P(\alpha_1, \ldots, \alpha_r, \beta_1, \ldots, \beta_s)$ is defined and $\forall (P_1, P_2) \in \Pi^2, \big(P_1 \neq P_2 \implies P_1(\alpha_1, \ldots, \alpha_r, \beta_1, \ldots, \beta_s) \neq P_2(\alpha_1, \ldots, \alpha_r, \beta_1, \ldots, \beta_s)\big)$.

Lemma XI.16 *Suppose that $n = p^\lambda$ for p a prime number. For a fixed k, we consider a collision-free pair of lists $(\mathrm{R}_k, \mathrm{S}_k)$, and some $(k + 1)^{th}$ call to an oracle. Writing all elements of R_k and S_k in reduced form, let d_1 be the maximum degree of numerators of R_k and S_k, d_2 be the maximum degree of denominators of R_k and S_k, and let $d = d_1 + d_2$. When $\rho_k + \sigma_k < \sqrt{2p/3d}$, the probability over all pairs of bijections compatible with $(\mathrm{R}_k, \mathrm{S}_k)$ that a new call leads to a pair of lists $(\mathrm{R}_{k+1}, \mathrm{S}_{k+1})$ with collision is bounded by*

$$\frac{6d\,(\rho_k + \sigma_k)}{2p - 3d\,(\rho_k + \sigma_k)^2} .$$

PROOF: We consider the set $\mathcal{C}$ of $(r_{k+1} + s_{k+1})$-uples which are simultaneously coherent (see Definition XI.15) with the set of rational functions in R_k and the set of rational functions in S_k : we compute the number of pairs of bijections compatible with $(\mathrm{R}_k, \mathrm{S}_k)$ using $\#\mathcal{C}$.

For all pairs (f, h) compatible with $(\mathrm{R}_k, \mathrm{S}_k)$, since there is no collision in $(\mathrm{R}_k, \mathrm{S}_k)$, the $(r_{k+1} + s_{k+1})$-uple $(f^{-1}(x_1), \ldots, f^{-1}(x_{r_{k+1}}), h^{-1}(y_1), \ldots, h^{-1}(y_{s_{k+1}}))$ is in $\mathcal{C}$.

Reciprocally, for all $(\alpha_1, \ldots, \alpha_{r_{k+1}}, \beta_1, \ldots, \beta_{s_{k+1}})$ in $\mathcal{C}$, we can build compatible pairs of bijections (f, h) such that $f^{-1}(x_i) = \alpha_i$ and $h^{-1}(y_j) = \beta_j$. To these r_{k+1} fixed values for f and s_{k+1} fixed values for h, the compatibility condition adds $\rho_k - r_k$ other fixed values for f and $\sigma_k - s_k$ other fixed values for h.

We have then exactly $(\#\mathcal{C})\,(n + r_k - r_{k+1} - \rho_k)!\,(n + s_k - s_{k+1} - \sigma_k)!$ pairs of bijections compatible with $(\mathrm{R}_k, \mathrm{S}_k)$.

Even if the sets R_{k+1} and S_{k+1} are not fully defined since the answer to the $(k+1)^{\text{th}}$ call is unknown, the rational functions in R_{k+1} and in S_{k+1} are already given by the new call. We can thus define the set $\mathcal{C}'$ of $(r_{k+1} + s_{k+1})$-uples which are simultaneously coherent with the set of rational functions in R_{k+1} and the set of rational functions in S_{k+1}. Using the same enumeration as previously, we compute the number of pairs of bijections compatible with $(\mathrm{R}_k, \mathrm{S}_k)$ and leading to some collision-free $(\mathrm{R}_{k+1}, \mathrm{S}_{k+1})$ from $\#\mathcal{C}'$:

We have exactly $(\#\mathcal{C}')\,(n+r_k-r_{k+1}-\rho_k)!\,(n+s_k-s_{k+1}-\sigma_k)!$ pairs of bijections compatible with $(\mathrm{R}_k, \mathrm{S}_k)$ and leading to some collision-free $(\mathrm{R}_{k+1}, \mathrm{S}_{k+1})$.

The probability that the $(k+1)^{\text{th}}$ call to an oracle leads to a pair of lists $(\mathrm{R}_{k+1}, \mathrm{S}_{k+1})$ with collision, is then $(\#\mathcal{C} - \#\mathcal{C}')/\#\mathcal{C}$. As previously mentioned, the $(k+1)^{\text{th}}$ call to an oracle corresponds to at most three new pairs in R_{k+1} and in S_{k+1} with only one of them, whose rational function is called P, resulting in a possible collision. If there is such a new pair in R_{k+1}, then $(\alpha_1, \ldots, \alpha_{r_{k+1}}, \beta_1, \ldots, \beta_{s_{k+1}})$ is simultaneously coherent with the set of rational functions in R_k, and not coherent with the set of rational functions in R_{k+1}: there is some $(v_R, P_R) \in \mathrm{R}_k$ such that $(P - P_R)(\alpha_1, \ldots, \alpha_{r_{k+1}}, \beta_1, \ldots, \beta_{s_{k+1}}) = 0$.

We can consider at most ρ_k differences $P - P_R$ of rational functions. Each one of theses differences, in the reduced form, has a numerator with at most $(d.n^{r_{k+1}+s_{k+1}}/p)$ roots, using [Sch80] Lemma 1. The number of $(r_{k+1} + s_{k+1})$-uples coherent with R_k, and not coherent with R_{k+1} is then bounded by $(\rho_k \,.\, d \,.\, n^{r_{k+1}+s_{k+1}}/p)$.

In the same way, the number of $(r_{k+1} + s_{k+1})$-uples coherent with S_k, and not coherent with S_{k+1} is bounded by $(\sigma_k \,.\, d \,.\, n^{r_{k+1}+s_{k+1}}/p)$. The probability that the $(k+1)^{\text{th}}$ call to an oracle leads to a pair of lists $(\mathrm{R}_{k+1}, \mathrm{S}_{k+1})$ with collision, is then bounded by $d \,.\, n^{r_{k+1}+s_{k+1}}\,(\rho_k + \sigma_k)/(p \,.\, \#\mathcal{C})$.

Using again [Sch80] Lemma 1 over the $(\rho_k(\rho_k - 1) + \sigma_k(\sigma_k - 1))/2$ numerators of differences of rational functions in R_k or in S_k, we obtain:

$$\#\mathcal{C} \geqslant \frac{n!}{(n - r_{k+1} - s_{k+1})!} - \frac{\rho_k(\rho_k - 1) + \sigma_k(\sigma_k - 1)}{2} \,.\, \frac{d \,.\, n^{r_{k+1}+s_{k+1}}}{p} \,.$$

Since $r_{k+1} + s_{k+1} \leqslant \sqrt{n}$, using a straightforward computation, we get

$$\frac{n!}{(n - r_{k+1} - s_{k+1})!} \geqslant \frac{n^{r_{k+1}+s_{k+1}}}{3} \,.$$

We obtain then $p \,.\, \#\mathcal{C}_k \geqslant n^{r_{k+1}+s_{k+1}} \,.\, (p/3 - d\,(\rho_k + \sigma_k)^2/2)$. The probability that the $(k+1)^{\text{th}}$ call to an oracle leads to a pair of lists $(\mathrm{R}_{k+1}, \mathrm{S}_{k+1})$ with collision is then bounded by:

$$\frac{6d\,(\rho_k + \sigma_k)}{2p - 3d\,(\rho_k + \sigma_k)^2} \,.$$

□

3.2. Illustration with the Bilinear Diffie-Hellman Problem

We now consider the bilinear Diffie-Hellman problem over the generic family of cyclic groups with pairing. In the two following corollaries, we assume that n is a prime power: $n = p^\lambda$. The case of a general composite order is considered later, in Theorem XI.19. Let A be an algorithm solving the bilinear Diffie-Hellman problem. Its input is $n \in \mathbb{N}^*$, together with $(0_f, g_f, 0_h, g_h) \in B(n)^4$ and $(x_1, x_2, x_3) \in B(n)^3$. At the beginning, $\mathrm{R}_0 = \{(0_f, 0), (g_f, 1), (x_1, X_1), (x_2, X_2), (x_3, X_3)\}$ and $\mathrm{S}_0 = \{(0_h, 0), (g_h, 1)\}$.

Corollary XI.17 *For a fixed $n = p^\lambda$, let $k \leqslant (\sqrt{p/3} - 5)/3$. When R_k and S_k contain polynomials of degree at most equal to 2, the probability that the pair of lists $(\mathrm{R}_k, \mathrm{S}_k)$ is with collision after k calls to the oracles is bounded by $2\,(3k+4)^2/(p - 3\,(3k+4)^2)$, where the probability is computed over all pairs of bijections compatible with $(\mathrm{R}_k, \mathrm{S}_k)$.*

PROOF: In each call to an oracle, at most 3 new pairs are added in $\mathrm{R}_k \cup \mathrm{S}_k$. We obtain then that for all i, $\rho_i + \sigma_i \leqslant 3i + 7$. We deduce that $6\,(\rho_i + \sigma_i)/(p - 3\,(\rho_i + \sigma_i)^2)$ is upper-bounded by $6\,(3i+7)/(p - 3\,(3i+7)^2)$.

The probability that the pair of lists $(\mathrm{R}_k, \mathrm{S}_k)$ is collision-free after k calls to the oracles is equal to the product of probabilities that the pair of lists $(\mathrm{R}_{i+1}, \mathrm{S}_{i+1})$ is collision-free, knowing that $(\mathrm{R}_i, \mathrm{S}_i)$ is collision-free, where $i \in \{0, \dots, k-1\}$. Since $0 < 6\,(3k+4) < p - 3\,(3k+4)^2$, we can give a lower bound for this probability of no-collision after k calls using Lemma XI.16:

$$\prod_{i=0}^{k-1} \left(1 - \frac{6\,(3i+7)}{p - 3\,(3i+7)^2}\right) \geqslant \left(1 - \frac{6\,(3k+4)}{p - 3\,(3k+4)^2}\right)^k \geqslant 1 - \frac{2\,(3k+4)^2}{p - 3\,(3k+4)^2} \ .$$

The probability that the pair of lists $(\mathrm{R}_k, \mathrm{S}_k)$ is with collision after k calls to the oracles is then bounded by: $2\,(3k+4)^2/(p - 3\,(3k+4)^2)$. □

Corollary XI.18 *Let A be an algorithm solving the bilinear Diffie-Hellman problem in the generic family of cyclic groups with pairing. When $k < (\sqrt{2p/9} - 7)/3$, the probability of success for A after k calls to the oracles, over groups of size p^λ, when $(\mathrm{R}_k, \mathrm{S}_k)$ is collision-free, is less than $18\,(3k+7)/(2p - 9\,(3k+7)^2)$.*

PROOF: Let z be the answer given by the algorithm A, after k calls to oracles leading to a collision-free pair of lists $(\mathrm{R}_k, \mathrm{S}_k)$.

If there is no P_z such that $(z, P_z) \in \mathrm{S}_k$, then z is indistinguishable from any other element of $B(n)$ with the same property (i.e. not in a pair in S_k). Thus, the answer z given by the algorithm A is valid with a probability less than $1/(p - \sigma_k) \leqslant 1/(p - 3k - 2)$.

Else, if the answer z is correct, a "virtual" call to an oracle corresponding to the polynomial $X_1.X_2.X_3$ would lead to a collision in S_{k+1}. We can then reuse Lemma XI.16 to obtain the following bound for the probability of such an event:

$$\frac{18\,(\rho_k + \sigma_k)}{2p - 9\,(\rho_k + \sigma_k)^2} \leqslant \frac{18\,(3k+7)}{2p - 9\,(3k+7)^2} \ .$$

The probability that A solves the problem is then bounded by the maximum of the two previously given probabilities, which is obviously the second one. □

We obtain then the following theorem:

Theorem XI.19 *Let* A *be an algorithm solving the bilinear Diffie-Hellman problem in the generic family of representations of cyclic groups with pairing.* A *is supposed to have unbound computational power, and be able to call the groups and pairing oracles in a probabilistic manner. After* k *calls to the oracles, over groups of order divisible by a prime number* p*, when* $k < (\sqrt{2p/9} - 7)/3$*, the probability that* A *succeeds is bounded by:*

$$\frac{2\,(3k+4)^2}{p - 3\,(3k+4)^2} + \frac{18\,(3k+7)}{2p - 9\,(3k+7)^2}\ .$$

PROOF: Let n be the common order of the groups. First suppose that $n = p^\lambda$ with p a prime number. An algorithm A may output a valid answer after k calls to oracles in two cases: $(\mathrm{R}_k, \mathrm{S}_k)$ is with collision, or collision-free. The first case is considered in Corollary XI.17, the second one is considered in Corollary XI.18.

We consider now the case of a general composite order: $n = p^\lambda t$, where p does not divide t. From any algorithm A_n solving the bilinear Diffie-Hellman problem in groups of order n, we build an algorithm A_{p^λ} solving the bilinear Diffie-Hellman problem in groups of order p^λ, with at least the same probability of success:

1. A_{p^λ} randomly chooses two bijections $\phi_1, \phi_2 : B(p^\lambda) \times \mathbb{Z}_t \to B(n)$, and 3 elements α_1, α_2, α_3 in $\mathbb{Z}_t$,
2. A_{p^λ} obtains $(0_f, g_f, 0_h, g_h) \in B(p^\lambda)^4$, and $(x_1, x_2, x_3) \in B(p^\lambda)^3$,
3. A_{p^λ} computes $0'_f = \phi_1(0_f, 0)$, $0'_h = \phi_2(0_h, 0)$, $g'_f = \phi_1(g_f, 1)$, $g'_h = \phi_2(g_h, 1)$, $x'_1 = \phi_1(x_1, \alpha_1)$, $x'_2 = \phi_1(x_2, \alpha_2)$, and $x'_3 = \phi_1(x_3, \alpha_3)$,
4. A_{p^λ} uses A_n with the following input: $(0'_f, g'_f, 0'_h, g'_h)$ and (x'_1, x'_2, x'_3),
5. when A_n outputs $z' = \phi_2(z, \alpha)$, A_{p^λ} outputs z.

For step 1, we remark that there is a natural bijection $\phi : \mathbb{Z}_{p^\lambda} \times \mathbb{Z}_t \to \mathbb{Z}_n$ given by $\phi(a, b) = a + p^\lambda b$, so that this step boils down to choose random permutations over $B(n)$.

A_{p^λ} must deal with oracle queries from A_n. The oracles $+'_f$, $-'_f$, $+'_h$, $-'_h$, and $\hat{e}'_{f,h}$ used by A_n are then simply deduced from the oracles $+_f$, $-_f$, $+_h$, $-_h$, and $\hat{e}_{f,h}$ given to A_{p^λ} using the following formulas:

- $+'_f : (\phi_1(a_1, b_1), \phi_1(a_2, b_2)) \in B(n)^2 \mapsto \phi_1(a_1 +_f a_2, b_1 + b_2) \in B(n)$,
- $-'_f : \phi_1(a, b) \in B(n) \mapsto \phi_1(-_f a, -b) \in B(n)$,
- $+'_h : (\phi_2(a_1, b_1), \phi_2(a_2, b_2)) \in B(n)^2 \mapsto \phi_2(a_1 +_h a_2, b_1 + b_2) \in B(n)$,
- $-'_h : \phi_2(a, b) \in B(n) \mapsto \phi_2(-_h a, -b) \in B(n)$,
- $\hat{e}'_{f,h} : (\phi_1(a_1, b_1), \phi_1(a_2, b_2)) \in B(n)^2 \mapsto \phi_2(\hat{e}_{f,h}(a_1, a_2), b_1.b_2) \in B(n)$.

If $z' = \phi_2(z, \alpha)$ is the solution of the bilinear Diffie-Hellman problem given to A_n, then z is the solution of the bilinear Diffie-Hellman problem given to A_{p^λ}. The probability of success of A_{p^λ} is then at least equal to the one of A_n. □

Remark XI.20 As an immediate consequence of this last theorem, if the number k of calls to oracles is very small in comparison with p, then the probability that A succeeds is bounded by $O(k^2/p)$.

Remark XI.21 Using a polynomial reduction of the bilinear Diffie-Hellman problem over the discrete logarithm problem, the Theorem XI.19 is immediately transposed with the same success probability to the usual discrete logarithm problem in the generic family of cyclic groups with pairing. In a way, the pairing does not help to solve the discrete logarithm problem.

Remark XI.22 Since there is also a polynomial reduction of the bilinear Diffie-Hellman problem over the usual Diffie-Hellman problem, the Theorem XI.19 immediately implies the difficulty of the computational Diffie-Hellman problem in the generic family of cyclic groups with pairing even if the decisional Diffie-Hellman problem is easy in this family.

3.3. Other Problems

In fact, the technique described in Section 3.1 is quite general. We illustrate its ubiquity by proving the difficulty of the q-$\mathcal{BDHI}$ problem, introduced in [BB04a].

Definition XI.23 Let $(\Omega, (L_\gamma)_{\gamma\in\Gamma}, (M_\delta)_{\delta\in\Delta}, (\hat{e}_\alpha)_{\alpha\in\Omega})$ be a family of representations of cyclic groups with pairing over two languages L and M, restricted to prime order groups. In this family, an algorithm solving the q-bilinear Diffie-Hellman inversion (q-$\mathcal{BDHI}$) problem computes the element $(1/\nu)\, g_\delta$ in the group M_δ from the inputs $(\gamma, \delta) \in \Omega$, $(\nu\, g_\gamma, \nu^2\, g_\gamma, \ldots, \nu^q\, g_\gamma) \in (L_\gamma)^q$.

Let A be an algorithm solving the q-$\mathcal{BDHI}$ problem in the generic family of cyclic groups with pairing. Its input is a prime number p, together with $(0_f, g_f, 0_h, g_h) \in B(p)^4$ and $(\nu\, g_f, \ldots, \nu^q\, g_f) \in B(p)^q$ for $\nu \in \mathbb{Z}_p^*$. At the beginning, $\mathrm{R}_0 = \{(0_f, 0), (g_f, 1), (\nu\, g_f, X_1), (\nu^2\, g_f, {X_1}^2), \ldots, (\nu^q\, g_f, {X_1}^q)\}$ and $\mathrm{S}_0 = \{(0_h, 0), (g_h, 1)\}$. This pair of lists is updated as described in Section 3.1.

Corollary XI.24 *For a fixed prime $n = p$, let $k \leqslant (\sqrt{p/3q} - q - 5)/3$. When R_k and S_k contain polynomials of degree at most equal to $2q$, the probability that the pair of lists $(\mathrm{R}_k, \mathrm{S}_k)$ is with collision, after k calls to the oracles, is bounded by*

$$\frac{2q\,.\,(3k+q+1)^2}{p - 3q\,(3k+q+1)^2} \;.$$

PROOF: The proof is exactly the same as the one of Corollary XI.17. Applying Lemma XI.16, we obtain the following lower bound for the probability of no-collision:

$$\prod_{i=0}^{k-1}\left(1 - \frac{6q\,(3i+q+4)}{p - 3q(3i+q+4)^2}\right) \geqslant \left(1 - \frac{6q(3k+q+1)}{p-3q(3k+q+1)^2}\right)^k$$
$$\geqslant 1 - \frac{2q\,(3k+q+1)^2}{p - 3q\,(3k+q+1)^2} \;.$$

□

Corollary XI.25 *Let* A *be an algorithm solving the q-$\mathcal{BDHI}$ problem in the generic family of cyclic groups with pairing. When* $k < (\sqrt{2p/(6q+3)} - q - 4)/3$*, the probability of success for* A *after* k *calls to the oracles, over groups of prime size* p*, when* $(\mathrm{R}_k, \mathrm{S}_k)$ *is collision-free, is less than*

$$\frac{6\,(2q+1)(3k+q+4)}{2p - 3\,(2q+1)(3k+q+4)^2}\ .$$

Proof: Let z be the answer given by the algorithm A, after k calls to oracles leading to a collision-free pair of lists $(\mathrm{R}_k, \mathrm{S}_k)$. As in the proof of corollary XI.18, if there is no P_z such that $(z, P_z) \in \mathrm{S}_k$ then the answer z is valid with a probability less than $1/(p-3k-2)$. Else, if z is a correct guess, a "virtual" call to the oracle corresponding to the rational function $1/X$ would produce a collision in S_{k+1}. The Lemma XI.16 gives the following bound for the probability of such an event:

$$\frac{6\,(2q+1)(\rho_k+\sigma_k)}{2p - 3\,(2q+1)(\rho_k+\sigma_k)^2} \leqslant \frac{6\,(2q+1)(3k+q+4)}{2p - 3\,(2q+1)(3k+q+4)^2}\ .$$

This second bound is larger than the first one, which implies the result. $\square$

From Corollaries XI.24 and XI.25, one can immediately deduce the following theorem.

Theorem XI.26 *Let* A *be an algorithm solving the q-$\mathcal{BDHI}$ problem in the generic family of cyclic groups with pairing.* A *is supposed to have unbound computational power, and able to call the groups and pairing oracles in a probabilistic manner. After* k *calls to the oracles, over groups of prime order* p*, when* $k < (\sqrt{2p/(6q+3)} - q - 4)/3$*, the probability that* A *succeeds is bounded by:*

$$\frac{2q\,(3k+q+1)^2}{p - 3q\,(3k+q+1)^2} + \frac{6\,(2q+1)(3k+q+4)}{2p - 3\,(2q+1)(3k+q+4)^2}\ .$$

3.4. Generalization for Polynomial-Based Problems

Following once more the same strategy, we give a generalization of Theorem XI.19 for problems based on inputs and output associated with polynomials:

Theorem XI.27 *We consider a problem such that inputs and queries are associated with polynomials of degree at most* d*, and output is associated with a polynomial of degree at most* d'*. Let* A *be an algorithm solving this problem in the generic family of representations of cyclic groups with pairing.* A *is supposed to have unbound computational power, and be able to call the groups and pairing oracles in a probabilistic manner. The output is supposed to be not computable by an universal formula from the inputs. After* k *calls to the oracles, over groups of order divisible by a prime number* p*, when* $k < (\sqrt{2p/(3\max(d,d'))} - \rho_0 - \sigma_0)/3$*, the probability that* A *succeeds is bounded by:*

$$\frac{2d\,(3k+\rho_0+\sigma_0-3)^2}{2p - 3d\,(3k+\rho_0+\sigma_0-3)^2} + \frac{6\,\max(d,d')\,(3k+\rho_0+\sigma_0)}{2p - 3\,\max(d,d')\,(3k+\rho_0+\sigma_0)^2}\ .$$

4. Pseudo-Random Family of Cyclic Groups

In this section, we introduce the notion of pseudo-random family of cyclic groups. Naively speaking a pseudo-random family of groups is the same thing as a generic family of groups except that the group law is not drawn at random in the set of all possible group laws: the group law follows a specific distribution which is computationally indistinguishable from a uniform distribution. We build a pseudo-random family of cyclic groups from a strong pseudo-random family of permutations.

Let $\mathfrak{P}$ be a set of permutations over $B(n)$. The notation $f \leftarrow \mathfrak{P}$ means that f is randomly and uniformly drawn in $\mathfrak{P}$. A distinguisher D is a Turing machine which has access to permutations over $B(n)$ through oracles and outputs a single bit. In the context of strong indistinguishability between two families of permutations $\mathfrak{P}_1$ and $\mathfrak{P}_2$ (see [LR88] for more details), a distinguisher D has access to f and f^{-1}, where $f \leftarrow \mathfrak{P}_1$ or $f \leftarrow \mathfrak{P}_2$. When D runs in time t and makes m oracle queries, its advantage is defined by the following formula:

$$\mathbf{Adv}^{s-prp}_{\mathfrak{P}_1,\mathfrak{P}_2}(\mathsf{D}_{t,m}) = \left|\Pr\left[\mathsf{D}^{f,f^{-1}}_{t,m} = 1 : f \leftarrow \mathfrak{P}_1\right] - \Pr\left[\mathsf{D}^{f,f^{-1}}_{t,m} = 1 : f \leftarrow \mathfrak{P}_2\right]\right| .$$

We say that $\mathfrak{P}_1$ and $\mathfrak{P}_2$ are (ϵ, t, m)-strongly indistinguishable if for all distinguishers D, the advantage $\mathbf{Adv}^{s-prp}_{\mathfrak{P}_1,\mathfrak{P}_2}(\mathsf{D}_{t,m})$ is upper-bounded by ϵ. We say that $\mathfrak{P}$ is a (ϵ, t, m)-strong pseudo-random family of permutations if it is (ϵ, t, m)-strongly indistinguishable from the set $\mathfrak{S}_n$ of all permutations over $B(n)$.

Definition XI.28 Let $\mathfrak{P}$ be a (ϵ, t, m)-strong pseudo-random family of permutations over $B(n)$. The pseudo-random family of cyclic groups associated to $\mathfrak{P}$ is the set of groups defined by the permutations f in $\mathfrak{P}$, with zero element $0_f = f(0)$, generator $g_f = f(1)$ and laws $+_f$ and $-_f$, as defined in Section 2.2.

Like in the generic family of cyclic groups, the group laws are given only through oracles. In this way, it is clear that a generic family of cyclic groups is a pseudo-random family of groups. As a consequence the notion of pseudo-random family of groups constitutes a generalization of the notion of generic family of cyclic groups.

We can now define the advantage of an algorithm for discrete logarithm-based problems in a pseudo-random family of cyclic groups: let A be an algorithm, which has access to group laws over $B(n)$ through oracles $+_f$ and $-_f$. When A runs in time t and makes m oracle queries, its advantages over the discrete logarithm, Diffie-Hellman and decisional Diffie-Hellman problems are defined by:

$$\begin{aligned}
\mathbf{Adv}^{\mathcal{DL}}_{\mathfrak{P}}(\mathsf{A}_{t,m}) &= \Pr\left[\mathsf{A}^{+_f,-_f}_{t,m}(0_f, g_f, x) = f^{-1}(x) : f \leftarrow \mathfrak{P},\, x \in B(n)\right] ,\\
\mathbf{Adv}^{\mathcal{DH}}_{\mathfrak{P}}(\mathsf{A}_{t,m}) &= \Pr\left[\mathsf{A}^{+_f,-_f}_{t,m}(0_f, g_f, x, y) = f^{-1}(x)\,.\,y : f \leftarrow \mathfrak{P},\, (x,y) \in B(n)^2\right] ,\\
\mathbf{Adv}^{\mathcal{DDH}}_{\mathfrak{P}}(\mathsf{A}_{t,m}) &= \Big|\Pr\left[\mathsf{A}^{+_f,-_f}_{t,m}(0_f, g_f, x, y, f^{-1}(x)\,.\,y) = 1 : f \leftarrow \mathfrak{P},\, (x,y) \in B(n)^2\right]\\
&\quad - \Pr\left[\mathsf{A}^{+_f,-_f}_{t,m}(0_f, g_f, x, y, z) = 1 : f \leftarrow \mathfrak{P},\, (x,y,z) \in B(n)^3\right]\Big| .
\end{aligned}$$

The maximum of these advantages over all algorithms A are respectively denoted by $\mathbf{Adv}^{\mathcal{DL}}_{\mathfrak{P}}(t,m)$, $\mathbf{Adv}^{\mathcal{DH}}_{\mathfrak{P}}(t,m)$, and $\mathbf{Adv}^{\mathcal{DDH}}_{\mathfrak{P}}(t,m)$. Theorems XI.29 and XI.30 give bounds of these probabilities, when $\mathfrak{P}$ is a $(\epsilon, t, 3(m+2))$-strong pseudo-random family of permutations over $B(n)$.

Theorem XI.29 *Let $\mathfrak{P}$ be a (ϵ, t, m)-strong pseudo-random family of permutations over $B(n)$. Then,*

$$\mathbf{Adv}_{\mathfrak{P}}^{\mathcal{DL}}(t, m/3-1) \leqslant \mathbf{Adv}_{\mathfrak{S}_n}^{\mathcal{DL}}(t, m/3-1) + \epsilon\,,$$

$$\mathbf{Adv}_{\mathfrak{P}}^{\mathcal{DH}}(t, m/3-2) \leqslant \mathbf{Adv}_{\mathfrak{S}_n}^{\mathcal{DH}}(t, m/3-2) + \epsilon\ .$$

PROOF: Let A be an algorithm for the discrete logarithm problem over groups, with laws given by the way of oracles built from a permutation f drawn randomly in $\mathfrak{P}$. From this algorithm A, we deduce a distinguisher D on the strong pseudo-random permutation family $\mathfrak{P}$. This distinguisher D takes as input a permutation f over $B(n)$ and its inverse f^{-1}. This permutation f has been randomly chosen in the strong pseudo-random permutation family $\mathfrak{P}$ or in the set $\mathfrak{S}_n$ of all permutations.

From this permutation, the distinguisher D builds an instance of the discrete logarithm problem for the algorithm A: it randomly chooses $r \in \mathbb{Z}_n$ and gives $f(0)$, $f(1)$ and $f(r)$ to the algorithm A. The distinguisher D builds the oracles corresponding to the group laws: $x +_f y = f\left(f^{-1}(x) + f^{-1}(y)\right)$, $-_f x = f\left(-f^{-1}(x)\right)$.

The algorithm A eventually outputs an answer to the discrete logarithm problem. If its answer is correct i.e. is equal to r, the distinguisher D outputs 1, else it outputs 0. If the permutation f has been chosen randomly in the strong pseudo-random permutation family $\mathfrak{P}$, the probability that the algorithm A outputs a correct answer is exactly its advantage for the discrete logarithm problem in the family of groups defined by $\mathfrak{P}$. In the other case, if the permutation f has been chosen randomly in the set $\mathfrak{S}_n$ of all permutations over $B(n)$, the probability that the algorithm A outputs a correct answer is exactly its advantage for the discrete logarithm problem in the generic family of groups.

The advantage of the distinguisher D is then exactly the difference of advantages of A in the two cases. Thus, if A runs in time t and can issue at most $(m/3-1)$ group oracle queries, this advantage is bounded by ϵ, when $\mathfrak{P}$ is a (ϵ, t, m)-strong pseudo-random permutation family. Thus, the advantage of an algorithm A for the discrete logarithm problem in the family of groups defined by $\mathfrak{P}$ is bounded by the advantage of solving this problem in the generic family of groups, plus ϵ. This proof can moreover be directly translated for the Diffie-Hellman problem. □

Theorem XI.30 *Let $\mathfrak{P}$ be a (ϵ, t, m)-strong pseudo-random family of permutations over $B(n)$. Then,*

$$\mathbf{Adv}_{\mathfrak{P}}^{\mathcal{DDH}}(t, m/3-2) \leqslant \mathbf{Adv}_{\mathfrak{S}_n}^{\mathcal{DDH}}(t, m/3-2) + 2\epsilon\ .$$

PROOF: Let A be an algorithm for the decisional Diffie-Hellman problem in a family of groups where the law is provided by oracles built from a permutation randomly drawn either from the random family of permutations $\mathfrak{S}_n$ or from a strong pseudo-random family of permutations $\mathfrak{P}$. Let b be the bit in the $\mathcal{DDH}$ problem which must be guessed by the algorithm. We have

$$\mathbf{Adv}_{\mathfrak{P}}^{\mathcal{DDH}}(\mathsf{A}_{t,m}) = \Big| \Pr\left[\mathsf{A}_{t,m}^{+f,-f} = 1 : f \leftarrow \mathfrak{P},\, b=1\right] - \Pr\left[\mathsf{A}_{t,m}^{+f,-f} = 1 : f \leftarrow \mathfrak{P},\, b=0\right]\Big| \ .$$

$$\mathbf{Adv}_{\mathfrak{S}_n}^{\mathcal{DDH}}(\mathsf{A}_{t,m}) = \Big| \Pr\left[\mathsf{A}_{t,m}^{+f,-f} = 1 : f \leftarrow \mathfrak{S}_n,\, b=1\right] - \Pr\left[\mathsf{A}_{t,m}^{+f,-f} = 1 : f \leftarrow \mathfrak{S}_n,\, b=0\right]\Big| \ .$$

From A, we build a distinguisher D against the strong pseudo-random family of permutations. This distinguisher has access to an oracle permutation f and its inverse f^{-1} drawn either from $\mathfrak{P}$ or $\mathfrak{S}_n$. Using the oracles f and f^{-1}, D can reply to the group law queries issued by A. It draws at random a bit b and two elements x, y in $B(n)$. If $b = 1$ it submits to A the input $(0_f, g_f, x, y, f^{-1}(x)\,.y)$ and if $b = 0$, it draws a random $z \in B(n)$ and submits to A the input $(0_f, g_f, x, y, z)$. At the end of the game A returns a bit b'. The distinguisher D returns 1 if $b' = b$ and 0 if $b' \neq b$. We have by definition

$$\mathbf{Adv}_{\mathfrak{P},\mathfrak{S}_n}^{s-prp}(\mathsf{D}_{t,m}) = \left| \Pr\left[\mathsf{D}_{t,m}^{f,f^{-1}} = 1 : f \leftarrow \mathfrak{P}\right] - \Pr\left[\mathsf{D}_{t,m}^{f,f^{-1}} = 1 : f \leftarrow \mathfrak{S}_n\right]\right| \ .$$

With this distinguisher, this means

$$\begin{aligned}\mathbf{Adv}_{\mathfrak{P},\mathfrak{S}_n}^{s-prp}(\mathsf{D}_{t,m}) = \Big| &\Pr\left[\mathsf{A}_{t,m/3-2}^{+f,-f} = 1 : f \leftarrow \mathfrak{P},\, b=1\right] \\ &+ \Pr\left[\mathsf{A}_{t,m/3-2}^{+f,-f} = 0 : f \leftarrow \mathfrak{P},\, b=0\right] \\ &- \Pr\left[\mathsf{A}_{t,m/3-2}^{+f,-f} = 1 : f \leftarrow \mathfrak{S}_n,\, b=1\right] \\ &- \Pr\left[\mathsf{A}_{t,m/3-2}^{+f,-f} = 0 : f \leftarrow \mathfrak{S}_n,\, b=0\right]\Big| \,/2 \ .\end{aligned}$$

As a consequence,

$$\mathbf{Adv}_{\mathfrak{P},\mathfrak{S}_n}^{s-prp}(\mathsf{D}_{t,m}) \geqslant \left| \mathbf{Adv}_{\mathfrak{P}}^{\mathcal{DDH}}(\mathsf{A}_{t,m/3-2}) - \mathbf{Adv}_{\mathfrak{S}_n}^{\mathcal{DDH}}(\mathsf{A}_{t,m/3-2})\right| /2 \ .$$

And thus, $\mathbf{Adv}_{\mathfrak{P}}^{\mathcal{DDH}}(\mathsf{A}_{t,m/3-2}) \leqslant \mathbf{Adv}_{\mathfrak{S}_n}^{\mathcal{DDH}}(\mathsf{A}_{t,m/3-2}) + 2\,\mathbf{Adv}_{\mathfrak{P},\mathfrak{S}_n}^{s-prp}(\mathsf{D}_{t,m})$. □

Remark XI.31 Theorems XI.29 and XI.30 and their proofs should serve as an illustration. Using the same kind of methods, one can prove any reasonable assumption in a pseudo-random family of cyclic groups. It is moreover possible to define pseudo-random families of cyclic groups with pairing from two strong pseudo-random families of permutations: bilinear assumptions can then be proved in this pseudo-random context.

In the two preceding proofs, the permutation is fixed at the beginning of the game in the reduction. This is something essential if the permutation is drawn from the pseudo-random family. Actually in that case it is not possible to build this pseudo-random permutation during the game by returning a random bit string to any fresh query. This fact contrasts with the usual generic group model and illustrates an important feature of our presentation of the generic group model where the group law is fixed at the beginning of the game.

There is some other subtle differences between the generic group model and the pseudo-random group model. For instance, it was mentioned in [AF07] that the generic model of groups should be able to take into account the following behavior of the algorithm A. Suppose that A receives a bit string $x \in B(n)$ from a group oracle query then A flips some bit, for instance the least significant bit, and use the resulting bit string in order to submit a new query. This kind of queries in not covered in the original model [Sho97] but it is easy to be convinced that these queries using fresh bit strings do not help to solve a given problem in the generic group model and this is what we prove in Section 3. In contrast in the pseudo-random group model this kind of queries may be used to attack the underlying pseudo-random permutation family and it is an important point to take them into account.

It should also be stressed that the pseudo-random groups problem depends on the computational power of the algorithm. This situation contrasts with the usual generic group model where all results are of information theoretic nature.

5. Summary

We have revisited the notion of generic group in order to describe a model which contains all features already seen in the literature: the ability to submit fresh bit strings and queries which correspond to rational functions of the exponents are worth mentioning. We have proved a bound on the problem of finding a collision in this model (Lemma XI.16). From this first bound, it is easy to derive precise bounds for all usual discrete logarithm related problems: we presented some examples (Theorems XI.19 and XI.26) to explain how to use our framework in a systematic manner.

From the presented model, it is possible to derive the notion of pseudo-random groups and to prove a reduction of some usual problems in the pseudo-random group model to their security in the generic group model and to the strong pseudo-random permutation hypothesis. Even though it constitute an improvement in term of realism, it is still not really satisfying. Actually, in the family of groups of interest in cryptography, the permutation on the underlying sets induced by the different group laws is far from being pseudo-random. It should be interesting to generalize further the notion of pseudo-random groups in order to make it more realistic.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-188

Chapter XII

Software Implementation of Pairings

Darrel HANKERSON [a], Alfred MENEZES [b] and Michael SCOTT [c]
[a] *Auburn University, USA*
[b] *University of Waterloo, Canada*
[c] *Dublin City University, Ireland*

Abstract. This chapter describes and compares the software implementation of popular elliptic curve pairings on two architectures, of which the Intel Pentium 4 and Core2 are representatives.

Keywords. Elliptic curve, Tate pairing, ate pairing, software implementation

Researchers have been studying methods for the efficient and secure implementation of conventional elliptic and hyperelliptic curve cryptographic schemes for over twenty years (see [HMV03,CF05]). Even though the arithmetic in low-genus hyperelliptic curves is conceptually quite simple and well understood, discoveries are still being made that significantly improve performance; two recent examples are the use of Edwards coordinates [BL07] and Theta functions [Gau07] to accelerate the addition rule for elliptic curves and genus 2 curves.

Since the arithmetic of pairings is considerably more complicated than conventional elliptic and hyperelliptic curve arithmetic, it is not surprising that there is substantial ongoing research on improving the performance of pairing-based protocols. As described in Chapter II, there have been numerous proposals for defining and computing cryptographically-suitable pairings $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. As a result, implementers of pairing-based protocols are faced with a bewildering selection of parameters choices. Among these choices are the embedding degree, the genus of the curve, the type of curve (supersingular or ordinary), the characteristic of the underlying field, and the prime-order groups $\mathbb{G}_1$ and $\mathbb{G}_2$. A particular selection of parameters can influence the functionality, efficiency, and security of the pairing application (see [GPS08]). This chapter focuses on the software implementation of pairings at the 128-bit security level. We provide detailed analyses and comparisons of pairing algorithms on three specific elliptic curves: an embedding degree 4 supersingular curve defined over $\mathbb{F}_{2^{1223}}$, an embedding degree 6 supersingular curve defined over $\mathbb{F}_{3^{509}}$, and an embedding degree 12 ordinary curve defined over a 256-bit prime field; these elliptic curves were previously considered in [AHM07] and [DSD07]. While our focus is on the pairing operation, we emphasize that a pairing-based protocol may involve other computationally-intensive operations such as point multiplication and field exponentiation.

Our implementations were done primarily on two architectures, of which the Intel Pentium 4 and Core2 are representatives. Although these processors are superficially similar, there are significant feature and performance differences, and we will examine how well the features of a particular platform can be exploited to accelerate a pairing implementation.

1. Symmetric Pairings

Let E be a supersingular elliptic curve defined over $\mathbb{F}_q$ with embedding degree $k > 1$. Let r be a prime divisor of $\#E(\mathbb{F}_q)$, let $P \in E(\mathbb{F}_q)$ be a point of order r, and let μ_r denote the order-r subgroup of $\mathbb{F}_{q^k}^*$. The symmetric pairing associated with E is a bilinear map $e_r : \langle P \rangle \times \langle P \rangle \to \mu_r$ defined by $e_r(P_1, P_2) = e(P_1, \phi(P_2))$, where e is a bilinear map and ϕ is a distortion map.

§1.1 and §1.2 describe two specific symmetric pairings derived from supersingular elliptic curves defined over the characteristic two field $\mathbb{F}_{2^{1223}}$ and the characteristic three field $\mathbb{F}_{3^{509}}$. These elliptic curves have embedding degrees 4 and 6, respectively. Both pairings attain the 128-bit security level because Pollard's rho method for computing discrete logarithms in the order-r subgroup of $E(\mathbb{F}_q)$ has running time at least 2^{128}, as do the index-calculus algorithms for computing discrete logarithms in the extension fields $\mathbb{F}_{q^k}$ [Len01].

1.1. Characteristic 2 Field ($k = 4$)

Field Representation Let $q = 2^{1223}$. We chose the following polynomial basis representation for $\mathbb{F}_{2^{1223}}$:

$$\mathbb{F}_{2^{1223}} = \mathbb{F}_2[z]/(z^{1223} + z^{255} + 1) \ .$$

That is, the elements of $\mathbb{F}_{2^{1223}}$ are the polynomials in $\mathbb{F}_2[z]$ of degree at most 1222, with multiplication performed modulo the irreducible trinomial $z^{1223} + z^{255} + 1$. Since $\mathbb{F}_{2^{1223}}$ has characteristic 2, squaring in $\mathbb{F}_{2^{1223}}$ is inexpensive relative to multiplication. Furthermore, since $\sqrt{c} = \sum c_{2i} z^i + \sqrt{z} \sum c_{2i+1} z^i$ for $c = \sum c_i z^i \in \mathbb{F}_{2^{1223}}$ and $\sqrt{z} = z^{612} + z^{128}$, square roots can also be computed inexpensively. The extension field $\mathbb{F}_{q^4}$ is represented using tower extensions $\mathbb{F}_{q^2} = \mathbb{F}_q[u]/(u^2 + u + 1)$ and $\mathbb{F}_{q^4} = \mathbb{F}_{q^2}[v]/(v^2 + v + u)$, whence a basis for $\mathbb{F}_{q^4}$ over $\mathbb{F}_q$ is $\{1, u, v, uv\}$.

Elliptic Curve The supersingular elliptic curve

$$E_1/\mathbb{F}_{2^{1223}} : y^2 + y = x^3 + x$$

has embedding degree $k = 4$. We have $\#E_1(\mathbb{F}_{2^{1223}}) = 5r$ where $r = (2^{1223} + 2^{612} + 1)/5$ is a 1221-bit prime. A distortion map is $\phi : (x, y) \mapsto (x + u^2, y + xu + v)$. Addition of two $E_1(\mathbb{F}_q)$ points using mixed affine-projective coordinates can be accomplished in 9 field multiplications [HMV03]. The doubling formula is $(x, y) \mapsto (x^4 + 1, x^4 + y^4 + 1)$, and hence the cost of doubling a point is relatively small.

Pairing Barreto, Galbraith, Ó' hÉigeartaigh and Scott [BGhS07] presented Algorithm XII.1 for computing the η_T pairing.

Algorithm XII.1 Computing the η_T pairing for $E_1/\mathbb{F}_{2^{1223}}$

```
Input: $P = (x_1, y_1)$ and $Q = (x_2, y_2) \in E_1(\mathbb{F}_{2^{1223}})[r]$.
Output: $\eta_T(P, Q)$.
1: $T \leftarrow x_1 + 1$.
2: $f \leftarrow T \cdot (x_1 + x_2 + 1) + y_1 + y_2 + (T + x_2)u + v$.
3: for $i$ from 1 to 612 do
4:     $T \leftarrow x_1$ ; $x_1 \leftarrow \sqrt{x_1}$ ; $y_1 \leftarrow \sqrt{y_1}$.
5:     $g \leftarrow T \cdot (x_1 + x_2) + y_1 + y_2 + x_1 + 1 + (T + x_2)u + v$.
6:     $f \leftarrow f \cdot g$.
7:     $x_2 \leftarrow x_2^2$ ; $y_2 \leftarrow y_2^2$.
8: return $f^{(q^2-1)(q-\sqrt{2q}+1)}$.
```

Analysis Step 5 costs 1 $\mathbb{F}_q$-multiplication. In Step 6, write $f = f_1 + f_2u + f_3v + f_4uv$ and $g = g_1 + g_2u + v$, where $f_i, g_j \in \mathbb{F}_q$. Then

$$\begin{aligned} f \cdot g &= (f_1g_1 + f_2g_2 + f_4) + (f_1g_2 + f_2g_1 + f_2g_2 + f_3 + f_4)u \\ &\quad + (f_1 + f_3 + f_3g_1 + f_4g_2)v + (f_2 + f_3g_2 + f_4 + f_4g_1 + f_4g_2)uv, \end{aligned}$$

which can be computed at a cost of 6 $\mathbb{F}_q$-multiplications. Now, qth-powering an element in $\mathbb{F}_{q^4}$ is essentially free because if $f = f_1 + f_2u + f_3v + f_4uv$ then

$$f^q = (f_1 + f_2 + f_3) + (f_2 + f_3 + f_4)u + (f_3 + f_4)v + f_4uv.$$

Note also that if $h = f^{q^2-1}$, then $h^{-1} = h^{q^2}$. It follows that the total cost of Step 8 is 1 inversion in $\mathbb{F}_{q^4}$, 3 multiplications in $\mathbb{F}_{q^4}$, and 612 squarings in $\mathbb{F}_q$ for the powering by $\sqrt{2q}$. Since these costs are dominated by the cost of Steps 3–7, a reasonable approximation for the overall cost of Algorithm XII.1 is the cost of these steps, namely $612 \times 7 = 4284$ $\mathbb{F}_q$-multiplications.

1.2. Characteristic 3 Field ($k = 6$)

Field Representation Let $q = 3^{509}$. We chose the following two polynomial basis representations for $\mathbb{F}_{3^{509}}$:

$$\mathbb{F}_3[z]/(z^{509} - z^{477} + z^{445} + z^{32} - 1) \text{ and } \mathbb{F}_3[z]/(z^{509} - z^{318} - z^{191} + z^{127} + 1).$$

Since $\mathbb{F}_{3^{509}}$ has characteristic 3, cubing in $\mathbb{F}_{3^{509}}$ is inexpensive relative to multiplication. Furthermore, the choice of the reduction polynomials enables cube roots to be computed significantly faster than an $\mathbb{F}_q$-multiplication (cf. §4.1.3). The extension field $\mathbb{F}_{q^6}$ is represented using tower extensions $\mathbb{F}_{q^3} = \mathbb{F}_q[u]/(u^3 - u - 1)$ and $\mathbb{F}_{q^6} = \mathbb{F}_{q^3}[v]/(v^2 + 1)$, whence a basis for $\mathbb{F}_{q^6}$ over $\mathbb{F}_q$ is $\{1, u, u^2, v, uv, u^2v\}$.

Elliptic Curve The supersingular elliptic curve

$$E_2/\mathbb{F}_{3^{509}} : y^2 = x^3 - x + 1$$

has embedding degree $k = 6$. We have $\#E_2(\mathbb{F}_{3^{509}}) = 7r$ where $r = (3^{509} - 3^{255} + 1)/7$ is an 804-bit prime. A distortion map is $\phi : (x, y) \mapsto (u - x, yv)$. Addition of two $E_2(\mathbb{F}_q)$ points using mixed affine-projective coordinates can be accomplished in 9 field multiplications [BKLS02]. The tripling formula is $(x, y) \mapsto (x^9 - 1, -y^9)$, and hence the cost of tripling a point is relatively small.

Pairing Barreto, Galbraith, Ó' hÉigeartaigh and Scott [BGhS07] presented Algorithm XII.2 for computing the η_T pairing.

Algorithm XII.2 Computing the η_T pairing for $E_2/\mathbb{F}_{3^{509}}$

```
Input: $P = (x_1, y_1)$ and $Q = (x_2, y_2) \in E_2(\mathbb{F}_{3^{509}})[r]$.
Output: $\eta_T(P, Q)$.
1: $f \leftarrow (-x_1 - x_2 + 1) \cdot y_1 + y_1 u + y_2 v$.
2: for $i$ from 1 to 255 do
3:     $T \leftarrow x_1 + x_2 + 1$.
4:     $g \leftarrow -T^2 - Tu - u^2 + y_1 \cdot y_2 v$.
5:     $f \leftarrow f \cdot g$.
6:     $x_1 \leftarrow \sqrt[3]{x_1}$ ; $y_1 \leftarrow \sqrt[3]{y_1}$ ; $x_2 \leftarrow x_2^3$ ; $y_2 \leftarrow y_2^3$.
7: return $f^{(q^3-1)(q+1)(q-\sqrt{3q}+1)}$.
```

Analysis The running time analysis of Algorithm XII.2 is similar to that of Algorithm XII.1. The dominant calculation is the main loop, each iteration of which costs 14 $\mathbb{F}_q$-multiplications — 2 to compute g and 12 to compute $f \cdot g$. The overall cost of Algorithm XII.2 is thus approximately $255 \times 14 = 3570$ $\mathbb{F}_q$-multiplications.

2. Asymmetric Pairings

Let E be an ordinary elliptic curve defined over $\mathbb{F}_q$ having even embedding degree k with respect to a prime divisor r of $\#E(\mathbb{F}_q)$. Suppose further that $r^3 \nmid \#E(\mathbb{F}_{q^k})$ and $r^2 \nmid q^k - 1$. Let $P \in E(\mathbb{F}_q)$ be a point of order r, let $\mathbb{G}_1 = \langle P \rangle$, and let μ_r denote the order-r subgroup of $\mathbb{F}_{q^k}^*$. Suppose that E admits a twist E' of degree d over $\mathbb{F}_{q^e}$, where $e = k/d$. Let E' be such a twist for which $r \mid \#E'(\mathbb{F}_{q^e})$; the existence of E' is guaranteed by Theorem 9 of [HSV06]. Let $Q' \in E'(\mathbb{F}_{q^e})$ be a point of order r, and let $\mathbb{G}_2' = \langle Q' \rangle$. Then there is an efficiently computable group monomorphism $\phi_d : \mathbb{G}_2' \to E(\mathbb{F}_{q^k})$ such that $Q = \phi_d(Q') \notin E(\mathbb{F}_q)$. The group $\mathbb{G}_2 = \langle Q \rangle$ is the Trace-0 subgroup of $E(\mathbb{F}_{q^k})[r]$. The asymmetric pairings considered in this section are the (reduced) Tate pairing $t_r : \mathbb{G}_1 \times \mathbb{G}_2 \to \mu_r$, the ate pairing $a_r : \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r$, and the R-ate pairing $R_r : \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r$.

We only consider these pairings for the Barreto-Naehrig (BN) [BN05] elliptic curves (cf. Chapter II). These elliptic curves E are defined over prime fields $\mathbb{F}_p$, have prime order $\#E(\mathbb{F}_p)$, and have embedding degree $k = 12$. They are especially well suited for the 128-bit security level because if p is a 256-bit prime then Pollard's rho method for computing discrete logarithms in $E(\mathbb{F}_p)$ has running time approximately 2^{128}, as does the number field sieve algorithm for computing discrete logarithms in the extension fields $\mathbb{F}_{p^{12}}$. The BN curves also admit sextic twists ($d = 6$), which means that many computations can be restricted to the field $\mathbb{F}_{p^2}$ by working with the points in $\mathbb{G}_2'$ rather than with points in $\mathbb{G}_2$.

The BN curve we work with is

$$E_3/\mathbb{F}_p : y^2 = x^3 + 3$$

with BN parameter $z =$ 6000000000001F2D (in hexadecimal) [DSD07]. For this choice of BN parameter, $p = 36z^4 + 36z^3 + 24z^2 + 6z + 1$ is a 256-bit prime of Hamming weight 87, $r = \#E_3(\mathbb{F}_p) = 36z^4+36z^3+18z^2+6z+1$ is a 256-bit prime of Hamming weight 91, and $t - 1 = p - r = 6z^2 + 1$ is a 128-bit integer of Hamming weight 28; here $t = p + 1 - r$ is the trace of $E_3/\mathbb{F}_p$. Note that $p \equiv 7 \pmod 8$ (whence -2 is a nonsquare modulo p) and $p \equiv 1 \pmod 6$.

Field Representation The extension field $\mathbb{F}_{p^{12}}$ is represented using tower extensions $\mathbb{F}_{p^2} = \mathbb{F}_p[u]/(u^2 + 2)$, $\mathbb{F}_{p^6} = \mathbb{F}_{p^2}[v]/(v^3 - \xi)$ where $\xi = -u - 1$, and $\mathbb{F}_{p^{12}} = \mathbb{F}_{p^6}[w]/(w^2 - v)$. We also have the representation $\mathbb{F}_{p^{12}} = \mathbb{F}_{p^2}[W]/(W^6 - \xi)$ where $W = w$. Hence an element $\alpha \in \mathbb{F}_{p^{12}}$ can be represented in any of the following three ways:

$$\begin{aligned}\alpha &= a_0 + a_1 w, \quad \text{where } a_0, a_1 \in \mathbb{F}_{p^6}\\ &= (a_{0,0} + a_{0,1}v + a_{0,2}v^2) + (a_{1,0} + a_{1,1}v + a_{1,2}v^2)w \quad \text{where } a_{i,j} \in \mathbb{F}_{p^2}\\ &= a_{0,0} + a_{1,0}W + a_{0,1}W^2 + a_{1,1}W^3 + a_{0,2}W^4 + a_{1,2}W^5 \ .\end{aligned}$$

We let $(\mathsf{m}, \mathsf{s}, \mathsf{i})$, $(\tilde{\mathsf{m}}, \tilde{\mathsf{s}}, \tilde{\mathsf{i}})$, $(\mathsf{M}, \mathsf{S}, \mathsf{I})$ denote the cost of multiplication, squaring, inversion in $\mathbb{F}_p$, $\mathbb{F}_{p^2}$, $\mathbb{F}_{p^{12}}$, respectively. Experimentally, we have $\mathsf{s} \approx 0.9\mathsf{m}$ and $\mathsf{i} \approx 41\mathsf{m}$.[1] In our cost estimates that follow, we will make the simplifying assumption $\mathsf{s} \approx \mathsf{m}$. If $a \in \mathbb{F}_p$ and $\alpha \in \mathbb{F}_{p^n}$ for $n \in \{2, 6, 12\}$, then the cost of computing $a \cdot \alpha$ is $n\mathsf{m}$. For $\mathbb{F}_{p^2}$ arithmetic, we have $\tilde{\mathsf{m}} \approx 3\mathsf{m}$ (using Karatsuba's method which reduces a multiplication in a quadratic extension to 3 (rather than 4) small field multiplications), $\tilde{\mathsf{s}} \approx 2\mathsf{m}$ (using the complex method: $(a + bu)^2 = (a - b)(a + 2b) - ab + (2ab)u$), and $\tilde{\mathsf{i}} \approx \mathsf{i} + 2\mathsf{m} + 2\mathsf{s}$ (since $(a + bu)^{-1} = (a - bu)/(a^2 + 2b^2)$). Note also that p-th powering is free in $\mathbb{F}_{p^2}$ since $(a + bu)^p = a - bu$.

Karatsuba's method reduces a multiplication in a cubic extension to 6 (rather than 9) multiplications in the smaller field.[2] Hence a multiplication in $\mathbb{F}_{p^6}$ costs $18\mathsf{m}$. Squaring in $\mathbb{F}_{p^6}$ costs $2\tilde{\mathsf{m}}+3\tilde{\mathsf{s}} = 12\mathsf{m}$ via the following formulae [CH07]: if $\beta = b_0+b_1v+b_2v^2 \in \mathbb{F}_{p^6}$ where $b_i \in \mathbb{F}_{p^2}$, then $\beta^2 = (A+D\xi)+(B+E\xi)v+(B+C+D-A-E)v^2$ where $A = b_0^2$, $B = 2b_0b_1$, $C = (b_0 - b_1 + b_2)^2$, $D = 2b_1b_2$, and $E = b_2^2$.[3] Finally, as shown in [Sco07a, Section 3.2], inversion in $\mathbb{F}_{p^6}$ can be reduced to 1 inversion, 9 multiplications, and 3 squarings in $\mathbb{F}_{p^2}$.

[1]We observed $\mathsf{i} \approx 41\mathsf{m}$ on a Pentium 4 and $\mathsf{i} \approx 85\mathsf{m}$ on a Core2.

[2]The Toom-Cook method requires 5 multiplications, but is slower in practice [DhSD06].

[3]Squaring in $\mathbb{F}_{p^6}$ can also be accomplished in $11\mathsf{m}$ using the SQU3 formulae in [CH07], but at the expense of several additions, subtractions, and a division by 2.

Since $\mathbb{F}_{p^{12}}$ is a tower of quadratic, cubic, and quadratic extensions, Karatsuba's method gives $\mathsf{M} \approx 54\mathsf{m}$. By using the complex method for squaring in $\mathbb{F}_{p^{12}}$ and Karatsuba for multiplication in $\mathbb{F}_{p^6}$ and $\mathbb{F}_{p^2}$, we have $\mathsf{S} \approx 36\mathsf{m}$. Note, however, that if $\alpha = a + bw \in \mathbb{F}_{p^{12}}$ satisfies $\alpha^{p^6+1} = 1$ (and hence $a^2 - b^2v = 1$), then we have $\alpha^2 = (a+bw)^2 = (a^2+b^2v) + (2ab)w = (2b^2v+1) + [(a+b)^2 - b^2 - b^2v - 1]w$. Hence squaring such α, an operation denoted by S', can be reduced to 2 squarings in $\mathbb{F}_{p^6}$ and so $\mathsf{S}' \approx 24\mathsf{m}$ [SL02]. Inverting such α is essentially free because $\alpha^{-1} = \alpha^{p^6}$. Since inversion in $\mathbb{F}_{p^{12}}$ can be reduced to 1 inversion, 2 multiplications, and 2 squarings in $\mathbb{F}_{p^6}$, it follows that $\mathsf{I} \approx \mathsf{i} + 97\mathsf{m}$.

E_3 has a degree-6 twist over $\mathbb{F}_{p^2}$, namely $E'/\mathbb{F}_{p^2} : y^2 = x^3 + 3/\xi$. The monomorpism $\phi_6 : \mathbb{G}_2' \to \mathbb{G}_2$ is given by $(x,y) \mapsto (xW^2, yW^3)$.

2.1. Tate Pairing

Algorithm XII.3 computes the Tate pairing $t_r : \mathbb{G}_1 \times \mathbb{G}_2 \to \mu_r$ defined by $t_r(P,Q) = f_{r,P}(Q)^{(p^{12}-1)/r}$, where $f_{r,P}$ is the Miller function [Mil04] as defined in Chapter II. Steps 3–10 for computing $f_{r,P}(Q)$ (called a Miller operation) are from [BKLS02], while the technique for the final exponentiation (Steps 11–16) is from [DSD07].

Algorithm XII.3 Computing the Tate pairing for $E_3/\mathbb{F}_p$

```
Input: $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$.
Output: $t_r(P,Q)$.
 1: Write $r$ in binary: $r = \sum_{i=0}^{L-1} r_i 2^i$.
 2: $T \leftarrow P$ ; $f \leftarrow 1$.
 3: for $i$ from $L-2$ downto 0 do   {Miller operation}
 4:     Let $\ell$ be the tangent line at $T$.
 5:     $T \leftarrow 2T$.
 6:     $f \leftarrow f^2 \cdot \ell(Q)$.
 7:     if $r_i = 1$ and $i \neq 0$ then
 8:         Let $\ell$ be the line through $T$ and $P$.
 9:         $T \leftarrow T + P$.
10:         $f \leftarrow f \cdot \ell(Q)$.
11: Compute $f^{(p^{12}-1)/r}$ as follows: {Final exponentiation}
12:     $f \leftarrow f^{p^6-1}$.
13:     $f \leftarrow f^{p^2+1}$.
14:     $a \leftarrow f^{-(6z+5)}$ ; $b \leftarrow a^p$ ; $b \leftarrow a \cdot b$.
15:     Compute $f^p$ ; $f^{p^2}$ ; $f^{p^3}$.
16:     $f \leftarrow f^{p^3} \cdot [b \cdot (f^p)^2 \cdot f^{p^2}]^{6z^2+1} \cdot b \cdot (f^p \cdot f)^9 \cdot a \cdot f^4$.
17: return $f$.
```

Analysis A point (X,Y,Z) in Jacobian coordinates corresponds to the point (x,y) in affine coordinates with $x = X/Z^2$ and $y = Y/Z^3$. In Jacobian coordinates the formulae for doubling a point $T = (X,Y,Z)$ are $2T = (X_3, Y_3, Z_3)$ where $X_3 = 9X^4 - 8XY^2$, $Y_3 = (3X^2)(4XY^2 - X_3) - 8Y^4$ and $Z_3 = 2YZ$. The tangent line at T, after clearing denominators, is $\ell(x,y) = Z_3Z^2y - 2Y^2 - 3X^2(Z^2x - X) \in \mathbb{F}_p[x,y]$ [CSB04]. The cost of Steps 5–6[4] is $3\mathsf{m} + 4\mathsf{s}$ for computing $2T$, $7\mathsf{m} + \mathsf{s}$ for evaluating $\ell(Q)$ (note that the computation of $2T$ yields X^2, Y^2 and Z_3), $36\mathsf{m}$ for computing f^2, and $39\mathsf{m}$ for computing the product $f^2 \cdot \ell(Q)$ (note that $\ell(Q)$ has the form $a + bW^2 + cW^3$ with $a \in \mathbb{F}_p$ and $b, c \in \mathbb{F}_{p^2}$).

The formulae for mixed Jacobian-affine addition are the following: if $P = (X_1, Y_1, Z_1)$ is in Jacobian coordinates and $Q = (X_2, Y_2)$ is in affine coordinates, then $P + Q = (X_3, Y_3, Z_3)$ where $X_3 = (Y_2Z_1^3 - Y_1)^2 - (X_2Z_1^2 - X_1)^2(X_1 + X_2Z_1^2)$, $Y_3 = (Y_2Z_1^3 - Y_1)[X_1(X_2Z_1^2 - X_1)^2 - X_3] - Y_1(X_2Z_1^2 - X_1)^3$, $Z_3 = (X_2Z_1^2 - X_1)Z_1$. The line through T and P is $\ell(x, y) = (y - Y_2)Z_3 - (Y_2Z_1^3 - Y_1)(x - X_2) \in \mathbb{F}_p[x, y]$ [CSB04]. The cost of Steps 7–10 is $8\mathsf{m} + 3\mathsf{s}$ for computing $T + P$, $6\mathsf{m}$ for evaluating $\ell(Q)$, and $39\mathsf{m}$ for computing the product $f \cdot \ell(Q)$ (where again $\ell(Q)$ has the form $a + bW^2 + cW^3$ with $a \in \mathbb{F}_p$ and $b, c \in \mathbb{F}_{p^2}$).

Table XII.1 lists the operation costs for Step 11. Observe that exponentiation by p^6 is free in $\mathbb{F}_{p^{12}}$. Also, since $W^p = (W^6)^{(p-1)/6}W = \xi^{(p-1)/6}W$, we have $(\sum_{i=0}^{5} a_iW^i)^p = \sum(a_i^p \cdot \gamma_i)W^i$ where $\gamma_i = \xi^{i(p-1)/6} \in \mathbb{F}_{p^2}$. Thus, if the γ_i are precomputed, then powering an element in $\mathbb{F}_{p^{12}}$ by p can be accomplished with 5 $\mathbb{F}_{p^2}$-multiplications. Similarly, powering by p^2 and p^3 each costs 5 $\mathbb{F}_{p^2}$-multiplications. The exponentiations f^{6z+5}, T^z and $(T^z)^{6z}$ are performed by repeated square-and-multiply; to derive the costs note that $6z + 5$ and $6z$ have bitlength 66 and Hamming weight 11, while z has bitlength 63 and Hamming weight 11.

Table XII.1. Operation costs for the final exponentiation.

Operation	Cost
f^{p^6-1}	$\mathsf{I} + \mathsf{M}$
f^{p^2+1}	$\mathsf{M} + 5\tilde{\mathsf{m}}$
$f^{-(6z+5)}$	$10\mathsf{M} + 65\mathsf{S}'$
$a^p, a \cdot b, f^p, f^{p^2}, f^{p^3}$	$\mathsf{M} + 20\tilde{\mathsf{m}}$
$T \leftarrow b \cdot (f^p)^2 \cdot f^{p^2}$	$2\mathsf{M} + \mathsf{S}'$
$T \leftarrow T^{6z^2+1}$	$21\mathsf{M} + 127\mathsf{S}'$
$f^{p^3} \cdot T \cdot b \cdot (f^p \cdot f)^9 \cdot f^4$	$7\mathsf{M} + 5\mathsf{S}'$

Thus our estimated costs of the Miller operation and the final exponentiation are $255(85\mathsf{m} + 5\mathsf{s}) + 89(53\mathsf{m} + 3\mathsf{s})$ and $\mathsf{I} + 43\mathsf{M} + 198\mathsf{S}' + 25\tilde{\mathsf{m}}$. Setting $\mathsf{s} = \mathsf{m}$ yields the estimated costs $27934\mathsf{m}$, $7246\mathsf{m} + \mathsf{i}$, and $35180\mathsf{m} + \mathsf{i}$ for the Miller operation, the final exponentiation, and Algorithm XII.3, respectively.

2.2. Ate Pairing

The ate pairing $a_r : \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r$, as proposed by Hess, Smart and Vercauteren [HSV06], is defined to be $a_r(Q, P) = f_{t-1,Q}(P)^{(p^{12}-1)/r}$. Algorithm XII.4 for computing the ate pairing modifies Algorithm XII.3 by interchanging the roles of P and Q, and by using $t - 1$ (instead of r) to determine the number of iterations in the Miller operation. Since $t \approx \sqrt{r}$ for the BN curve E_3, the number of iterations in the Miller operation is halved.

[4]The first one or two iterations at Step 3 are cheaper than subsequent iterations because f may be sparse; for example $f = 1$ at the beginning of the first iteration of Step 6. Our counts will ignore these small differences.

Algorithm XII.4 Computing the ate pairing for $E_3/\mathbb{F}_p$

```
Input: $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$.
Output: $a_r(Q,P)$.
1: Write $t-1$ in binary: $t-1 = \sum_{i=0}^{L-1} t_i 2^i$.
2: $T \leftarrow Q$ ; $f \leftarrow 1$.
3: for $i$ from $L-2$ downto 0 do   {Miller operation}
4:     Let $\ell$ be the tangent line at $T$.
5:     $T \leftarrow 2T$.
6:     $f \leftarrow f^2 \cdot \ell(P)$.
7:     if $t_i = 1$ then
8:         Let $\ell$ be the line through $T$ and $Q$.
9:         $T \leftarrow T + Q$.
10:        $f \leftarrow f \cdot \ell(P)$.
11: return $f^{(p^{12}-1)/r}$, where $f^{(p^{12}-1)/r}$ is computed as in Algorithm XII.3.
```

Analysis The analysis of Algorithm XII.4 is similar to that of Algorithm XII.3. The doubling and addition formulae are the same, however they are actually applied to points in $E'(\mathbb{F}_{p^2})$, thus ensuring that elliptic curve arithmetic is over $\mathbb{F}_{p^2}$ (instead of over $\mathbb{F}_{p^{12}}$); a Jacobian point $(X,Y,Z) \in E'(\mathbb{F}_{p^2})$ conveniently maps to the Jacobian point $(XW^2, YW^3, Z) \in E(\mathbb{F}_{p^{12}})$. In Step 2, T is initialized to the Jacobian point $(xW^2, yW^3, 1)$, where $Q = (xW^2, yW^3) \in \mathbb{G}_2$. The point doubling in Step 5 costs $3\tilde{\mathsf{m}} + 4\tilde{\mathsf{s}}$. The tangent line at the affine point corresponding to $T = (XW^2, YW^3, Z)$ is $\ell(x,y) = Z_3Z^2y - 2Y^2W^3 - 3X^2W(Z^2x - XW^2) \in \mathbb{F}_{p^{12}}[x,y]$, where $2T = (X_3W^2, Y_3W^3, Z_3)$. Computing $\ell(P)$ and f^2 in Step 6 costs $3\tilde{\mathsf{m}} + \tilde{\mathsf{s}} + 4\mathsf{m}$ and $36\mathsf{m}$, respectively. Noting that $\ell(P)$ is of the form $a + bW + cW^3$ with $a,b,c \in \mathbb{F}_{p^2}$, we can write $\ell(P) = a + (b + cv)w$ where a and $(b + cv)$ are considered to be elements of $\mathbb{F}_{p^6}$. It follows that the product of $\ell(P)$ and $f^2 = f_0 + f_1w$ (where $f_0, f_1 \in \mathbb{F}_{p^6}$) can be computed using Karatsuba's technique at a cost of $13\tilde{\mathsf{m}}$. The cost of Steps 7–10 is $8\tilde{\mathsf{m}} + 3\tilde{\mathsf{s}}$ for computing $T+Q$, $2\tilde{\mathsf{m}} + 4\mathsf{m}$ for evaluating $\ell(P)$, and $13\tilde{\mathsf{m}}$ for computing $f \cdot \ell(P)$. Here, the line (after clearing denominators) through the affine point corresponding to $T = (X_1W^2, Y_1W^3, Z_1)$ and the point $Q = (X_2W^2, Y_2W^3)$ is $\ell(x,y) = (y - Y_2W^3)Z_3 - (Y_2Z_1^3 - Y_1)W(x - X_2W^2) \in \mathbb{F}_{p^{12}}[x,y]$ where $T+Q = (X_3W^2, Y_3W^3, Z_3)$; and $\ell(P)$ is of the form $a+bW+cW^3$ with $a,b,c \in \mathbb{F}_{p^2}$. Setting $\mathsf{s} = \mathsf{m}$ yields the estimated costs $15722\mathsf{m}$, $7246\mathsf{m} + \mathsf{i}$, and $22968\mathsf{m} + \mathsf{i}$ for the Miller operation, the final exponentiation, and Algorithm XII.4, respectively.

2.3. R-ate Pairing

The R-ate pairing $R_r : \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r$ is a generalization of the ate pairing due to Lee, Lee and Park [LLP08]. For the BN curve E_3, the R-ate pairing is defined by

$$R_r(Q,P) = \left(f \cdot (f \cdot \ell_{bQ,Q}(P))^p \cdot \ell_{\pi(bQ+Q),bQ}(P)\right)^{(p^{12}-1)/r},$$

where $b = 6z+2$, $f = f_{b,Q}(P)$, $\ell_{A,B}$ denotes the line through A and B, and $\pi : (x,y) \mapsto (x^p, y^p)$ is the Frobenius map. Since $b \approx \sqrt{t}$, the Miller operation in Algorithm XII.5 has half as many iterations as in Algorithm XII.4.

Algorithm XII.5 Computing the R-ate pairing for $E_3/\mathbb{F}_p$

```
Input: $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$.
Output: $R_r(Q,P)$.
 1: Write $b = 6z + 2$ in binary: $b = \sum_{i=0}^{L-1} b_i 2^i$.
 2: $T \leftarrow Q$ ; $f \leftarrow 1$.
 3: for $i$ from $L-2$ downto 0 do
 4:     Let $\ell$ be the tangent line at $T$.
 5:     $T \leftarrow 2T$.
 6:     $f \leftarrow f^2 \cdot \ell(P)$.
 7:     if $b_i = 1$ then
 8:         Let $\ell$ be the line through $T$ and $Q$.
 9:         $T \leftarrow T + Q$.
10:         $f \leftarrow f \cdot \ell(P)$.
11: $f \leftarrow f \cdot (f \cdot \ell_{T,Q}(P))^p \cdot \ell_{\pi(T+Q),T}(P)$.
12: return $f^{(p^{12}-1)/r}$, where $f^{(p^{12}-1)/r}$ is computed as in Algorithm XII.3.
```

Analysis The analysis of Steps 3–10 is the same as for Algorithm XII.4. Hence, since b has bitlength 66 and Hamming weight 9, the cost of Steps 3–10 is $7587\mathsf{m}$. The cost of Step 11 is: $10\tilde{\mathsf{m}} + 3\tilde{\mathsf{s}} + 4\mathsf{m}$ to compute $T + Q$ and evaluate $\ell_{T,Q}(P)$; $2\tilde{\mathsf{m}}$ to compute $\pi(T+Q)$; $\tilde{\mathsf{i}} + 3\tilde{\mathsf{m}} + \tilde{\mathsf{s}}$ to convert T to affine coordinates, and $10\tilde{\mathsf{m}} + 3\tilde{\mathsf{s}} + 4\mathsf{m}$ to compute $\pi(T+Q) + T$ and evaluate $\ell_{\pi(T+Q),T}(P)$; $30\tilde{\mathsf{m}}$ to multiply the two line evaluations into the accumulator; $5\tilde{\mathsf{m}}$ for the pth-power; and M for the multiplication by f. Setting $\mathsf{s} = \mathsf{m}$ yields an estimated cost of $7847\mathsf{m} + \mathsf{i}$ for Steps 3–11, $7246\mathsf{m} + \mathsf{i}$ for the final exponentiation, and $15093\mathsf{m} + 2\mathsf{i}$ for Algorithm XII.5.

3. Platform and Algorithm Notes

In algorithm analysis, operation counts (for the more expensive of the basic operations) typically suffices for rough comparisons. However, a significant portion of experimental results is often not explained by this higher-level analysis, in part because such analysis fails to adequately capture platform characteristics such as cache size and speed, number of registers, and pipelining. In this section, we provide context and technical details for the platforms and algorithms used in the timings. In particular, we discuss features of platforms that have been widely used for experimental data and their influence on algorithm, field, and curve selection.

3.1. Platform Selection

The selection of a specific processor can significantly affect experimental results and algorithm selection, even among processors that are of similar class or possibly even in the same family. In the past dozen years, processors from the Intel Pentium family have been the favourite for benchmarks, in large part because of their dominance in the consumer market. Among these processors, we will restrict our attention to the 32-bit "P6 family" (e.g., Pentium II, III) and Pentium 4 models 0–2.[5]

[5] In the transition to newer architectures, Intel confusingly introduced "Pentium 4" processors that were very different from the earlier models.

This choice has meant 32-bit platforms with only eight general-purpose registers and relatively fast access to memory. Although these processors are instruction-set similar, there are significant differences that are easily seen in practice. The Pentium 4 promised to scale to very high clock speeds, but the design had a significant penalty in cycle counts for integer multiplication, add-with-carry, and branch misprediction compared with earlier P6 designs. The penalty for arithmetic in general-purpose registers was offset by extensions in the single-instruction multiple-data (SIMD) instruction set that permit implementation of large integer multiplication that is cycle-competitive with the Pentium III.

The existence of a de facto reference platform has aided comparisons, but the industry has moved decisively to 64-bit platforms. These have been the standard on workstation-class systems for years, but are now commonplace on commodity hardware. The Core2 (and Xeon) from Intel and the Athlon64 (and Opteron) from AMD are, roughly speaking, extensions of the Pentium family processors to 64-bit instruction sets. Interestingly, Intel abandoned the Pentium 4 architecture, in part due to the success of the competing AMD Athlon. As a specific consequence, integer multiplication in general-purpose registers no longer suffers the significant penalty of the Pentium 4. Our goal here, in part, is to discuss issues specific to these 64-bit systems and to contrast with existing results on 32-bit systems.

The work by Avanzi and Thériault [AT07] provides a concrete example where the choice of "similar" platform significantly influences conclusions. In this case, the comparison is for scalar multiplication on elliptic vs. hyperelliptic curves. The processors are the Motorola PowerPC (G4) and the Intel Core2, superficially similar in the sense that both can be described as "workstation class." For 32-bit code, they observe that the "Core2 offers better multiplication performance...especially for larger fields." Times for point multiplication are given for the PowerPC, and genus 4, for example, is competitive with elliptic curves (although point halving methods for elliptic curves were not exploited). The difference in multiplication performance will mean that higher-genus is more attractive on the PowerPC than the Core2 in this scenario. A portion of this discrepancy can be explained by the RISC architecture on the PowerPC that more heavily (compared with the Core2) favours smaller fields where elements occupy a few registers.

3.2. Special Hardware

The processors reported here all possess "special purpose" hardware in the form of SIMD and floating-point registers. The SIMD registers are easily employed to extend operations in characteristic 2 or 3 fields to 64 or 128 bits. On the register-poor Pentium 4, this hardware supplies eight 64- or 128-bit registers for vector operations. For fields of characteristic 2 or 3, the basic idea is to use the hardware as wide registers. A factor 2 acceleration over conventional registers can be expected, although the precise improvement depends on instruction timings and specific operations. For example, many of the instructions on 128-bit registers have latency and throughput that are worse than their 64-bit counterparts. Further, shifting through 128 bits requires two or three (depending on shift amount) instructions unless the amount is a multiple of 8 bits.

For the 32-bit processors, the floating-point approach has been used by Bernstein [Ber01,Ber06] to obtain very fast point multiplication for elliptic curves over prime fields. The technique is not as straightforward as it may appear, in part because conversion to canonical form is expensive. On the Pentium 4 (where integer multiplication is expensive), an alternative is available in the SSE2 extensions to the SIMD registers. Coding with SSE2 integer operations obtains most of the speed improvement of the floating-point registers, but does not require the commitment across code demanded by the floating-point approach.

For the 64-bit processors considered here, the advantage of the 128-bit registers (for field arithmetic) is less clear. These processors have twice the number of general-purpose registers as the Pentium 4, and instruction timings in SIMD vary between the Intel and AMD offerings. On the AMD, for example, several of the operations of interest have better instruction timings with 64-bit general-purpose registers than with 128-bit SIMD registers. Some experimental results are discussed in §4.

For integer multiplication with 64-bit code, the general-purpose registers can directly multiply 64-bit quantities, while the SIMD registers are limited to operands of 32 bits. Similarly, the floating-point registers are of the same size as on 32-bit systems, and so this approach is less attractive (especially on the AMD which has a 5-cycle multiply with 64-bit operands).[6] On the other hand, multiplication with general-purpose registers has restrictive register requirements, while SIMD can perform two 32-bit multiplications per instruction and, along with the floating-point approach, does not place restrictions on registers.

In addition to the special registers, implementers have considered other attached hardware for cryptographic use. For example, some display adapters possess considerable computing power, although implementers have had mixed success in adapting their instruction set and interface for cryptographic use [MPS07,CIKL05]. Hardware targeted to cryptography has usually been an add-on, but the recent UltraSPARC T2 from Sun Microsystems may be a harbinger of widespread on-chip cryptographic hardware on common systems. Even a narrow hardware instruction set enhancement to include a characteristic 2 multiplier could have a dramatic effect: in tests on a SmartMIPS, a factor 5 speedup in multiplication was observed with the addition of a polynomial multiplier.

Finally, we note that the 64-bit hardware is commonly configured with multiple processors and/or cores. Algorithms that can be parallelized are perhaps of less interest here, since aggregate throughput (that is, number of pairings per unit time) is probably the measurement of interest on this class of hardware. For scenarios where an expensive pairing is to be calculated by a device with multiple but weak processing units, algorithms that can be parallelized in software would be desirable.

3.3. Sixty-Four Versus Thirty-Two

While 64-bit systems have been standard on workstations for years, the analysis for processors such as the Core2 and Athlon64 relative to the Pentium 4 "reference standard" may not be as straightforward as it appears. For example, the popular 64-bit Sun UltraSPARC was introduced without a full 64-bit multiplier [WG94]. In contrast, the Core2 and Athlon64 have a relatively fast and full 64-bit integer multiplier. Further, code on all the Intel and AMD offerings considered in this paper can exploit 128-bit SIMD registers and 80-bit extended floating-point capabilities.

[6]All the processors have 64-bit double-precision floating point capability, along with 80-bit extended precision. In the present context, the measurement of interest is the size of the significand, which is effectively 53-bit and 64-bit, resp. Since there is no penalty, the wider operand size available with the 80-bit format is preferred for integer arithmetic.

In the present comparison against the Pentium 4, the most interesting features of the 64-bit Intel and AMD processors are the increased number of registers and the 64-bit multiplier. All operations can potentially benefit from the 64-bit register size, but characteristic 2 or 3 arithmetic in the Pentium 4 was already exploiting wide operations in SIMD. Instruction timings must be considered for complete analysis, but we expect that the relatively fast 64-bit multiplier will mean that curves over prime fields benefit more in the move to these 64-bit architectures (in part, because the multiplication in characteristic 2 or 3 is still essentially a few bits at a time).

4. Implementation

In this section, we discuss specifics of the implementations and compare experimental results against estimates based on operation counts. In part, our goal is to obtain realistic estimates of the performance penalty in using supersingular curves rather than BN curves at the 128-bit security level on platforms with varying hardware features. To be certain, shortcomings in the implementations remain, but it is hoped that the results provide meaningful benchmarks for future work.

4.1. Implementation Details

We begin with details for the field arithmetic implemented for the example pairings. This includes three fields at sizes dictated by the 128-bit security level for the corresponding pairings, namely the 256-bit prime field $\mathbb{F}_p$ (§2), the 1223-bit characteristic 2 field $\mathbb{F}_{2^{1223}}$ (§1.1), and the 807-bit characteristic 3 field $\mathbb{F}_{3^{509}}$ (§1.2). The focus is on the platforms described in §3, although much of the material applies more widely.

4.1.1. Prime fields

Field multiplication is Montgomery form, and a fully-unrolled "comba" multiplier calculates integer products column-wise [Mon85,Com90]. The code is largely in assembler, with the SSE2 registers used on the Pentium 4 (due to slow arithmetic in general-purpose registers) and general-purpose registers used in the 64-bit case. The SSE2 registers can perform multiplication on vectors of operands up to size 32 bits, but does not possess the carry handling common in general-purpose instruction sets. MIRACL [Sco] uses multiplication on a pair of 32-bit operands, and then performs a "shuffle" to split the 64-bit result across the 128-bit register so that several products can be accumulated. Roughly speaking, the UltraSPARC is treated as a 32-bit processor with a 64-bit accumulator for multiplication, due to limitations on the integer multiplier.[7]

Inversion is via a Euclidean algorithm variant. For the BN case, an inversion has cost equivalent to approximately 85 multiplications on the Core2. Only one or two inversions are performed in the pairings, and so the performance of inversion is not a significant factor.

On processors such as the UltraSPARC and Pentium 4, design "shortcomings" encouraged the use of floating-point hardware for integer multiplication. The strategy isn't

[7]The UltraSPARC has 64-bit addition and corresponding condition codes, but add-with-carry uses the 32-bit condition code [WG94]. Hence, multi-precision addition involves more instructions than on systems with conventional 64-bit carry handling.

new, but the performance obtained by Bernstein [Ber01,Ber06] for point multiplication and other operations was dramatic. The implementation in floating point is decidedly more complicated, in part because the commitment to (redundant) floating point representation is substantial, and bounds conditions on intermediate results must be verified.

The case for floating-point arithmetic on the Core2 and Athon64 is different, due to the existence of a relatively fast 64-bit multiplier. A "quadratic complexity" estimate suggests a factor 4 acceleration in integer multiplication in general purpose registers, although this is admittedly less than convincing. A more complete analysis can be made from experimental data and instruction timings in Table XII.2. Roughly speaking, latency

Table XII.2. Instruction timings for Pentium 4 (32-bit, model 2), Core2 (64-bit), and Athlon64/Opteron (64-bit) [Gra07]. Times are in cycles for latency (L) and throughput (T).

	Pentium 4		Core2		Athlon64	
	L	T	L	T	L	T
add-with-carry	7-8	1/6	2	1	1	2.3
multiply	14	1/10	8	1/4	5	1/2
SSE2 multiply	6	1/2	3	1	3	1/2

is the number of cycles that must pass before the results can be used, and throughput is the number of the instructions that can issue per cycle.[8]

Compared to the Pentium 4, the 64-bit systems have twice the operand size and also significantly better instruction timings for integer operations in general-purpose registers. In contrast, the operand size for floating point and SSE2 multiplication is the same across these systems. On the downside, the 64-bit multiplier retains the restrictive register requirements of the Pentium 4. The floating-point approach can operate (in a stack-based fashion) on any pair of inputs. On the Athlon, floating-point was especially attractive since a multiply and add could be issued in the same cycle. Nonetheless, the smaller operand size is a significant penalty relative to the 64-bit multiplier.

Experimentally, the Core2 can perform multi-precision integer multiplication of 256-bit inputs at a cost of approximately 8 cycles to calculate and accumulate each product of 64-bit inputs. On this system, floating point multiplies can be issued only every two cycles. The analysis is somewhat different on the AMD due to the timings and pipeline properties, but we expect that the 64-bit multiplier will be preferred over floating-point on both these processors.

Finally, we note that SSE2 multiplication is more interesting in the Core2 than the Athlon64 due to integer and SSE2 instruction timings. The SSE2 hardware can in fact perform two 32-bit multiplications per instruction. However, arranging the data for this vector operation is inelegant, and earlier experiments (on a Pentium 4) with Intel's demonstration code were no faster than a scalar approach on 224-bit integers [HMV03, Section 5.4]. We also note that the SSE2 registers do not have the usual carry handling of conventional instruction sets, a consideration in accumulation (the 224-bit example split the input into eight 28-bit segments). Our examination is not conclusive, but we suspect SSE2 will not improve on our timings for integer multiplication on the 64-bit systems.

[8]The definition of throughput is the reciprocal of Intel's use in [Int01]. Under the current definitions, small latency and large throughput are desirable.

4.1.2. Characteristic 2 fields

Most of the material in this section applies more generally, but we will focus on the example field represented as $\mathbb{F}_{2^{1223}} = \mathbb{F}_2[z]/(f)$ where $f(z) = z^{1223} + z^{255} + 1$. As noted in §1.1, square roots are inexpensive in this representation since $\sqrt{c} = \sum c_{2i} z^i + \sqrt{z} \sum c_{2i+1} z^i$ for $c \in \mathbb{F}_{2^{1223}}$, where $\sqrt{z} = z^{612} + z^{128}$. Note that the product in this expression is obtained with a few shifts and additions, and does not require reduction since $\deg \sqrt{z} \leq 612$. The even and odd coefficients in c are extracted simultaneously via lookup on 8 bits.

Several approaches were tested for multiplication. The comb method [LD00] has generally been among the fastest, in part because of reduced shifting and more efficient use of precomputation compared with a traditional Karatsuba-style approach. For efficiency, it is necessary to code for a specific size multiplication (where size is in words), although code expansion can be controlled by using Karatsuba down to a few fixed sizes. For the platforms considered, there are multiple register sizes and we selected comb sizes according to the following table:

Register size	Karatsuba depth	Comb size	Notes
32	2	10	32-bit only
64	1	10	via MMX on Pentium 4
128	1	5	via SSE2

The MMX and SSE2 registers in the table are SIMD, but MMX was introduced earlier in the family while SSE first appeared on the Pentium 3 and the SSE2 extensions on the Pentium 4. Capabilities and instruction timings differ between MMX and SSE2; in particular, the SSE2 registers require multiple instructions to shift across 128 bits unless the amount is divisible by 8. Further, the instruction timings differ between platforms. The Intel and AMD 64-bit offerings have the same instruction timings for logical operations in general-purpose registers, but the difference in the SSE2 timings partially explain our results: the AMD has fastest multiplication with general-purpose registers while Intel can efficiently exploit SSE2.

The multiplication in MIRACL is based on a Karatsuba approach down to word-sized operands where a traditional polynomial multiplier is used with data-dependent precomputation. If the comparison of interest is with general-purpose registers, then combing gives less than 20% improvement to the MIRACL approach. MIRACL optionally implements SIMD register use at the word level (rather than at the size of the SIMD register), giving approximately 20% acceleration on the Pentium 4. Compared to this approach, combing was significantly faster in our tests, with acceleration of 47% and 60% with MMX and SSE2, resp., over combing with general-purpose registers on the Pentium 4.

Coding was primarily in C with intrinsics for the SIMD instructions (see §4.2). Inversion was via a Euclidean algorithm variant, with general-purpose registers only and without aggressive optimizations but with an assembly fragment to aid in finding the degree of a polynomial. Comb width 4 (16 elements of data-dependent precomputation) was used in all the multiplication routines. To exploit the cheaper shifting by multiples of 8 bits in the 128-bit SSE2 registers, we used two passes through the multiplicand with shifts by 8 and a 4-bit shift between passes.

4.1.3. Characteristic 3 fields

The specific example considered is represented as $\mathbb{F}_{3^{509}} = \mathbb{F}_3[z]/(f)$ where $f(z) = z^{509} + r(z)$ is an irreducible pentanomial. There are irreducible trinomials for this extension; however, none give $\sqrt[3]{z}$ with few terms. There is a tetranomial with a 17-term root, but the pentanomials were chosen so that $\deg r$ is at most $509 - 32$ or $509 - 64$, resp., and so that the combined number of terms in $z^{1/3}$ and $z^{2/3}$ is as small as possible (to speed the cube root calculation $\sqrt[3]{c} = \sum c_{3i}z^i + z^{1/3}\sum c_{3i+1}z^i + z^{2/3}\sum c_{3i+2}z^i$). Under these criteria, we looked for examples where f and the roots had terms where the exponents differed by a multiple of the word size. We selected:

	32-bit	64-bit
$f(z)$	$z^{509} - z^{477} + z^{445} + z^{32} - 1$	$z^{509} - z^{318} - z^{191} + z^{127} + 1$
$z^{1/3}$	$z^{361} - z^{329} + z^{297} - z^{202} - z^{170} + z^{43}$	$z^{467} + z^{361} - z^{276} + z^{255} + z^{170} + z^{85}$
$z^{2/3}$	$z^{181} + z^{149} + z^{22}$	$-z^{234} + z^{128} - z^{43}$

These choices have $\deg r$ relatively large, although this is not a concern for our environment.

As in [HPS02], each coefficient $a_i \in \mathbb{F}_3$ is represented uniquely in $\{0, 1, -1\}$ using a pair (a_i^0, a_i^1) of bits, where $a_i = a_i^0 - a_i^1$ and not both bits are 1. Elements a are represented by vectors $a^j = (a_{m-1}^j, \ldots, a_0^j)$, $j \in \{0, 1\}$. Addition $c = a + b$ is

$$t \leftarrow (a^0 \vee b^1) \oplus (a^1 \vee b^0), \quad c^0 \leftarrow (a^1 \vee b^1) \oplus t, \quad c^1 \leftarrow (a^0 \vee b^0) \oplus t.$$

The seven operations involve only bitwise "or" ($\vee$) and "exclusive-or" ($\oplus$), and it is easy to order the instructions to cooperate with processor pipelining. Negation is $-a = (a^1, a^0)$. Techniques from characteristic 2 fields extend directly in this representation; in particular, our multiplication is via comb:

Register size	Karatsuba depth	Comb size	Notes
32	1	8	32-bit only
64	0	8	via MMX on Pentium 4
128	0	4	via SSE2

The "comb size" is in pairs of registers, and a depth of 0 means that combing was on field elements.

Coding considerations are similar to those of characteristic 2. However, addition was written in assembly in order to coerce better sequences from compilers. Comb multiplication in SSE2 uses the same strategy to exploit faster shifting by multiples of 8; however, our comb width of 3 (27 points of data-dependent precomputation) means that there are more passes and "fixups" than in the binary case.

Precomputation is less expensive than it appears, since half the elements are obtained by simple negation. We note that Takahashi, Hoshino, and Kobayashi [THK07] report substantial savings from sharing precomputations from $\mathbb{F}_q$ multiplications in fg of Step 5 in Algorithm XII.2. However, the amount of re-use is for approaches using more than the 12 multiplications described in §1.2, and the proportion of re-use decreases as the number of multiplications for fg decreases. For the approach with 15 multiplications in fg, only 7 precomputations are done, saving a reported 25% in this product. Their multiplication is for $q = 3^{97}$, and their precomputation cost, as a proportion of an $\mathbb{F}_q$ product, is higher than our estimates for $q = 3^{509}$. A more direct proposal to accelerate Algorithm XII.2 appears in [BSTO07b,BBD$^+$07], where an "unrolling" technique adapted from [GPS06] reduces the multiplication count in the main loop from 14 to 12.5 (and with 11 cubings where Algorithm XII.2 has 2 cubings and 2 roots).

The reduction polynomial is more favourable in the 32-bit case than in the 64-bit case, since there are several exponents that differ by a multiple of 32; see also [Sco07b] for related material. As an alternative, the pairing algorithm can be constructed so that root calculations are avoided [BSTO07a], in which case a trinomial can be selected.

4.2. Development Environment

Development was done on a variety of systems, including Sun Solaris (Blade 2000 with UltraSPARC III and X4200 with Opteron), Linux/x86, and Microsoft Windows with the Sun Studio (5.9), GNU C (gcc 3.4 and 4.1), Intel (6.0, 32-bit only), and Microsoft (6.0) tools. Compilers exhibit a frustrating amount of sensitivity to the precise way the code is written and can produce object code of quite different quality. The use of multiple compilers provided useful sanity tests and debugging help.

Most of the code was written in C with some assembly language for critical sections and to work around compiler shortcomings; an exception is the prime field arithmetic where substantial portions are in assembler. The code for small-characteristic fields involving SIMD registers was primarily with compiler intrinsics. Roughly speaking, programming with intrinsics is similar to assembly language, but register allocation is managed by the compiler and code optimizations can be performed. This is a double-edged sword, however, and compilers sometimes substitute awful code sequences rather than a direct translation. For example, we were surprised to see that gcc 4.1 and 4.2 produced dramatically different code for shifting sequences involving SSE2 intrinsics in 64-bit code, where version 4.2 chose an expensive strategy involving moves to conventional registers and double-register shifts.

We chose the GNU compilers (gcc) for the timings because these are widely available across systems and have been a common choice for benchmarks. These compilers generally produce fairly good code and have an excellent interface for insertion of assembly language fragments (adopted by Intel and recent Sun compilers). Compared to the Sun compiler, gcc can require more code tuning to coerce better sequences and register allocation. In particular, gcc does not optimize as well when the data is written in structures or arrays (even when the indices are known at compile-time), and we tuned critical sections for gcc by breaking aggregates into scalars. The Sun and Intel compilers are much less sensitive to this scalar-vs-aggregate issue.

4.3. Timings

The estimates in §1 and §2 obtained by counting field multiplications ignore other operations and overheads. These omissions can be significant and also hard to estimate, and so experimental data is often an important part of the analysis. Field and pairing timings appear in Table XII.3. The times are given in units involving the clock speed; to obtain the elapsed time for the test machines, divide by 2.8 GHz for the Pentium 4 and Opteron, 2.4 GHz for the Core2, and 1.2 GHz for the UltraSPARC.

Table XII.3. Timings (in clock cycles) for field operations and pairings on a Pentium 4 model 2, Opteron, and Core2. Registers for field multiplication are GP (general-purpose), MMX (64-bit SIMD), and SSE (128-bit SIMD, SSE2 extensions). τ denotes the field characteristic. Compilers are Sun 5.9 on UltraSPARC and GNU 4.1 on the others.

	Multiplication						η_T/Ate		R-ate	
Field	GP	MMX	SSE	a^τ	$\sqrt[\tau]{a}$	a^{-1}	GP	SSE	GP	SSE
32-bit, Pentium 4										
$\mathbb{F}_{p_{256}}$	2.2		1.4	—	—	56	81	58	54	38
$\mathbb{F}_{2^{1223}}$	43.4	23.0	16.2	1.7	1.1	627	201	81	—	—
$\mathbb{F}_{3^{509}}$	32.5	22.1	19.0	2.0	5.0	518	139	86	—	—
64-bit, Opteron										
$\mathbb{F}_{p_{256}}$	0.25			—	—	32	15		10	
$\mathbb{F}_{2^{1223}}$	10.4	13.4	14.6	0.7	0.6	200	48		—	—
$\mathbb{F}_{3^{509}}$	10.6	12.9	15.1	1.1	1.3	116	46		—	—
64-bit, Core2										
$\mathbb{F}_{p_{256}}$	0.31			—	—	25	15		10	
$\mathbb{F}_{2^{1223}}$	10.3	12.5	8.2	0.6	0.5	162	48	39	—	—
$\mathbb{F}_{3^{509}}$	10.8	11.0	7.7	0.9	1.2	98	46	33	—	—
UltraSPARC III										
$\mathbb{F}_{p_{256}}$	2.4			—	—	40	77		52	
$\mathbb{F}_{2^{1223}}$	11.8			0.8	0.7	217	57		—	—
$\mathbb{F}_{3^{509}}$	14.5			1.4	1.8	145	63		—	—
Units:			10^3 cycles					10^6 cycles		

The entries involving use of SIMD facilities require some explanation. On the Pentium 4, the 128-bit SSE2 capabilities give significant improvement across the three fields. However, the improvement for characteristic 3 is less dramatic than for characteristic 2. As discussed in §4.1.3, a portion of this difference is explained by restrictions on shifts in these registers. The current implementation has some assembly language enhancements, but also has has excessive memory operations and more careful tuning of this code could offer some acceleration. The Core2 timings for small characteristic illustrate the faster SSE2 operations relative to the Opteron (where SSE2 was not effective). Although the entries for SIMD on the UltraSPARC are empty, the processor does in fact have such capabilities in the VIS instruction set. Unlike SSE2, these registers are 64-bit and the multiplication is 8-bit by 16-bit, limiting their usefulness in our context. The MMX entries for prime fields are also omitted since these registers do not possess the 32-bit multiplier introduced in the SSE2 extensions.

We consider the ratio of the estimated time from multiplication counts over the actual time (and so ratios much less than 1 indicate that significant time is not represented in the counts). For pairing in the characteristic 2 case (where field additions are very inexpensive compared to multiplication), the ratio is approximately 0.9 across systems when multiplication is via general-purpose registers, and falls to 0.8 when multiplication is via SIMD (in the cases where SIMD is advantageous). Part of the explanation for this difference is that SIMD was applied only to field multiplication, and other operation costs remain unchanged. For the characteristic 3 case, the ratio falls to 0.76–0.8 depending on registers employed.

The more troublesome case is for BN, where field additions are now a more significant part of the overall cost of the pairing. For the Tate pairing, if the fastest multiplication is considered on the Pentium 4, then the ratio of interest drops to 0.62. On the Core2, it is even lower at 0.52. However, good estimates from operation counts can be obtained if field additions are included—the corresponding ratios rise to 0.9 and 0.83, resp. As illustration, the code from [DSD07] on a 1.66 GHz Core2 performs a Tate pairing in 14.5 ms, using 39824 field multiplications and 132893 additions. In isolation, the multiplications run at 7.6 ms and the additions at 4.5 ms.

The relatively strong showing for small characteristic on the UltraSPARC is due primarily to a weak integer multiplier that is restricted to 64-bit output. The other data is cycle-competitive with the newer Opteron and Core2 designs, although it should be noted that the UltraSPARC III was typically at half the clock speed of common Opteron and Core2 systems.

At the 128-bit security level, the R-ate pairing for BN curves is also substantially faster than other pairings on elliptic curves defined over large prime fields, including the MNT curves [MNT01]. A recent IETF standard [BM07] for identity-based encryption mandates the use of supersingular elliptic curves over prime fields—these curves have embedding degree $k = 2$. Our implementation of the Tate pairing on such an elliptic curve E defined over a 1536-bit prime field $\mathbb{F}_p$ (and where $\#E(\mathbb{F}_p)$ has a 256-bit prime divisor) took 111 ms on a 32-bit 3GHz Pentium 4, versus versus 14.2 ms for the R-ate pairing on the BN curve of §2.

5. Summary

In this chapter, we have attempted to quantify pairing performance on popular systems at the 128-bit security level. The selected platforms have relatively fast integer multiplication and no special support for small characteristic fields. As expected, the BN curves hold a substantial edge, thanks to fast multiplication, "ideal" match of embedding degree and security level, and accelerations in the R-ate algorithm.

However, the difference is smaller than earlier reports may have suggested. In some cases, this has been due in part to overly pessimistic timings for characteristic 2 and especially characteristic 3 fields. Estimates based on multiplication counts can also be misleading. Operation counts often provide meaningful relative comparisons, even if fairly rough when used to estimate actual performance. However, the common comparison involving multiplications gets coarser when other operations have very different cost relative to field multiplication (as in the case of multiplication to addition for prime fields versus characteristic 2 fields). This kind of estimate inflates the advantage of the BN case over characteristic 2 and 3.

We remark briefly on extrapolating from the experimental data. A common practice is to implement in the general-purpose register set, under the assumption that results may be more widely applicable across "similar" systems. This of course gets the absolute performance wrong, but may adequately capture relative differences. As an example, for characteristic 3 on the Pentium 4, the restriction to general-purpose registers gives roughly the same "BN vs. small characteristic" conclusions as allowing SIMD. On the other hand, this characterization is wrong for the preceding generation Pentium III, where there is a relatively fast integer multiplier (but not the SSE2 32-bit multiplication of the Pentium 4) and only small characteristic benefits from SIMD. In the Core2 vs. Opteron comparison, we see that the Opteron has faster integer multiplication, and the two are similar when using general-purpose registers for small characteristic. However, the Core2 has faster SIMD timings that can be exploited for small characteristic. Necessarily, the small characteristic case looks somewhat less bleak against BN on the Core2 if SIMD is permitted.

On other devices, small characteristic can of course look more attractive. For example, favourable reviews for characteristic 2 implementations on smartcards and wireless sensor networks can be found in [SCA06] and [SOS$^+$08]. As noted earlier, the addition of a hardware characteristic 2 polynomial multiplier could dramatically change the performance. Intel has announced a characteristic 2 multiplier for its next generation processors, with multiplication on 64-bit operands [GK08]. Implementing small characteristic across systems is arguably easier, although admittedly the amount of “difficult” coding can typically be limited to a fairly small portion of the arithmetic and still exploit most of the available performance with a fairly generic approach. Small characteristic also seems to lead to smaller code sizes.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-207

Chapter XIII

Hardware Implementation of Pairings

Maurice KELLER, Robert RONAN, Andrew BYRNE,
Colin MURPHY and William MARNANE
University College Cork, Ireland

Abstract. In this chapter the efficient hardware implementation of pairings is considered. For each of the three fields $\mathbb{F}_{2^m}$, $\mathbb{F}_{3^m}$ and $\mathbb{F}_p$ suitable curves, pairing algorithms and field arithmetic architectures are presented. An example architecture for computing the η_T pairing in characteristic 2 or 3 is presented and its implementation results on a Xilinx FPGA considered. Finally the implementation results of several pairing architectures from the literature are discussed.

Keywords. Tate pairing, η_T, BKLS, hardware implementation, parallel arithmetic, FPGA

The efficient implementation of bilinear pairings is vital if Identity Based Cryptography is to be successful in real-world scenarios. Even with the many algorithmic optimizations to date, pairing calculation remains a relatively complicated and time consuming operation. Dedicated hardware architectures can provide a large acceleration in calculation time when compared to implementations on general purpose serial processors.

The main advantage of using dedicated hardware to accelerate cryptographic operations is that it enables the exploitation of wide scale parallelism within the underlying algorithms. Pairings are highly suited to hardware implementation as there are two different types of parallelism possible. Firstly, underlying Galois field arithmetic operations can be performed in parallel, such as field multiplications. Secondly, in general, the main computational cost of pairing algorithms is an iterative loop. With appropriate scheduling it is possible to *unroll* the loop and perform operations on different iterations of the loop in parallel.

Another reason for using dedicated hardware is that it can provide a large degree of reconfigurability, especially when a device such as a Field Programmable Gate Array (FPGA) [Xil] is used. This reconfigurability can come in several guises. The number and structure of the underlying Galois field components within a design can be varied to offer a trade off between low power, low area implementations suitable for the client side of a protocol, and high throughput, larger implementations more suited to a server environment. Using a reconfigurable architecture and platform also allows a user to quickly and easily change the underlying mathematical parameters, such as the field size and resultant security of the system. This can be highly advantageous as the minimum field size considered to be secure is constantly changing as attacks become more powerful.

1. Algorithms for Computing the Tate Pairing

The Tate pairing can be computed using Miller's algorithm (Algorithm 1, Chapter II). Since constructive uses of pairings in cryptography were first suggested by Joux in 2000 [Jou00] there have been several improvements to Miller's algorithm.

The BKLS/GHS algorithm incorporates the improvements to Miller's basic algorithm suggested by Barreto et al. in [BKLS02] and Galbraith et al. in [GHS02]. It uses the distortion map ϕ introduced in Chapter II to map an element of $E(\mathbb{F}_q)[l]$ to an element of $E(\mathbb{F}_{q^k})[l]/E(\mathbb{F}_q)[l]$. The BKLS/GHS algorithm improves Miller's version by allowing the evaluation of the line functions l and v in terms of the point Q rather than a divisor. It also removes the need to evaluate the denominators in Miller's algorithm when updating f. The final exponentiation is introduced in order to produce a unique value for cryptographic applications. The runtime of the algorithm depends on the total number of bits and the hamming weight of the curve order. The curve order will be comparable in size to the size of the underlying field F_q. Therefore the runtime of the BKLS algorithm depends on the size of the underlying field.

Duursma and Lee [DL03] decoupled the pairing computation from the group law on hyperelliptic curves. This resulted in a more efficient algorithm for computing the Tate pairing as it reduces the number of underlying field operations that are necessary. Barreto et al. generalized the Duursma-Lee (DL) algorithm to elliptic curves in [BGhS07]. The general form of the DL algorithm is referred to as the η pairing. The loop in the η algorithm runs for the same number of iterations as the BKLS algorithm, however, it requires less field operations per iteration. The η_T [BGhS07] algorithm is an improvement to that proposed by Duursma and Lee. The main advantage is that it halves the loop length, thereby significantly reducing the computation time. As is the case with the BKLS algorithm, a final exponentiation is performed when evaluating the η_T pairing to obtain a unique value for cryptographic applications.

The output of the η_T algorithm is a bilinear, degenerate pairing value suitable for implementing pairing based protocols. Beuchat et al. [BBD$^+$08] took advantage of the bilinearity of the pairing and showed that the Tate pairing can be obtained from the η_T pairing at the cost of a few extra operations in $\mathbb{F}_q$. An important note is that the η and η_T algorithms work only for supersingular curves since they depend on the existence of the distortion map described in Chapter II.

The Ate pairing [HSV06] can shorten the length of the Miller loop on certain ordinary elliptic curves in characteristic p. The Ate pairing has a similar structure to the BKLS algorithm i.e. group operation on the elliptic curve plus additional operations to evaluate the pairing value. It has been shown that the Ate pairing and its derivatives are at least as efficient as computing the Tate pairing using BKLS [MKHO07]. However, in certain special cases it can provide an improvement. There has been no hardware implementation of the Ate pairing to date in the literature. Therefore its efficiency in hardware is unknown.

More details on the mathematical background and derivation of the algorithms can be found in Chapter II.

2. Tate Pairing over Supersingular Curves in Characteristic 2

2.1. Curve

A supersingular elliptic curve in characteristic 2 is defined by Equation (1),

$$E_2(\mathbb{F}_{2^m}) : y^2 + y = x^3 + x + b,\ b \in \{0, 1\}\ . \tag{1}$$

The group law on $E_2(\mathbb{F}_{2^m})$ is performed as per (2) and (3)

$$\begin{array}{l} P_2 = P_0 + P_1 \\ \lambda = \frac{y_0+y_1}{x_0+x_1} \\ x_2 = \lambda^2 + x_1 + x_0 \\ y_2 = \lambda(x_0 + x_2) + y_0 + 1 \end{array} \quad (2) \qquad \begin{array}{l} P_2 = [2]P_0 = P_0 + P_0 \\ \lambda = {x_o}^2 + 1 \\ x_2 = \lambda^2 \\ y_2 = x_0^4 + y_0^4 \end{array} \quad (3)$$

in affine coordinates. Addition requires one field multiplication and one field division. Mixed coordinates can be used to perform the addition using nine field multiplications [HMV03] but without the need for an expensive field division.

The curve in Equation (1) has order $\#E_2 = 2^m + \nu 2^{\frac{m+1}{2}} + 1$, where ν is given by

$$\nu = \begin{cases} (-1)^b, & m \bmod 8 = 1, 7 \\ (-1)^{1-b}, & m \bmod 8 = 3, 5 \end{cases}$$

and embedding degree $k = 4$. The distortion map ϕ for the curve defined in Equation (1) is $\phi(x, y) = (x + s^2, y + sx + t)$ where $s, t \in \mathbb{F}_{2^{4m}}$, $s^2 = s + 1$ and $t^2 = t + s$.

2.2. Characteristic 2 Finite Field Arithmetic

Addition in $\mathbb{F}_2 = \{0, 1\}$ is implemented using an XOR gate. Multiplication in $\mathbb{F}_2$ corresponds to an AND gate. This direct and efficient implementation of $\mathbb{F}_2$ arithmetic in hardware is the main advantage of using $\mathbb{F}_{2^m}$. Elements of the field $\mathbb{F}_{2^m}$ can be represented in polynomial basis polynomials of degree $(m - 1)$ with coefficients in $\mathbb{F}_2$. Other possible basis include dual and normal basis [MBG+93].

Addition is performed coefficient wise over $\mathbb{F}_{2^m}$ using m *xor*-gates and has a latency of less than one clock cycle. Addition and subtraction are equivalent in $\mathbb{F}_{2^m}$. Squaring can also be performed using only combinatorial logic and can be completed in less than one clock cycle using a bit-parallel squaring architecture [OP00].

There are many different architectures suitable for implementing multiplication in $\mathbb{F}_{2^m}$. Bit-serial architectures [TSP86,Mas91] offer a very low area solution. The disadvantage is that m clock cycles are required to compute the result. Bit-parallel architectures [Mas91,Paa94] utilize maximum parallelism to compute the result in one clock cycle at the cost of a very large circuit area. One of the most popular multiplier types is the digit-serial architecture of the type described in [SP98], as it allows an area/speed trade off to be explored between the two extremes of the bit-serial and bit-parallel architectures. In a digit-serial multiplier, D coefficients of the input polynomial are operated on in parallel, where D is known as the *digit size*. Multiplication is completed in $n = \lceil m/D \rceil$ clock cycles. As D increases, so too does the area of the multiplier, providing an area/speed trade-off.

$\mathbb{F}_{2^m}$ inversion can be performed efficiently in hardware using the Extended Euclidean Algorithm (EEA) [Ber68]. An example of an EEA based inverter can be found in [BCH93]. An inversion takes $2m$ clock cycles. On some occasions a field division is required. This can be done using an inversion followed by a multiplication. A dedicated EEA based divider such as one based on the algorithm described in [Sha01] computes the result in $2m$ clock cycles and can be more efficient than an inverter when division is required. A more area efficient option for performing inversion is the algorithm of Itoh-Tsujii [IT88], which allows the inversion to be computed using repeated multiplications and squarings.

Table XIII.1 gives implementation results for the basic $\mathbb{F}_{2^m}$ arithmetic operations on a Xilinx Virtex-II Pro FPGA. The area is measured in terms of both slices and LUTs (look up tables). On a Virtex-II Pro device each slice contains two four input LUTs which can be programmed to implement any four input logic function. It is noted that both addition and squaring are combinational circuits and hence have very low latency. The low area of the squarer is due to the low complexity of the irreducible trinomial used ($f(x) = x^{313} + x^{79} + 1$). The squarer requires only 195 two input XOR gates compared to the adders 313 gates.

Table XIII.1. $\mathbb{F}_{2^m}$ arithmetic architecture examples implemented on a Virtex-II Pro FPGA, $m = 313$.

Architecture	Slices	LUTs	Freq. (MHz)	Clock Cycles	Time (ns)
Adder	313	313	649.8	1	1.54
Digit-Serial Mult, $D = 1$	650	971	278.6	313	1123.48
Digit-Serial Mult, $D = 8$	1613	2840	213.1	40	187.71
Bit-Parallel Squarer	117	156	400.6	1	2.50
EEA Divider	1592	2223	233.1	626	2685.54

Arithmetic in the extension field $\mathbb{F}_{2^{4m}}$ is naturally more complex than that in the base field $\mathbb{F}_{2^m}$. However, by choosing an appropriate basis for the extension field, the $\mathbb{F}_{2^{4m}}$ operations can be broken down into operations in $\mathbb{F}_{2^m}$. One popular basis is $\{1, s, t, st\}$ i.e. $A = a_0 + a_1 s + a_2 t + a_3 st \in \mathbb{F}_{2^{4m}}$, where s and t are given by the distortion map ϕ. The cost of the arithmetic operations in $\mathbb{F}_{2^{4m}}$ is summarized in Table XIII.2.

Table XIII.2. Arithmetic operations over $\mathbb{F}_{2^{4m}}$.

Op	Method	Cost on $\mathbb{F}_{2^m}$
$A + B$	$(a_0 + b_0, a_1 + b_1, a_2 + b_2, a_3 + b_3)$	4 adds
A^2	$({a_0}^2 + {a_1}^2 + {a_3}^2, {a_1}^2 + {a_2}^2, {a_2}^2 + {a_3}^2, {a_3}^2)$	4 sq, 4 adds
$A * B$	Karatsuba methods are used to trade $\mathbb{F}_{2^m}$ multiplications against more trivial additions. All $\mathbb{F}_{2^m}$ multiplications can be performed in parallel if desired [BBD+08]	9 muls, 20 adds

The η_T algorithm also requires arithmetic in $\mathbb{F}_{2^{2m}}$ in order to efficiently perform the final exponentiation. $A \in \mathbb{F}_{2^{2m}}$ is represented as $A = a_0 + a_1 s$. The arithmetic operations over $\mathbb{F}_{2^{2m}}$ are summarized in Table XIII.3.

Table XIII.3. Arithmetic operations over $\mathbb{F}_{2^{2m}}$.

Op	Method	Cost on $\mathbb{F}_{2^m}$
$A+B$	(a_0+b_0, a_1+b_1)	2 adds
A^2	$({a_0}^2+{a_1}^2, a_1^2)$	2 squares, 1 adds
$A*B$	$(a_0b_0+a_1b_1, (a_o+b_0)(b_0+b_1)+a_0b_0)$ [BBD+08]	3 muls, 4 adds
A^{-1}	$(w^{-1}(a_0+a_1), w^{-1}a_1)$ where $w = a_0^2+(a_0+a_1)a_1$ [BBD+08]	3 muls, 2 adds, 1 sq., 1 inv

2.3. Pairing Algorithm

The η_T [BGhS07] algorithm is the most efficient method of computing the Tate pairing in characteristic 2. The reversed loop η_T pairing is given by:

$$\eta_T(P,Q) = l_{P'}(\phi(Q)) \prod_{j=0}^{\frac{m-1}{2}} \left(g_{\left[2^{\frac{m-1}{2}-j}\right]P'}(\phi(Q)) \right)^{2^j}$$

where $P' = [-\nu]P$, $g_V(x,y) =$ is the rational function defined over $E(\mathbb{F}_{2^{4m}})[l]$ corresponding to the doubling of V, and $l_V(x,y)$ is the equation of the line corresponding to the addition of $\left[2^{\frac{m+1}{2}}\right]V$ and $[\nu]V$. This can be computed using Algorithm XIII.1 [BBD+08] where

$$\delta = \begin{cases} b, & m \bmod 8 = 1,7 \\ 1-b, & \text{otherwise} \end{cases}$$

$$\alpha = \begin{cases} 0, & m \bmod 4 = 3 \\ 1, & m \bmod 4 = 1 \end{cases} \qquad \beta = \begin{cases} b, & m \bmod 8 = 1,3 \\ 1-b, & m \bmod 8 = 5,7 \end{cases}.$$

Algorithm XIII.1 Characteristic 2 reversed loop η_T pairing with square roots

```
Input: $P, Q \in E(\mathbb{F}_{2^m})[l]$.
Output: $\eta_T(P,Q) \in \mathbb{F}^*_{2^{4m}}$.
 1: $y_p = y_P + (1-\delta)$; $u = x_P + \alpha$; $v = x_Q + \alpha$                       – 3 add
 2: $g_0 = uv + y_P + y_Q + \beta$; $g_1 = u + x_Q$; $g_2 = v + x_P^2$   – 1 mul, 1 square, 5 adds
 3: $G = g_0 + g_1 s + t$
 4: $L = (g_0 + g_2) + (g_1 + 1)s + t$                                                   – 2 adds
 5: $F = L.G$                                                          – 2 mul, 1 square, 7 adds
 6: for $j$ from 1 to $\frac{m-1}{2}$ do
 7:     $x_P = \sqrt{x_P}$; $y_P = \sqrt{y_P}$; $x_Q = x_Q^2$; $y_Q = y_Q^2$          – 2 roots, 2 squares
 8:     $u = x_P + \alpha$; $v = x_Q + \alpha$                                            – 2 adds
 9:     $g_0 = uv + y_P + y_Q + \beta$; $g_1 = u + x_Q$                             – 1 mul, 4 adds
10:     $G = g_0 + g_1 s + t$
11:     $F = F.G$                                                     – sparse $\mathbb{F}_{2^{4m}}$ mul
12: return $F^M$.
```

Algorithm XIII.1 requires square root computation. This computation can be expensive in hardware and its complexity depends heavily on the particular irreducible polynomial in use. However, the field square root computation can be replaced by field squarings by taking advantage of the bilinearity of the pairing as shown in Equation (4),

$$\begin{aligned}\eta_T(P,Q) &= \eta_T\left(\left[2^{-\frac{m-1}{2}}\right]P,Q\right)^{2^{\frac{m-1}{2}}} \\ &= l_{\left[2^{-\frac{m-1}{2}}\right]P'}(\phi(Q))^{2^{\frac{m-1}{2}}} \prod_{j=0}^{\frac{m-1}{2}} \left(\left(g_{[2^{-j}]P'}(\phi(Q))\right)^{2^{2j}}\right)^{2^{\frac{m-1}{2}-j}} \quad .\end{aligned} \tag{4}$$

This can be computed using Algorithm XIII.2 [BBD+08].

Algorithm XIII.2 Characteristic 2 reversed loop η_T pairing without square roots.

```
Input: $P, Q \in E(\mathbb{F}_{2^m})[l]$.
Output: $\eta_T(P,Q) \in \mathbb{F}^*_{2^{4m}}$.
 1: $y_p = y_P + \delta;\ x_P = x_P^2;\ y_P = y_P^2$          – 2 squares, 1 add
 2: $y_P = y_P + b;\ u = x_P + 1;\ g_1 = u + x_Q$             – 3 adds
 3: $g_0 = x_P x_Q + y_P + y_Q + g_1$                         – 1 mul, 3 adds
 4: $x_Q = x_Q + 1;\ g_2 = x_P^2 + x_Q$                       – 1 square, 2 add
 5: $G = g_0 + g_1 s + t$
 6: $L = (g_0 + g_2) + (g_1 + 1)s + t$                        – 2 adds
 7: $F = L.G$                                                 – 2 mul, 1 square, 7 adds
 8: for $j$ from 1 to $\frac{m-1}{2}$ do
 9:     $F = F^2$                                             – $\mathbb{F}_{2^{4m}}$ square
10:     $x_Q = x_Q^4;\ y_Q = y_Q^4$                           – 4 squares
11:     $x_Q = x_Q + 1;\ y_Q = y_Q + x_Q$                     – 2 adds
12:     $g_0 = u x_Q + y_P + y_Q;\ g_1 = x_P + x_Q$           – 1 mul, 3 adds
13:     $G = g_0 + g_1 s + t$
14:     $F = F.G$                                             – sparse $\mathbb{F}_{2^{4m}}$ mul
15: return $F^M$.
```

The extension field multiplication $F = F.G$ required in both Algorithm XIII.1 and XIII.2 can be computed in only six $\mathbb{F}_{2^m}$ multiplications and fourteen $\mathbb{F}_{2^m}$ additions due to the sparseness of the G operand [BBD+08].

The output of both Algorithm XIII.1 and XIII.2 must be reduced in order to produce a unique result for cryptographic applications. This is achieved via the final exponentiation to the power $M = (2^{2m}-1)(2^m - \nu 2^{\frac{m+1}{2}} + 1)$. Beuchat et al. [BBD+08] showed that this could be performed efficiently by computing

$$\eta_T(P,Q)^M = \left(\eta_T(P,Q)^{2^{2m}-1}\right)^{2^m+1}\left(\eta_T(P,Q)^{\nu(1-2^{2m})}\right)^{2^{\frac{m+1}{2}}} \tag{5}$$

Letting $F = \eta_T(P,Q) \in \mathbb{F}^*_{2^{4m}}$ and writing $F = F_0 + F_1 t$, where F_0 and $F_1 \in \mathbb{F}_{2^{2m}}$ then $F^{2^{2m}-1}$ and $F^{1-2^{2m}}$ are given by Equations (6) and (7) respectively.

$$F^{2^{2m}-1} = \frac{F_0^2 + F_1^2 + F_1^2 s + F_1^2 t}{F_0^2 + F_0F_1 + F_1^2 s} \quad (6) \qquad F^{1-2^{2m}} = \frac{F_0^2 + F_1^2 s + F_1^2 t}{F_0^2 + F_0F_1 + F_1^2 s} \quad . \quad (7)$$

Since $F_0^2 + F_0F_1 + F_1^2 s \in F_{2^{2m}}$ this requires a single inversion over $F_{2^{2m}}$.

Full details of the final exponentiation calculation can be found in [BBD$^+$08]. The total cost of the final exponentiation is 17 multiplications, 1 inversion, 7 squarings and 31 additions over $\mathbb{F}_{2^m}$ and $(\frac{m+1}{2})$ squarings and 1 multiplication over $\mathbb{F}_{2^{4m}}$.

One common method of improving the performance of hardware implementations is to use loop unrolling. Unfortunately such loop unrolling was found not to provide any advantage when using the η_T algorithm in characteristic 2 [BBD$^+$08].

The Tate pairing $t(P, Q)$ can be computed using an η_T accelerator using just a few additional $\mathbb{F}_{2^m}$ operations by noting that $t(P, Q) = \eta_T([2^m]P, Q)$ [BBD$^+$08]. The only additional computation required is $[2^m]P = (x_P + 1, x_P + y_P + \alpha + 1)$. This requires only $\mathbb{F}_{2^m}$ additions and could be performed as a preprocessing step in software.

3. Tate Pairing over Supersingular Curves in Characteristic 3

3.1. Curve

A supersingular elliptic curve in characteristic 3 is defined by Equation (8)

$$E_3(\mathbb{F}_{3^m}) : y^2 = x^3 - x + b,\ b \in \{-1, 1\}\ . \tag{8}$$

The group law on this elliptic curve is given by Equations (9) and (10). The field division can be avoided by using projective coordinates [HPS02]. In this case a point addition and doubling will require twelve and seven multiplications respectively.

$$\begin{array}{l} P_2 = P_0 + P_1 \\ \lambda = \frac{y_0 - y_1}{x_0 - x_1} \\ x_2 = \lambda^2 - (x_0 + x_1) \\ y_2 = (y_0 + x_1) - \lambda^3 \end{array} \quad (9) \qquad \begin{array}{l} P_2 = [2]P_0 = P_0 + P_0 \\ \lambda = \frac{1}{y_0} \\ x_2 = \lambda^2 + x_0 \\ y_2 = -(y_0 + \lambda^3) \end{array} \quad . \quad (10)$$

The curve in Equation (8) has order $\#E_3 = 3^m + \gamma 3^{\frac{m+1}{2}} + 1$, where γ is given by equation (11), and embedding degree $k = 6$. The distortion map ϕ for the curve defined in equation (8) is $\phi(x, y) = (\rho - x, \sigma y)$ where $\rho, \sigma \in \mathbb{F}_{3^{6m}}$, $\sigma^2 = -1$ and $\rho^3 = \rho + b$.

$$\gamma = \begin{cases} b, & m \bmod 12 = 1, 11 \\ -b, & m \bmod 12 = 5, 7 \end{cases} . \tag{11}$$

3.2. Characteristic 3 Finite Field Arithmetic

Implementing the η_T algorithm over $\mathbb{F}_{3^m}$ requires the basic arithmetic operations in $\mathbb{F}_{3^m}$: addition, subtraction, multiplication, inversion and cubing. The field F_3 consists of three elements $\{0, 1, 2\}$, therefore, each element requires two binary bits to represent it. The circuits to perform addition, subtraction and multiplication of elements of F_3 depend on the choice of representation. Using the mapping $0 = \{0X\}, 1 = \{10\}$ and $2 = \{11\}$, for example, an $\mathbb{F}_3$ addition requires nine AND gates, five OR gates and three NOT gates. Clearly this is more complex than $\mathbb{F}_2$ addition. However, on Xilinx FPGA technology the additional complexity is offset somewhat by the fact that $\mathbb{F}_3$ addition and multiplication both map efficiently into two 4:1 LUTs.

Arithmetic in $\mathbb{F}_{3^m}$ can be performed using the same type of architectures as for $\mathbb{F}_{2^m}$. The difference is that addition and multiplication of the base elements in $\mathbb{F}_3$ are more complex than addition and multiplication of elements of $\mathbb{F}_2$. Therefore, arithmetic in $\mathbb{F}_{3^m}$ is more complex than in $\mathbb{F}_{2^m}$. The larger embedding degree and additional complexity mean that for a given security level, the value of m required in characteristic 3 will be lower than that required in characteristic 2. For example, pairing based systems over $\mathbb{F}_{3^{97}}$ and $\mathbb{F}_{2^{271}}$ are both considered to give security equivalent to an 80 bit symmetric cipher.

Table XIII.4 gives implementation results for sample $\mathbb{F}_{3^m}$ arithmetic units. Compared to the characteristic 2 case it can be seen that the $\mathbb{F}_{3^m}$ operations require additional area due the complexity of the underlying F_3 arithmetic. This complexity also reduces the clock frequency and hence the computation time is higher for $\mathbb{F}_{3^m}$. Note that the inverter requires a significant amount of FPGA resources. Fermat's Little theorem can also be used in characteristic 3 to perform the inversion using repeated multiplications and cubings [BBS$^+$08].

Table XIII.4. $\mathbb{F}_{3^m}$ arithmetic architecture examples implemented on a Virtex-II Pro FPGA, $m = 97$.

Architecture	Slices	LUTs	Freq. (MHz)	Clock Cycles	Time (ns)
Adder	97	194	500.5	1	2.00
Subtracter	97	194	488.8	1	2.05
Digit-Serial Mult, $D = 1$	712	1002	156.5	97	619.82
Digit-Serial Mult, $D = 8$	2267	3807	88.2	13	147.39
EEA Inverter	2327	4391	126.7	194	1531.04

Implementing the η_T algorithm in characteristic 3 also requires arithmetic in $\mathbb{F}_{3^{6m}}$. As in the characteristic 2 case, operations in $\mathbb{F}_{3^{6m}}$ can be broken down into operations in the base field $\mathbb{F}_{3^m}$. The distortion map allows the elements of $\mathbb{F}_{3^{6m}}$ to be represented as an extension of $\mathbb{F}_{3^m}$ using the basis $\{1, \sigma, \rho, \sigma\rho, \rho^2, \sigma\rho^2\}$. The cost of the various $\mathbb{F}_{3^{6m}}$ operations are summarized in Table XIII.5.

Table XIII.5. Arithmetic operations over $\mathbb{F}_{3^{6m}}$.

Op	Method	Cost on $\mathbb{F}_{3^m}$
$A + B$	$(a_0 + b_0, a_1 + b_1, a_2 + b_2, a_3 + b_3)$	4 adds
$A - B$	$(a_0 - b_0, a_1 - b_1, a_2 - b_2, a_3 - b_3)$	4 subs
A^3	$(a_0{}^3 + ba_2{}^3 + a_4{}^3, -(a_1{}^3 + ba_3{}^3 + a_5{}^3), a_2{}^3 - ba_4{}^3, ba_5{}^3 - a_3{}^3, a_4{}^3, -a_5{}^3)$	6 cubes, 8 adds/subs
$A * B$	Method based on the Fast Fourier Transform. See [GPS07] for more details.	15 muls, 67 adds/subs

The η_T algorithm also requires arithmetic in $\mathbb{F}_{3^{3m}}$ in order to evaluate the final exponentiation efficiently. An element of $\mathbb{F}_{3^{3m}}$ is represented as $A = a_0 + a_1\rho + a_2\rho^2 \in \mathbb{F}_{3^{3m}}$. The arithmetic operations required in $\mathbb{F}_{3^{3m}}$ are summarized in Table XIII.6.

Table XIII.6. Arithmetic operations over $\mathbb{F}_{3^{3m}}$.

Op	Method	Cost on $\mathbb{F}_{3^m}$
$A * B$	$(b(ba_1+a_2)(b_1+bb_2)+a_0b_0-a_1b_1-a_2b_2, (a_0+a_1)(b_0+b_1)+(ba_1+a_2)(b_1+bb_2)-a_0b_0-(b+1)a_1b_1, (a_0+a_2)(b_0+b_2)-a_0b_0+a_1b_1)$ [BBD+07] where $b \in \{-1,1\}$ from (8)	6 muls, 12 adds/subs
A^2	$(a_0^2-ba_1a_2, ba_2^2-a_0a_1-a_1a_2, (a_0+a_1)(a_0+a_1+a_2)-a_0^2+a_0a_1+a_1a_2)$ [BBD+07]	5 muls, 7 adds/subs
A^{-1}	$(w^{-1}(a_0^2-(a_1^2-a_2^2)-a_2(a_0+ba_1)), w^{-1}(ba_2^2-a_0a_1), w^{-1}(a_1^2-a_2^2-a_0a_2))$ where $w=a_0^2(a_0-a_2)+a_1^2(-a_0+ba_1)+a_2^2(-(-a_0+ba_1)+a_2)$ [BBD+07]	12 muls, 11 adds/subs, 1 inv

3.3. Pairing Algorithm

The η_T algorithm is the most efficient algorithm for computing the Tate pairing in characteristic 3. Barreto et al. proposed the reversed loop approach to computing the η_T pairing [BGhS07]:

$$\eta_T(P,Q) = l_{P'}(\phi(Q)) \left(\prod_{j=0}^{\frac{m-1}{2}} g_{\left[3^{\frac{m-1}{2}-j}\right]P'}(\phi(Q))^{3^j} \right) \tag{12}$$

where $P' = [-\gamma]P$, $l_V(x,y)$ is the line equation of the line corresponding to the addition of $\left[3^{\frac{m+1}{2}}\right]V$ and $[\gamma]V$, and $g_V(x,y)$ is the rational function arising during the doubling of V. Equation (12) can be implemented using Algorithm XIII.3 [BBD+07], where

$$\nu = \begin{cases} 1, & m = 5, 11 \bmod 12 \\ -1, & m = 1, 7 \bmod 12 \end{cases} \quad (13) \qquad \lambda = \begin{cases} 1, & m = 7, 11 \bmod 12 \\ -1, & m = 1, 5 \bmod 12 \end{cases}. \quad (14)$$

Algorithm XIII.3 Characteristic 3 reversed loop η_T pairing with cube roots

```
Input: $P, Q \in E(\mathbb{F}_{3^m})[l]$.
Output: $\eta_T(P,Q) \in \mathbb{F}^*_{3^{6m}}$.
1: $x_P = x_P - \nu b$; $y_p = -\gamma y_P$                                    – 1 add
2: $t = x_P + x_Q$                                                              – 1 add
3: $R = (\lambda y_P t + y_Q\sigma - \lambda y_P \rho)(-t^2 - \lambda y_p y_Q \sigma - t\rho - \rho^2)$   – 6 muls, 1 cube, 6 adds
4: for $j$ from 1 to $\frac{m-1}{2}$ do
5:     $x_P = \sqrt[3]{x_P}$; $y_P = \sqrt[3]{y_P}$; $x_Q = x_Q^3$; $y_Q = y_Q^3$                – 2 roots, 2 cubes
6:     $t = x_P + x_Q$; $u = y_P y_Q$                                          – 1 muls, 1 add
7:     $S = -t^2 - \lambda u\sigma - t\rho - \rho^2$                                   – 1 mul
8:     $R = R.S$                                                               – Sparse $\mathbb{F}_{3^{6m}}$ mul
9: return $R^M$.
```

The $\mathbb{F}_{3^{6m}}$ multiplication required on line 8 of Algorithm XIII.3 can be optimized due to the sparseness of the multiplicands. It can be performed in 12 $\mathbb{F}_{3^m}$ multiplications and 59 $\mathbb{F}_{3^m}$ additions [BBD+07]. Cubed root extraction is required on line 5 of Algorithm XIII.3. It is however, possible to remove the need for a hardware cube root extractor by first precomputing the cubes of x_P and y_P and then getting the cube roots by accessing the list in reverse order [BGhS07]. The cube roots can also be avoided by exploiting the bilinearity of the pairing as shown in Equation (15) [BBD$^+$07]. This can be evaluated using Algorithm XIII.4 [BBD$^+$07]:

$$\begin{aligned}\eta_T(P,Q) &= \eta_T(P,\left[3^{-\frac{m-1}{2}}\right]Q)^{3^{\frac{m-1}{2}}} \\ &= l_{P'}(\phi(Q))^{3^{\frac{m-1}{2}}}\left(\prod_{j=0}^{\frac{m-1}{2}} g_{\left[3^{\frac{m-1}{2}-j}\right]P'}(\phi(Q))^{3^{\frac{m-1}{2}+j}}\right) .\end{aligned} \tag{15}$$

Algorithm XIII.4 Characteristic 3 reversed loop η_T pairing without cube roots

```
Input: $P, Q \in E(\mathbb{F}_{3^m})[l]$.
Output: $\eta_T(P,Q) \in \mathbb{F}^*_{3^{6m}}$.
 1: $x_P = x_P + b;\ y_p = -\gamma y_P$                                   – 1 add
 2: $x_P = x_P^3;\ y_P = y_P^3$                                           – 2 cubes
 3: $t = x_P + x_Q$                                                        – 1 add
 4: $R = (\lambda y_P t - \lambda y_Q \sigma - \lambda y_P \rho)(-t^2 + y_p y_Q \sigma - t\rho - \rho^2)$   – 6 muls, 1 cube, 6 adds
 5: for $j$ from 1 to $\frac{m-1}{2}$ do
 6:     $R = R^3;\ x_Q = x_Q^9 - b;\ y_Q = -y_Q^9$                       – $\mathbb{F}_{3^{6m}}$ cubing, 4 cubes, 1 add
 7:     $t = x_P + x_Q;\ u = y_P y_Q$                                      – 1 mul, 1 add
 8:     $S = -t^2 + u\sigma - t\rho - \rho^2$                              – 1 mul
 9:     $R = R.S$                                                          – Sparse $\mathbb{F}_{3^{6m}}$ mul
10: return $R^M$.
```

In characteristic 3 it can provide a speed up if the loop is unrolled and operations on different iterations of the loop are performed in parallel. For example see [BBD$^+$07, GPS06].

The final exponentiation is to the power of $M = (3^{3m}-1)(3^m+1)(3^m+1-\gamma 3^{\frac{m+1}{2}})$. This can be performed using the scheme presented in [STO07]. The first computation is $R^{3^{3m}-1}$ where $R = \eta_T(P,Q) \in \mathbb{F}^*_{3^{6m}}$. Representing $R = R_0 + R_1\sigma$ where R_0 and $R_1 \in \mathbb{F}^*_{3^{3m}}$ then $R^{3^{3m}-1}$ is computed via:

$$R^{3^{3m}-1} = \frac{(R_0^2 - R_1^2) + R_0 R_1 \sigma}{R_0^2 + R_1^2} .$$

This can be implemented using multiplications, squarings and a single inversion over $\mathbb{F}_{3^{3m}}$. One important note is that $R^{3^{3m}-1}$ is an element of the torus $T_2(\mathbb{F}_{3^{3m}}) = \{X_0 + X_1\sigma \in \mathbb{F}^*_{3^{6m}} : X_0^2 + X_1^2 = 1\}$ introduced by Granger et al. in [GPS06]. This is highly advantageous as arithmetic on the torus is cheaper than arithmetic on $\mathbb{F}^*_{3^{6m}}$. In particular, inversion on the torus is a simple conjugation i.e. if $A = A_0 + A_1\sigma \in T_2(\mathbb{F}_{3^{3m}})$ then $A^{-1} = A_0 - A_1\sigma$. Full details of the complete final exponentiation can be found in [STO07]. The total cost of the final exponentiation is 58 multiplications, 1 inversion and 103 or 105 (depends on m mod 6) additions over $\mathbb{F}_{3^m}$ and $(\frac{m+1}{2})$ cubings and 1 multiplication over $\mathbb{F}_{3^{6m}}$.

The characteristic 3 Tate pairing $t(P, Q)$ can be computed using Algorithm XIII.3 at the cost of just a few additional $\mathbb{F}_{3^m}$ operations by noting that $t(P, Q) = \eta_T(\left[-\gamma 3^{\frac{3m-1}{2}}\right] P, Q)$ [BBD+08]. The only additional computation required is $\left[-\gamma 3^{\frac{3m-1}{2}}\right] P = (\sqrt[3]{x_P} - b, -\gamma\epsilon\sqrt[3]{y_P})$ where ϵ is given by

$$\epsilon = \begin{cases} 1, & m \bmod 12 = 7, 11 \\ -1, & m \bmod 12 = 1, 5 \end{cases} .$$

When using Algorithm XIII.4 the Tate pairing can computed using $\eta_T = ([-\gamma]P, [3^{\frac{3m-1}{2}}]Q)$. This actually saves one $\mathbb{F}_{3^m}$ addition and two $\mathbb{F}_{3^m}$ cubings [BBD+08].

4. Tate Pairing over Supersingular Curves in Characteristic p

4.1. Curve

An elliptic curve over $\mathbb{F}_p$ is defined by Equation (16).

$$E_p(\mathbb{F}_p) : y^2 = x^3 + ax + b, \text{ where } 4a^3 + 27b^2 \neq 0, \ a, b \in \mathbb{F}_p \ . \tag{16}$$

The group law on this curve is given by Equations (17) and (18)

$$\begin{aligned} P_2 &= P_0 + P_1 \\ \lambda &= \tfrac{y_1 - y_0}{x_1 - x_0} \\ x_2 &= \lambda^2 - x_0 - x_1 \\ y_2 &= \lambda(x_0 - x_2) - y_0 \end{aligned} \tag{17}$$

$$\begin{aligned} P_2 &= [2]P_0 \\ \lambda &= \tfrac{3x_0^2 + a}{2y_0} \\ x_2 &= \lambda^2 - 2x_0 \\ y_2 &= \lambda(x_0 - x_2) - y_0 \end{aligned} \tag{18}$$

in affine coordinates. Jacobian coordinates [BSS05] can remove the need for expensive field inversion at the cost of additional field multiplications. Point addition and doubling require sixteen and ten multiplications respectively when using Jacobian coordinates. The equation of the line which arises during a point addition or doubling is $l_T(x, y) = y - y_T - \lambda(x - x_T)$.

The curve in Equation (16),

$$E_{pss}(\mathbb{F}_p) : y^2 = x^3 + (1 - b)x + b, \ b \in \{0, 1\} \tag{19}$$

can be either supersingular with embedding degree $k = 2$ or non supersingular with any finite embedding degree. In the supersingular case the curve equation simplifies to that given in (19), which has order $\#E_{pss}(\mathbb{F}_p) = p + 1$. The distortion map ϕ for the supersingular curve defined in Equation (19) is $\phi(x, y) = (-x, iy)$ where $i \in \mathbb{F}_{p^2}$ and $i^2 = -1$.

In the non supersingular case it is still desirable to select a curve with embedding degree $k = 2$ for the reasons outlined in [Sco05a]. Other embedding degrees can be used. For example, a suitable Barreto-Naehrig (BN) curve with $k = 12$ is outlined in Section 2 of Chapter XII.

4.2. Characteristic p Finite Field Arithmetic

Implementing the Tate pairing over $\mathbb{F}_p$ will require modular addition, subtraction, multiplication and inversion. The bit length of the field is denoted p_b. An $\mathbb{F}_p$ addition is carried out using two adders. The first adds the two elements of the field together while the second one subtracts the modulus. The carry out of the second adder determines which result is the correct one. The type of adder used depends on the design goals. Carry look-ahead (CLA) or carry select (CSA) adders can be used to achieve minimum latency. A more area efficient choice would be a carry-propagate adder (CPA). A $p_b + 3$ bit CPA is required to handle the overflow of the intermediate results. Subtraction is carried out in a similar fashion. Both addition and subtraction can be computed in one clock cycle.

Montgomery proposed an efficient method for performing modular multiplication using a series of additions and right shifts in [Mon85]. This method avoids the need for costly trial division of the modulus. The output of the Montgomery multiplication is, however, a factor $2^{-(p_b+2)}$ times smaller than the desired result. In order to correct the result, the output must be Montgomery multiplied by $(2^{2(p_b+2)} \mod p)$. When a large number of multiplications are required it becomes inefficient to correct every result. A better solution is to initially convert the numbers to the Montgomery domain. To do this, the number is Montgomery multiplied by $(2^{2(p_b+2)} \mod p)$. To convert a number back, it is Montgomery multiplied by 1. Therefore, all arithmetic operations in $\mathbb{F}_p$ are carried out in the Montgomery domain. At the end of all the multiplications, the final result can be converted back to the integer domain. Montgomery multiplication requires $p_b + 2$ clock cycles. Squaring in $\mathbb{F}_p$ is the same operation as multiplication. There is no gain in terms of hardware or speed when performing squarings. Koç et al. [KAK96] proposed faster algorithms for Montgomery multiplication in $\mathbb{F}_p$. However, an implementation of these algorithms would require significantly more hardware resources.

The Montgomery inverse was defined in [Kal95] as the integer $x = a^{-1}2^{p_b} \mod p$. The algorithm to compute the Montgomery inverse is based on the Extended Euclidean Algorithm (EEA). It is broken into two phases. The first phase performs the EEA and a modular correction step. The result of this phase is given by $r = a^{-1}2^k \mod p$ where $p_b < k < 2p_b$. The second phase corrects this result to give the Montgomery Inverse. Inversion using such an algorithm requires $2p_b + 3$ clock cycles. A complete review of Montgomery inversion can be found in [CDM05].

Example implementation results for the previously discussed $\mathbb{F}_p$ architectures are given in Table XIII.7. It can be seen that arithmetic in $\mathbb{F}_p$ is considerably more expensive in terms of time than in either $\mathbb{F}_{2^m}$ or $\mathbb{F}_{3^m}$. The reason for this is that addition in $\mathbb{F}_p$ is not carry free. Resources in Xilinx FPGAs are organized into columns. Each column has fast routing built in for the carry bit in integer arithmetic. However, the bit lengths used in cryptography are so large that the carry must be split between two columns which introduces large routing delays. Hence $\mathbb{F}_p$ arithmetic has a low operating frequency on FPGAs which results in low overall performance. Addition architectures such as CLAs [BBBP08] or CSAs [CDM05] can be used to increase the clock frequency but at the expense of additional area requirements. To illustrate this, the implementation results for the three architectures using CSAs are presented in Table XIII.8.

Multiplication and squaring are also required in the extension field $\mathbb{F}_{p^k}$. Here, arithmetic in the popular $k = 2$ case will be outlined. For larger embedding degrees the extension field can be represented using a tower of extensions idea. As an example please refer to Chapter XII for some detail on the $k = 12$ case.

Table XIII.7. $\mathbb{F}_p$ CPA arithmetic architecture examples implemented on a Virtex-II Pro FPGA, $p_b = 512$

Architecture	Slices	LUTs	Freq. (MHz)	Clock Cycles	Time (ns)
Modular Adder/Subtracter (CPA)	1404	1544	29.9	1	33.44
Montgomery Multiplier	1406	2061	29.9	514	17190.64
Montgomery Inverter	3015	5715	29.2	1027	35171.23

Table XIII.8. $\mathbb{F}_p$ CSA arithmetic architecture examples implemented on a Virtex-II Pro FPGA, $p_b = 512$.

Architecture	Slices	LUTs	Freq. (MHz)	Clock Cycles	Time (ns)
Modular Adder/Subtracter (CSA)	1943	2640	40.0	1	25.00
Montgomery Multiplier	1848	2748	40.8	514	12598.04
Montgomery Inverter	4502	7248	40.8	1024	25098.04

Elements of $\mathbb{F}_{p^2}$, or the *quadratic extension field* as it is known, can be represented by polynomials of the form $a_0 + a_1 i$ where $a_0, a_1 \in \mathbb{F}_p$ and i is a Quadratic Non Residue (QNR) with respect to p [Sco05c]. The condition that $p \mod 4 = 3$ ensures that -1 is a quadratic non residue with respect to p. In this case an element of $\mathbb{F}_{p^2}$ can be considered as a complex number with $i = \sqrt{-1}$. Using this representation $\mathbb{F}_{p^2}$ multiplication can be performed in four $\mathbb{F}_p$ multiplications and three additions/subtractions. The Karatsuba method can be used to perform $\mathbb{F}_{p^2}$ multiplication in three $\mathbb{F}_p$ multiplications but at the cost of six additions/subtractions. The cost of arithmetic in $\mathbb{F}_{p^2}$ is summarized in Table XIII.9.

Table XIII.9. Arithmetic operations over $\mathbb{F}_{p^2}$.

Op	Method	Cost on $\mathbb{F}_p$
$A + B$	$(a_0 + b_0) + (a_1 + b_1)i$	2 adds
A^2	$(a_0 + a_1 i)^2 = (a_0 + a_1)(a_0 - a_1) + 2a_0 a_1 i$	2 muls, 3 adds
$A * B$	$A * B = (a_0 b_o - a_1 b_1) + ((a_0 + a_1)(b_0 + b_1) - a_0 b_0 - a_1 b_1)i$ Karatsuba method used to trade $\mathbb{F}_p$ multiplications for additions.	3 muls, 6 adds

4.3. Pairing Algorithm

The BKLS/GHS algorithm for computing the Tate pairing on $E_{pss}(\mathbb{F}_p)$ is presented in Algorithm XIII.5. The algorithm requires elliptic curve point addition and doubling on $E_{pss}(\mathbb{F}_p)$. Squaring, multiplication and exponentiation in F_{p^k} are also necessary.

As already mentioned, the most popular choice for implementing the Tate pairing in $\mathbb{F}_p$ is to use $k = 2$ [Sco05a,BBBP08]. Therefore, to implement the BKLS/GHS algorithm it will be necessary to perform arithmetic in $\mathbb{F}_p$ and $\mathbb{F}_{p^2}$. In general the final exponent is $\frac{p^k - 1}{n}$ which can be efficiently computed using the Lucas laddering technique [SB04]. In Algorithm 3 in Chapter XII the final exponentiation is described for the case where $k = 12$.

5. η_T Implementation Example

A processor implementing the characteristic 3 η_T pairing is illustrated in Figure XIII.1. The architecture contains a controller, memory and one arithmetic unit for each of $\mathbb{F}_{3^m}$

Algorithm XIII.5 BKLS/GHS algorithm for computing the n^{th} order Tate pairing

```
Input: $P, R \in E_{pss}(\mathbb{F}_p)[l]$, $n = \sum_{i=0}^{L} n_i 2^i$.
Output: $f = t_l(P, Q) \in \mathbb{F}^*_{p^k}$.
1: $f \leftarrow 1$; $T \leftarrow P$; $n = n - 1$; $Q = \phi(R)$
2: for $i$ from $L - 1$ downto 0 do
3:     $T \leftarrow [2]T$                          – EC Point Double
4:     $f \leftarrow f^2.l_T(Q)$                    – $l(Q)$, $\mathbb{F}_{p^k}$ square, $\mathbb{F}_{p^k}$ mul
5:     if $n_i = 1$ then
6:         $T \leftarrow T + P$                     – EC Point Addition
7:         $f \leftarrow f.l_T(Q)$                  – $l(Q)$, $\mathbb{F}_{p^k}$ mul
8: return $f^{\frac{p^k-1}{n}}$.                    – $\mathbb{F}_{p^k}$ exponentiation
```

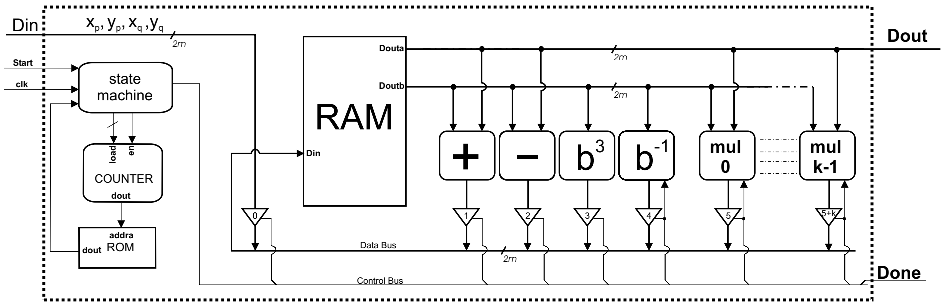


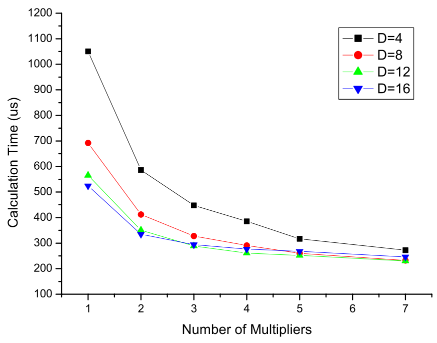
Figure XIII.1. The elliptic curve $\eta_T{}^M$ processor in the characteristic 3 case.

addition, subtraction, cubing and inversion plus a variable number of $\mathbb{F}_{3^m}$ multipliers. Operations for pairing computation are then scheduled efficiently through these arithmetic units. The components are connected via a $2m$ bit data bus.

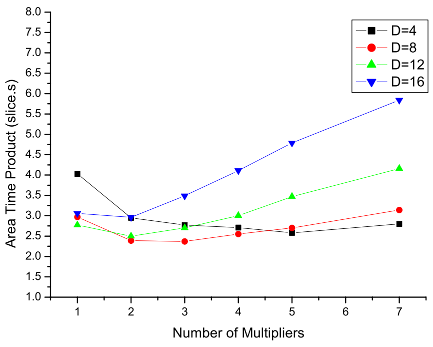
Multiplication is performed often, and is a relatively time consuming operation. The processor is designed such that the number of multipliers in the processor can easily be varied with little impact on the overall design (apart from area). By varying the number of multipliers, and indeed their digit sizes, the processor can be tailored for a high throughput implementation, a low area implementation or a balance between the two.

Dual port RAM is used to store the intermediate variables required for computation. The required input coordinates are read serially into the RAM before computation begins. During computation, two data lines bring variables to the field arithmetic components to be operated on. Tri-state buffers at the output of the arithmetic components are used to select which result is written to RAM when an $\mathbb{F}_{3^m}$ operation has been completed.

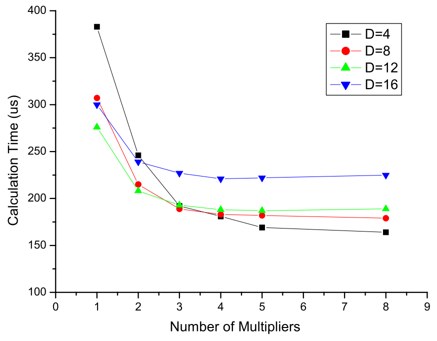
The control ROM of Figure XIII.1 contains a set of instructions for each stage. The Finite State Machine (FSM) Controller cycles through these ROM instructions (via the counter) in order to evaluate each stage. The counter has several load pins. These are used to jump to the start of the different stages within the ROM instruction set. It is vital that the scheduling of operations be as efficient as possible to ensure that clock cycles are not wasted. If possible, additions, subtractions, squarings and cubings are performed and their results written to RAM while multiplication or inversion is in progress. In particular, the scheduling of operations through the iterative loop of the pairings is vital since the loops are performed a number of times.



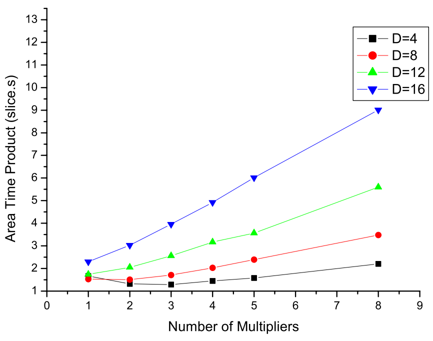
(a) Time for computation of $\eta_T(P,Q)^M$ in μs, $\mathbb{F}_{2^{313}}$



(b) Area Time Product for $\mathbb{F}_{2^{313}}$



(c) Time for computation of $\eta_T(P,Q)^M$ in μs, $\mathbb{F}_{3^{97}}$



(d) Area Time Product for $\mathbb{F}_{3^{97}}$

Figure XIII.2. $\mathbb{F}_{2^{313}}$ and $\mathbb{F}_{3^{97}}$ ${\eta_T}^M$ micro architecture results for various multiplier configurations.

The architecture of Figure XIII.1 can also be used to perform the η_T pairing in characteristic two. The $\mathbb{F}_{3^m}$ arithmetic units are simply replaced by $\mathbb{F}_{2^m}$ units. The data bus width is also be reduced from $2m$ to m bits. Some minor changes to the ROM instruction set are also required. In [RhM+07a] a C program was written to generate the VHDL implementation of the processor. The required field, field extension m and the number of underlying arithmetic units can be specified as inputs to the program. In this way the architecture can be quickly implemented using different parameters. More details on this processor, as well as the exact algorithms used, can be found in [RhM+07a].

5.1. Results

The characteristic 2 and 3 pairing processors described in this section were implemented on a Xilinx Virtex II Pro FPGA (xc2vp100-6ff1696). In characteristic 2, results are returned for the field size $m = 313$. Note that the processor is fully reconfigurable for any suitable field size. Results returned by the processor when $\eta_T(P,Q)^M$ is computed with 1, 2, 3, 4, 5 and 7 multipliers with digit sizes of 4, 8, 12 and 16 are illustrated in Figures XIII.2(a) and XIII.2(b). Note that an implementation with six multipliers would not yield an efficient use of resources. This is because the multiplications required within the loop dominate the pairing calculation time. The latency of these fourteen multiplications does not improve between five and seven multipliers.

In characteristic 2, the processor can be configured to return its fastest pairing in 230 μs for $m = 313$. Figure XIII.2(a) illustrates trends in the time taken to compute $\eta_T(P,Q)^M$ for the various multiplier configurations and provides some interesting results. It can be seen that some of the $D = 16$ configurations are actually slower than configurations with smaller multiplier digit sizes due to the reduction in clock frequency. Figure XIII.2(b) illustrates variations in the area-time product (in $slices.seconds$) of the designs on this field and provides an indication of the efficiency of each of the configurations (a lower product signifies a more efficient design). For example, the fastest pairing time is returned in the {D=12, number of mults=7} case. In this case, 18111 FPGA slices return a pairing in 230 μs, yielding an area-time product of 4.2 $slice.s$. For a digit size of 12, a more efficient use of resources would be returned by a configuration with only three multipliers. The pairing is now computed in 289 us, but uses only 9329 slices (49% fewer), yielding an area-time product of 2.69 $slice.s$. For a digit size of 4, the most efficient utilization of resources is reached at five multipliers. In the $D = 16$ case, the most efficient option is two multipliers. Overall, the most efficient design is the {D=8, number of mults=3} case, which returns a pairing in 328μs, costs 7232 slices and returns an area-time product of 2.37 $slice.s$.

In characteristic 3, the processor is generated for the field size $m = 97$. Results returned by the cryptographic processor for 1, 2, 3, 4, 5 and 8 multipliers of digit sizes 4, 8, 12 and 16 are illustrated in Figures XIII.2(c) and XIII.2(d). The fastest pairing is returned in 179 μs for $m = 97$. The computation times are illustrated in Figure XIII.2(c). It can again be seen that larger digit sizes are not warranted due to the decrease in clock frequency and resultant decrease in computation time. The area-time products of the configurations are plotted in Figure XIII.2(d). It can be seen that in the $D = 12$ and $D = 16$ cases, the most efficient utilization of resources is provided by configurations with just one multiplier. Overall, the most efficient configuration is the {D=4, number of mults=3} case, which returns a pairing in 192 μs and consumes 6690 slices, returning an area-time product of 1.28 $slice.s$.

In both the characteristic 2 and 3 cases, these results compare favorably with software implementations. In Barreto et al. [BGhS07] it is reported that a characteristic 2 η_T pairing takes 3.5 ms when implemented on a Pentium IV processor running at 3 GHz over a field size of $m = 307$. A characteristic 3 pairing on the $m = 97$ field requires 2.72 ms.

6. Pairing Architecture Examples

In broad terms, there are three distinct architectural options when creating a dedicated pairing processor. The first is a *macro* type implementation, of the type discussed in detail in [RhM$^+$06,KRMM06,KMPB05,RhM$^+$07b,SKG06,BSTO07a,KRMM07]. In this case high throughput is achieved by taking advantage of large amounts of parallelism. For example, the arithmetic units required for extension field operations could be hardwired. However, this type of architecture requires many underlying arithmetic components, resulting in a high area cost.

The second type of hardware configuration is a *micro* implementation, of the form discussed in the previous section and in [RhM+07a,KRMM07,GP05]. In this case, the extension field architectures are not hardwired, instead the processor contains a small number of sub field units, with operations to compute the pairing scheduled efficiently. These micro implementations are reconfigurable: the number of arithmetic units can be varied to provide a high throughput implementation, a low area implementation or an appropriate compromise between the two.

The third general type is a *hybrid* implementation [BBDO07]. These are based on hybrid arithmetic logic units (ALUs). Such ALUs can perform the basic underlying field arithmetic such as addition and multiplication. Through resource sharing they provide several field operations at very low cost in terms of circuit area.

These are just three of the general hardware architectures employed to accelerate pairing computation and are by no means the only possibilities as there are a wealth of different options which face a hardware designer. These range from the architectures to use for the basic field operations to the top level pairing architecture to the implementation platform. These choices will be influenced by the design constraints. For example: maximum speed, minimum area or minimum power and energy consumption. Table XIII.10 gives an example of some of the architectures presented in the literature. These results merely represent a small example of the massive design space available to hardware engineers. It is meant to give an idea of the performance available from hardware accelerators.

The results of Ronan et al. [RhM+06] and Keller et al. [KRMM07] show that increasing the digit size of the underlying multipliers does not always yield a faster computation time due to increased critical path and routing delays which reduce the overall clock frequency above a certain digit size.

Grabher and Page [GP05] compare the DL and the η algorithm on the same hardware. On their particular architecture the DL algorithm is marginally faster as it uses dedicated hardware cube root circuits. They also compare the effect of using different field basis representations. Over $\mathbb{F}_{3^{89}}$ elements of the field are represented in normal basis. While polynomial basis is used over $\mathbb{F}_{3^{97}}$. Polynomial basis was found to be a faster as the critical multiplier circuit was less complex resulting in higher attainable clock frequencies.

In this chapter and [BBDO07], a C program is used to automatically generate the architectures based on several different design parameters such as field size, irreducible polynomial, number and type of underlying field arithmetic units etc. Such an approach yields flexible architectures that can easily be tailored to a specific application.

The fastest architectures reported in the literature report computation time of 43 μs over $\mathbb{F}_{2^{239}}$ [SKG06] and 20.9 μs over $\mathbb{F}_{3^{97}}$ [Jia07]. Both of these architectures are highly suited to application where speed is the primary design constraint. Their large area footprints, 31719 and 74105 slices respectively, make them less suited to application in area constrained devices. They have area-time (AT) products of 1.36 and 1.55 $slices.second$ respectively. The AT product is a metric used to measure the area speed trade-off. The architecture of Beuchat et al. [BBD+07] for computing the pairing over $\mathbb{F}_{3^{97}}$ exhibits an AT product of just 0.25 $slice.s$ when implemented on a Virtex 4 device. While this architecture is slower than that of [Jia07], in terms of area/speed it is a better option in an area constrained environment. A hardware designer should therefore take into account

Table XIII.10. Example results from the literature.

Ref.	Field, Algorithm	Curve, k	Digit Size	Platform	Area (slices)	Freq. (MHz)	Time (μs)
	$\mathbb{F}_{2^{239}}$,η		$D = 16$		18202	100	55
			$D = 32$		31719	83	43
[SKG06]	$\mathbb{F}_{2^{283}}$,η	E_2,	$D = 16$	Virtex II Pro	22726	84	87
		4	$D = 32$		37803	72	61
	$\mathbb{F}_{2^{239}}$,η_T		$D = 16$		25487	84	41
	$\mathbb{F}_{2^{283}}$,η_T		$D = 32$		33525	56	78
	$\mathbb{F}_{2^{251}}$,		$D = 1$		20312	51	3030
	BKLS		$D = 6$		26132	43	1070
			$D = 8$		29380	35	1160
	$\mathbb{F}_{2^{283}}$,	E_2,	$D = 4$		27411	35	1950
[KRMM07]	BKLS	4	$D = 6$	Virtex II	29421	33	1670
	$\mathbb{F}_{2^{251}}$,		$D = 6$, 1 M		3788	40	4900
	BKLS		$D = 6$, 9 M		13387	40	2600
	$\mathbb{F}_{2^{283}}$,		$D = 6$, 1 M		4273	40	6000
	BKLS		$D = 6$, 9 M		15065	40	3000
			$D = 4$		21021	51	206
[RhM+07b]	$\mathbb{F}_{2^{103}}$, η_T	hyper,	$D = 8$	Virtex II Pro	24290	43	135
		12	$D = 12$		27182	51	206
			$D = 16$		30464	41	132
		E_2,	$D = 4$		34675	55	203
[RhM+06]	$\mathbb{F}_{2^{313}}$,η_T	4	$D = 8$	Virtex II Pro	41078	50	124
			$D = 12$		44060	33	146
This	$\mathbb{F}_{2^{313}}$, η_T	E_2,	$D = 8$, 3 M	Virtex II Pro	7232	95	328
chapter		4	$D = 12$, 7 M		18111	89	230
				Virtex II Pro	1833	145	192
[BBD+07]	$\mathbb{F}_{3^{97}}$, η_T	E_3,	$D = 3$	Virtex 4	1851	203	137
		6		Spartan 3	1857	100	278
				Cyclone-II	3216 LE	152	183
[BSTO07a]	$\mathbb{F}_{3^{97}}$, η_T	E_3,	$D = 3$	Cyclone-II	14895 LE	149	33
		6	$D = 4$		18553 LE	147	27
[Jia07]	$\mathbb{F}_{3^{97}}$, η_T	E_3, 6	n/a	Virtex 4	74105	77.76	20.9
[KMPB05]	$\mathbb{F}_{3^{97}}$,	E_3,	$D = 4$	Virtex II	46590	18.3	6700
	BKLS	6	$D = 8, 4$	Virtex II Pro	50286	19.0	5920
	$\mathbb{F}_{3^{89}}$, DL		$D = 2$		4233	85	1141
[GP05]	η	E_3,		Virtex II Pro		85	1196
	$\mathbb{F}_{3^{97}}$, DL	6	$D = 4$		4481	150	432
	η					150	464
	$\mathbb{F}_{3^{97}}$, η_T		$D = 3$		1888	147	222
[BBDO07]	$\mathbb{F}_{3^{97}}$, η_T	E_3,	$D = 7$	Virtex II Pro	2189	117	146
	$\mathbb{F}_{3^{193}}$, η_T	6	$D = 3$		2811	126	877
	$\mathbb{F}_{3^{193}}$, η_T		$D = 7$		4450	108	495
This	$\mathbb{F}_{3^{97}}$,η_T	E_3,	$D = 4$, 3 M	Virtex II Pro	6690	98	192
chapter		6	$D = 4$, 8 M		13369	98	164
[BBBP08]	$\mathbb{F}_{p-512}$, BKLS	E_p, 2		Virtex II	33857	135	1610

their particular design constraints when evaluating what is the best method to use for implementing a pairing based accelerator.

7. Summary

In this chapter the most efficient algorithms for computing the Tate pairing on supersingular elliptic curves in fields of characteristic 2, 3 and p have been presented. Architectures to implement the required arithmetic in the underlying finite fields have been discussed and example implementation results on a Xilinx FPGA presented. This gives an idea of the complexity of the underlying arithmetic in each of the three fields. The cost of the necessary arithmetic in the degree k extension fields in terms of subfield operations was also presented.

An example architecture to compute the η_T pairing in fields of characteristic 2 or 3 was presented. The implementation results of this architecture on a Xilinx FPGA were used to highlight some of the trade offs facing the designer of a Tate pairing accelerator. The main trade off is usually area/speed. The efficiency of the design was evaluated in terms of its area-time product.

Finally, different hardware implementation results from the literature were presented and discussed. The fastest Tate pairing evaluation time in the literature to date is 20.9 μs over $\mathbb{F}_{3^{97}}$. However the most efficient architecture in terms of both area and speed reported a calculation time of 137 μs while occupying only 1851 Virtex 4 FPGA slices.

In general it was shown that hardware implementations can significantly outperform software implementations by taking advantage of available parallelism within the algorithms. Fields of characteristic 2 and 3 have to date outperformed characteristic p in hardware implementations due to the efficiency of the underlying arithmetic.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-58603-947-9-226

Chapter XIV

Implementation Attacks & Countermeasures

Claire WHELAN [a], Dan PAGE [b], Frederik VERCAUTEREN [c], Michael SCOTT [d] and William MARNANE [a]

[a] *University College Cork, Ireland*
[b] *University of Bristol, UK*
[c] *Katholieke Universiteit Leuven, Belgium*
[d] *Dublin City University, Ireland*

Abstract. In this chapter the effect of implementation attacks on the security of pairings is examined. This is a particularly pertinent issue given the increasing efficiency of pairing algorithms, which have recently proved viable for implementation on resource constrained devices where such attacks pose a serious threat. Specifically, we discuss power analysis and fault attacks on a range of pairing algorithms. Countermeasures and their impact on performance are also described.

Keywords. Bilinear maps, pairings, side channel analysis, differential power analysis, correlation power analysis, fault analysis

As the previous two chapters have discussed, the efficient implementation of pairings is vital if Identity Based Encryption (IBE) [BF01,BF03] and other pairing-based cryptosystems are to be adopted in a practical environment. Research efforts have produced fast pairings recently shown to be efficiently computable on a resource constrained smart card [SCA06]. This makes pairing-based cryptosystems eligible for increasingly diverse and practical applications where resource constrained devices are essential. For IBE in particular, this development is beneficial since smart cards are often used as identity-aware tokens. It also however introduces a range of problems related to physical security, such as the need for countermeasures against implementation attacks.

The security of cryptosystems such as IBE is typically analyzed within a mathematical setting; this is evidenced by work on, for example, pairing inversion [GHV07] and related hard problems. However, instead of targeting the mathematical underpinnings of a cryptographic algorithm, implementation attacks target the implementation itself, and are often able to recover secret information from a physical device by simply monitoring its execution features [GMO01,Koc96,KJJ99] or tampering with the device during execution [BECN$^+$06,BDL97]. This is particularly problematic for smart cards, which are carried into and used in an adversarial environment. In this context, a potential attacker can often secure levels of access to the device, which would be unrealistic in a desktop computing setting. This includes supplying the device with power and clock signals in order to glean information from the results.

Clearly the threat of attack through physical means presents a significant hurdle to the deployment of secure, ubiquitous computing devices. While attack and defence techniques of this sort are well understood for Elliptic Curve Cryptography (ECC) [BSS05], until recently the same understanding was not evident in the context of pairing-based cryptography. This chapter provides an overview of the work performed in this field to date [KTH$^+$06,PV06,WS07,WS06], which has demonstrated that pairings, if implemented as proposed in the literature, can succumb to both power analysis and fault attacks. In addition, it has highlighted that there is an obvious trade-off between efficiency and security since the proposed countermeasures that are viewed as most secure in this setting often introduce the highest performance overhead. Therefore, investigating potential attacks and developing efficient and effective defensive mechanisms is an ongoing research challenge.

1. Overview of Power Analysis Attacks

Power analysis is a type of side-channel attack, which exploits the fact that the instantaneous power consumption of a cryptographic device, such as a smart card, depends on the data that it processes and on the operations that it performs [MOP07]. Power analysis attacks typically consist of two stages. The first stage of a power analysis attack consists of a collection or data acquisition phase, which provides the attacker with profiles of execution. Here a profile is the instantaneous power consumption of the device while it executes. The power consumption of the device for a particular execution, will be referred to as a power trace. This phase requires data acquisition equipment such as a digital oscilloscope, suitable probes, and a target device. A typical experimental apparatus to perform a power analysis attack is depicted in Figure XIV.1.

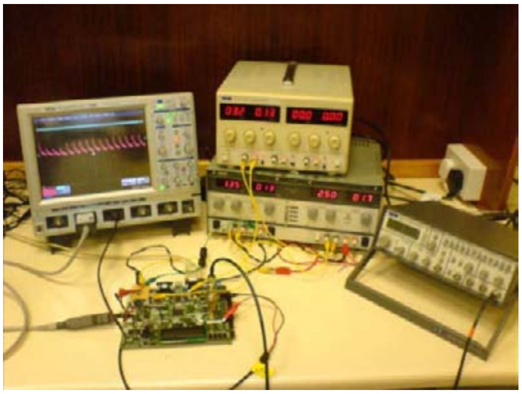


Figure XIV.1. Typical side-channel/power analysis attack laboratory.

The second stage consists of an analysis phase, where the profiles or power traces acquired from the first stage are analyzed to reveal secret information. The analysis procedure depends on the type of attack and associated acquired number of profiles. Power analysis attacks are typically split into two classes, simple and differential. Simple Power Analysis (SPA) is where the attacker can only collect one profile and is required to recover secret information by focusing mainly on the operation being executed. In contrast, Differential Power Analysis (DPA) uses statistical methods to form a correlation between a number of profiles and secret information by focusing mainly on the data items being processed. In particular, it involves identifying an intermediate operation in the computation involving data that is known or computable by the adversary and data that is related to the secret. This operation, referred to in the literature as a selection function, enables a type of power analysis attack known as first-order power analysis. DPA and Correlation Power Analysis (CPA) are types of first-order power analysis. We refer the reader to [MOP07] for further information on power analysis attacks.

1.1. Differential and Correlation Power Analysis

Let $S(\cdot,\cdot)$ denote a selection function. The function S accepts input α_i and K, where α_i is known by the adversary, and K is related to the secret. The target algorithm/device will be executed a number of times with input α_i, where $1 \leq i \leq N$. On accepting input α_i and K, the selection function will produce an unknown intermediate output value denoted by β_i, i.e. $\beta_i = S(\alpha_i, K)$. For each execution a power trace t_i is captured. Generally the selection function accepts n-bit input, where n is either the target device processor word size or in the case of symmetric algorithms, the number of bits input to the S-box. Hence, since K only relates to an n-bit portion of the secret, calculating $\beta_{i,j} = S(\alpha_i, K_j)$ for $0 \leq K_j < 2^n$ hypotheses of n bits of K, is feasible. The process of determining which K_j relates to the secret depends on the attack.

DPA [KJJ99] categorizes the power traces into two sets depending on a single bit b of $\beta_{i,j}$, where the same bit is used to categorize the power traces for all hypotheses of K_j. The first set S_0 will contain all power traces where b is equal to zero, and the second set S_1 will contain all remaining power traces. A differential trace Δ_{K_j} for each K_j is calculated by finding the average of each set and then subtracting the resulting values from each other,

$$\Delta_{K_j} = \frac{\sum_{t_i \in S_0} t_i}{|S_0|} - \frac{\sum_{t_i \in S_1} t_i}{|S_1|} .$$

Each differential trace is plotted. The differential trace relating to K_j which produces the highest peak is identified as the correct value for the secret K (which will be n bits of the overall secret).

CPA [BCO04] is a more accurate variant of DPA that considers the processor's entire word. CPA estimates the power consumption of $\beta_{i,j}$, which is calculated depending on a power model. The power model is chosen to most accurately describe the target device's underlying architecture. The estimated power consumption is then compared to the actual power consumption using a correlation test such as Pearson's correlation coefficient [MR02],

$$\rho_{T,Y} = \frac{E(TY) - E(T)E(Y)}{\sqrt{E(T^2) - E^2(T)}\sqrt{E(Y^2) - E^2(Y)}},$$

where T is a matrix containing the power consumption at a given point in time in all of the power traces t_i, Y is the estimated power consumption of $\beta_{i,j} = S(\alpha_i, K_j)$ for a given K_j and $1 \leq i \leq N$. The value of K_j producing the highest correlation coefficient is identified as the correct n bits of the secret K in question.

2. Overview of Fault Attacks

Performing a fault attack involves disrupting the normal execution of the device in some way so that faulty output is produced [BDL97]. Various mechanisms for fault creation and propagation have been researched and developed. One of the most obvious approaches is to alter the normal working environment of the device. During chip manufacture, a number of predefined thresholds for characteristics such as supply voltage, clock or temperature are set. By forcing the device to work outside of these boundaries, abnormal behavior can be provoked. Another approach to induce a fault is to expose the chip to extreme conditions. For example, by exposing a chip to intense light or a laser beam, since all electric circuits are sensitive to light due to photoelectric effects, the current induced by photons can trigger a fault [SA02].

Once a fault has been injected, two types of faults may be produced, namely provisional (transient) and destructive (permanent) faults. Provisional faults have reversible effects where the circuit will recover its original behavior after the system is reset or when the fault's stimulus ceases. Hence, when using fault injection as an attack strategy, provisional faults are the method of choice since faults under numerous experimental conditions can be attempted until the desired effect is achieved.

Generally a provisional fault attack can have a number of effects: memory can be modified, data can be misread, operations such as read and write operations can be broken. In an algorithm these effects are witnessed in the form of data being corrupted, loops running over or ending prematurely, or instructions being omitted or misinterpreted. We refer the reader to [BECN$^+$06] for a survey on fault attacks.

3. Power Analysis of Pairings

To date, three research papers have examined the effect of power analysis attacks on pairings. The first, by Page and Vercauteren [PV04] in 2004, investigated how SPA and a classic DPA style attack could be applied to the multiplication operation in the Duursma-Lee algorithm for computing the Tate pairing on supersingular elliptic curves in characteristic three. The second, by Whelan and Scott [WS06] in 2006, examined how CPA could be applied to the BKLS algorithm for the Tate pairing, the η_T pairing and the Ate pairing. The effect of CPA on each of these pairings was also determined by examining how finite field operations, such as multiplication, square root and reduction, central to the pairing computation could be attacked using power analysis. The third, by Kim et al. [KTH$^+$06], applied DPA to the multiplication operation over the binary field, which was examined in the context of the η_T pairing. A countermeasure based on the use of randomized projective coordinates in the pairing instead of affine coordinates was also suggested.

Each of the above papers are theoretical. Empirical knowledge, which has identified the characteristics that power analysis attacks exploit in an algorithm, was used to determine whether the examined pairings were potentially susceptible to power analysis. This section describes how power analysis attacks can in theory be applied to pairings. Practical results of applying CPA to the η_T pairing, which validates the proposed attacks in the literature, are also given.

3.1. Power Analysis of Pairings in Theory

SPA can be used when secret information influences the execution path and the sequence of operations being executed. In the context of pairings, the execution path is determined by the order of the Miller loop. This value is public and so identifying the execution path via SPA will provide no insight into the secret. SPA can also be used to attack the underlying finite field operations at the heart of the pairing computation. However, this is dependent on the target finite field operation being naively implemented. For example, if the field multiplication is implemented using the shift-and-add method, one can mount an attack by monitoring execution in the same way as one would monitor binary exponentiation [PV04]. For this reason, first-order power analysis attacks such as DPA and CPA are of greater concern.

As described above, first-order power analysis attacks exploit operations of the form $\beta = S(\alpha, K)$ in a cryptographic algorithm. In the context of bilinear pairings, and in particular in relation to IBE schemes such as $\mathcal{BF}$-$\mathcal{IBE}$ where the pairing in the decryption operation $\hat{e}(usk, C_1)$ accepts the users secret key usk as input, any operation in the calculation of the pairing where the coordinates of the ciphertext C_1 (which can be eavesdropped by an adversary) interact directly with the coordinates of the point usk, is a potential selection function and hence will be the target of a first-order power analysis attack. For the remainder of this chapter we will refer to the pairing in the decryption operation in an IBE scheme as $e(P, Q)$, where as above P relates to the users secret key and Q relates to the ciphertext.

During the point scalar multiplication of the point rP, which upon completion should result in $\mathcal{O}$ (the point at infinity), a distance relationship between the lines produced from the point addition and a static point Q is calculated. If $u_{A,B}$ and v_{A+B} are the diagonal and vertical lines which arise in the addition rule for adding $A = aP$ to $B = bP$ to produce $C = A + B$, the point A has coordinates (x_A, y_A), the point C has coordinates (x_C, y_C), the point Q has coordinates (x_Q, y_Q), and the line through A and B has a slope of λ, then

$$\begin{aligned} u_{A,B}(Q) &= (y_Q - y_A) - \lambda(x_Q - x_A)\,, \\ v_{A+B}(Q) &= (x_Q - x_C)\ . \end{aligned}$$

In each round of the Miller loop $u_{A,B}$ is divided by v_{A+B} to produce an element in the extension field $f \in \mathbb{F}_{q^k}$ referred to as the Miller variable. This value is multiplicatively accumulated and eventually outputted from Miller's algorithm. As can be seen here, known information interacts directly with secret information on a number of occasions. Hence, while the line function may vary from pairing to pairing, the line function in its native form is vulnerable to a power analysis attack.

In each of the pairings, the interaction between the known and secret data will be via finite field operations. DPA and CPA attacks deterministically predict the output of the target finite field operation given only partial input. It is possible to do this for certain finite field operations. For example, there are many methods to perform multiplication, which can vary depending on the finite field. However, regardless of the multiplication algorithm, the least significant word of the result of the multiplication, is a function of the least significant word of both the multiplier and multiplicand. When the two values to be multiplied consist of data known to an adversary and data related to the secret, this forms a selection function, suitable for the application of power analysis attacks like DPA and CPA.

3.2. Power Analysis of Pairings in Practice

The research to date demonstrates in theory how each of the assessed pairings succumb to first-order power analysis attacks. To validate the practicality of such potential weaknesses, access to a pairing implementation on a smart card-like embedded device is required. Practical results of power analysis being applied to a pairing algorithm will be presented here. These results describe a CPA attack on the η_T pairing. The pairing was implemented on a 32-bit microprocessor, which did not have any side-channel countermeasures or protections in place.

3.2.1. Case study: The η_T pairing

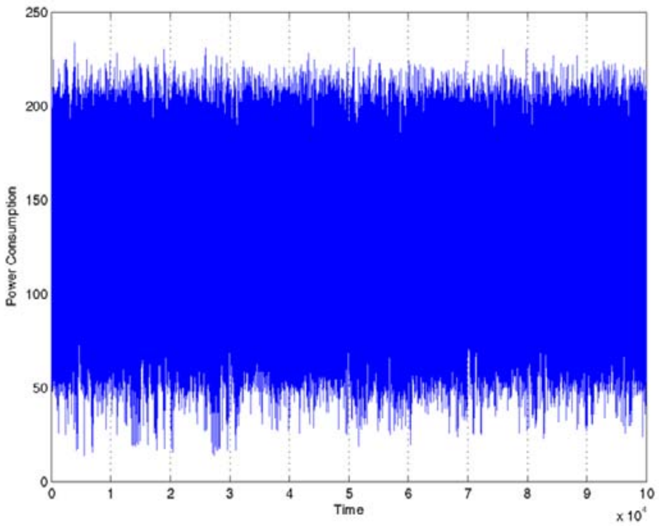
The η_T pairing, described in Chapter XII, is a closed form algorithm with a truncated Miller loop. The BGOhES algorithm [BGhS07], presented in Algorithm XII.1 in Chapter XII, is used to calculate the η_T pairing. Here, we consider the implementation over $E(\mathbb{F}_{2^{4m}})$ with $m = 271$ from the MIRACL library [Sco]. This implementation is similar to that described in [SCA06], which provided the first timings for pairings implemented on a smart card platform. The execution time of the η_T pairing on the target device was 2.078s running at a clock frequency of 7.3728MHz.[1]

Capturing the power consumption of $\eta_T(P, Q)$ involves following the data acquisition phase as described in Section 1. The power trace of the η_T execution is not as well formed as that of a DES or AES power trace [KJJ99], where each round of the cryptographic algorithm can be identified as 16 or 10 distinct patterns respectively. In fact, no identifiable patterns can be seen from the η_T execution at first glance. Even when specific parts of the power trace are examined, discernable features of the algorithm are difficult to identify. For example, Figure XIV.2 illustrates the first round of Miller's algorithm in the η_T pairing. Only when examined at a higher resolution are patterns in the algorithm identified. Figure XIV.3 depicts two distinct patterns in each of the power traces. The power trace on the left represents the square root of x_P and y_P, which is computed at the beginning of each Miller loop round. The power trace on the right represents the squaring of x_Q and y_Q, which is computed at the end of each Miller loop round. (Notice that after the execution of y_Q^2, the next round of Miller's algorithm begins with the operation of $\sqrt{x_P}$, which is identifiable through the power trace pattern.)

There are many operations in the η_T pairing involving the interaction between known and secret data, hence numerous possibilities for selection functions exist. For instance, the line function

$$f = f \cdot (x_P \cdot (\sqrt{x_P} + x_Q) + \sqrt{y_P} + y_Q + \sqrt{x_P} + 1 + (x_P + x_Q)u + v)$$

[1]The chip can run up to speeds of 25MHz, hence it is expected that faster computation times can be achieved.



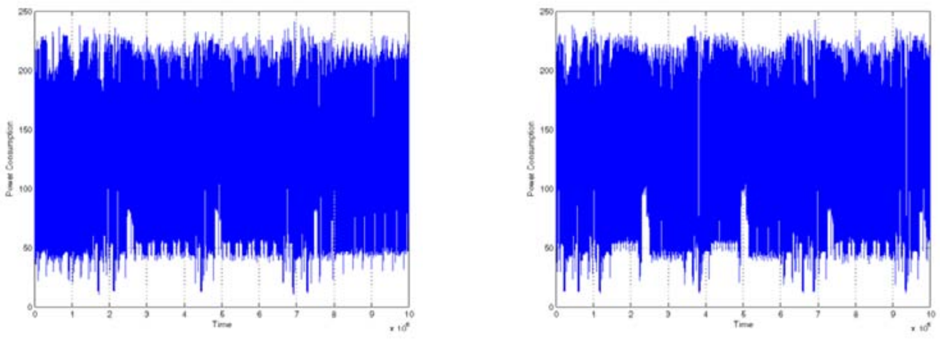


Figure XIV.3. Power consumption of the square root of x_P and y_P (left) and square of x_Q and y_Q (right) in round 1 of Miller's algorithm in the η_T pairing.

presents an adversary with many options. We will assume that $P = (x_P, y_P)$ relates to the users secret key and $Q = (x_Q, y_Q)$ to known data, as is the case in IBE schemes. One target selection function is the operation $\sqrt{x_P} + x_Q$, which is executed in the first round of Miller's algorithm. To identify this operation in the power trace, the $\sqrt{x_P}$, $\sqrt{y_P}$, x_Q^2 and y_Q^2 operation patterns can be used.

The hypothetical output and estimated power consumption of a portion of the output of $\sqrt{x_P} + x_Q$ is calculated given hypotheses for the value of $\sqrt{x_P}$ and N known values of x_Q. Note that in the attack $N = 1000$, which also refers to the number of power traces acquired. For a CPA attack, the size of the portion of $\sqrt{x_P}$ that is predicted depends on the underlying processor word size. The target device has a 32-bit processor, however calculating 2^{32} possible values of $\sqrt{x_P}$ is infeasible and unnecessary as demonstrated by Brier et al. [BCO04]. In [BCO04], partial CPA was proposed which calculates feasible portions of the word at a time. The results for the correlation for 8 bits (16 and 32 bits give similar plots) of the least significant word of $\sqrt{x_P}$ are plotted in Figure XIV.4. The correct guess for the respective portion of $\sqrt{x_P}$ produces the highest correlation, returning a correlation of 0.4043 for 8 bits, 0.561 for 16 bits and 0.8068 for 32 bits of $\sqrt{x_P}$, thus validating the attacks of [KTH$^+$06,PV04,WS06]. To proceed with extracting

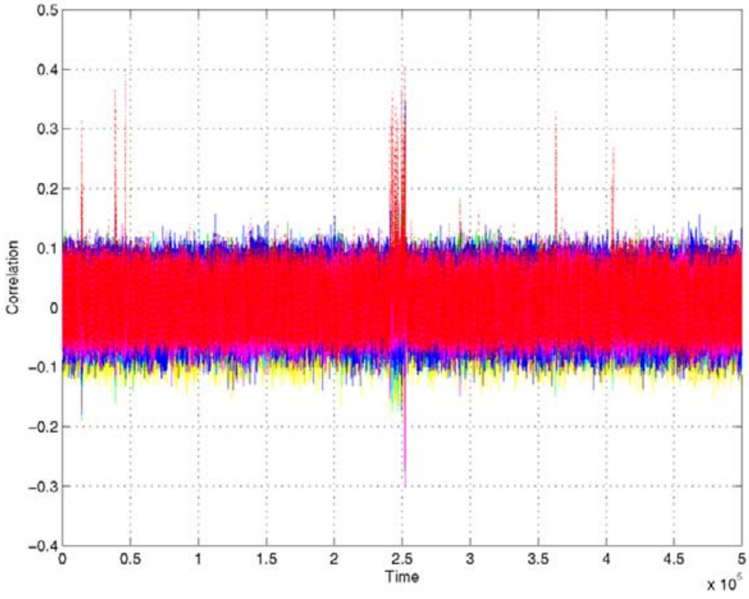


Figure XIV.4. Correlation of correct (dashed red trace) and incorrect guesses for 8 bits

the remainder of $\sqrt{}x_P$, and so one of the coordinates of the users secret key, a similar approach, where the relevant sections of $\sqrt{}x_P$ and x_Q are focused on, can be performed.

4. Fault Analysis of Pairings

Analysis of pairings from the point of view of fault attacks, presents both a problem and an opportunity: realizing an attack is typically more complicated in comparison to power analysis attacks, but there are more potential avenues of attack. One can, for example, easily identify the following three types of faults which can be manifested as:

1. Faulty inputs to the pairing, for example the inputs might not reside in the correct group.
2. Faults in any intermediate values, for example the so-called Miller variable which acts as a form of accumulator for the operation.
3. Faults in any pre-computed values or parameters, for example group parameters such as the order or pre-computed generator.

Thus far, only two main research papers have investigated these issues. The first, by Page and Vercauteren [PV06] in 2004, formulates an attack based on the third type of fault above. More specifically, they investigate the Duursma-Lee algorithm which was the first in an ongoing lineage of closed form algorithms for pairing evaluation over specific curves of characteristic three.

The second paper by Whelan and Scott [WS07] in 2007, examined the second type of fault. Their strategy was to analyze the implications of corrupting intermediate values within the pairing. They assessed three pairing algorithms, namely the η, Weil and Tate pairing, which were chosen based on the varying complexity of their final exponentiation. This work, along with [GHV07,PV06], highlight the importance of the final exponentiation and how the composition of it impacts on the security of the pairing.

The two attacks follow a similar approach. This is to execute the pairing twice given the same input elliptic curve points. In one of the pairing executions the pairing executes normally. This is termed a valid pairing execution and denoted as per normal by $e(P, Q)$. In the other, a fault with a particular effect is injected. This is termed a faulty pairing execution. The valid pairing is then divided by the faulty pairing. The parts of the pairing not affected by the fault will be canceled by the division. The remaining parts of the pairing are then manipulated in an effort to extract the secret elliptic curve point of interest. The structure and content of the remaining parts of the pairing affected by the fault, and the ability to extract the secret from this information, depends on the fault. In this section, the attacks of [PV06] and [WS07], which examine two such exploitable locations in the pairing execution, will be described.

4.1. Attacking the Order of the Miller Loop

The order of the Miller loop is chosen such that it satisfies a number of conditions. In the pairing it is the value that determines the number of rounds in the Miller loop and consequently the number and sequence of point double and additions. Attacking the order of the Miller loop results in the loop executing more times than it should. This attack was proposed in [PV06], which described it in the context of the Duursma-Lee algorithm for the Tate pairing. This attack will be presented here.

4.1.1. Case study: The Duursma-Lee algorithm for the Tate pairing

Duursma and Lee [DL03] introduced a closed form algorithm to compute pairings on a specific family of hyperelliptic curves, including supersingular curves in characteristic three. For $\mathbb{F}_q$ with $q = 3^m$, the supersingular curves used in cryptography are defined by an equation of the form $E : Y^2 = X^3 - X + b$, with $b = \pm 1$. Let $\mathbb{F}_{q^3} = \mathbb{F}_q[\rho]/(\rho^3 - \rho - b)$, with $b = \pm 1$ depending on the curve equation, and let $\mathbb{F}_{q^6} = \mathbb{F}_{q^3}[\sigma]/(\sigma^2 + 1)$.

Given the secret point $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$, a point chosen by the attacker, assume for the moment that we can eliminate the effect of the final powering. Let $\overline{e}_\Delta$ denote the Duursma-Lee algorithm where, through tampering, some transient fault causes the loop bound m to be replaced with Δ. Given such a pairing, instead of producing a product of polynomials of the form

$$\prod_{i=1}^{m} \left[(-y_P^{3^i} \cdot y_Q^{1/3^{i-1}} \sigma - (x_P^{3^i} + x_Q^{1/3^{i-1}} + b)^2) - (x_P^{3^i} + x_Q^{1/3^{i-1}} + b)\rho - \rho^2 \right]$$

one computes

$$\prod_{i=1}^{\Delta} \left[(-y_P^{3^i} \cdot y_Q^{1/3^{i-1}} \sigma - (x_P^{3^i} + x_Q^{1/3^{i-1}} + b)^2) - (x_P^{3^i} + x_Q^{1/3^{i-1}} + b)\rho - \rho^2 \right]$$

for some value Δ (given we currently neglect the final powering).

If one were able to inject a fault so that $\Delta = m + 1$, then recovering the secret point is trivial: we compute one correct pairing R_1 and one faulty pairing R_2 via

$$\begin{aligned} R_1 &= e_m(P, Q) \\ R_2 &= \overline{e}_{m+1}(P, Q)\,. \end{aligned}$$

If we use $g_{(i)}$ to denote the i-th factor of a product produced by the Duursma-Lee algorithm, dividing the two results above leaves a single factor

$$g_{(m+1)} = (-y_P^{3^{m+1}} \cdot y_Q^{1/3^m} \sigma - (x_P^{3^{m+1}} + x_Q^{1/3^m} + b)^2) - (x_P^{3^{m+1}} + x_Q^{1/3^m} + b)\rho - \rho^2.$$

Since we have that $z^{3^m} = z^{1/3^m} = z$ for all elements $z \in \mathbb{F}_q$, we can extract x_P or y_P, given that we know x_Q and y_Q, and hence reconstruct the secret point. However, the ability to selectively induce a controlled fault so $\Delta = m + 1$ seems tenuous. It is far more realistic to assume that we can induce $\Delta = m \pm l$ for random, unknown values of l. Using this ability, we calculate many erroneous pairing values with the aim of collecting a pair

$$\begin{aligned} R_1 &= \overline{e}_{m\pm l}(P, Q) \\ R_2 &= \overline{e}_{m\pm l+1}(P, Q)\,, \end{aligned}$$

so that after division we are again left with a single term of the product $g_{(m\pm l+1)}$ and can hence use a similar approach as above; clearly the only difference is the need to compensate for the differing powers of x_P, y_P, x_Q and y_Q introduced by l.

Of course this means we need to know what value l takes as a result of the injected fault, i.e. when the faults provoke usable values R_1 and R_2. Fortunately, since the algorithm is control-flow invariant, given the time taken to compute the pairing or a power trace of execution it is generally straightforward to recover how many loop iterations were performed.[2] We posit that finding suitable R_1 and R_2 requires a reasonably small number invocations of the pairing on the target device due to a similar argument as the birthday paradox.

The remaining problem is to reverse the final powering which takes the output of the pairing and produces a unique representative for the coset in $\mathbb{F}_{q^k}^*/(\mathbb{F}_{q^k}^*)^r$. To reverse this operation given the result $R = e(P, Q)$ we want to recover S, the value that was computed by the algorithm before the final powering so that $R = S^{(q^k-1)/r}$; for the Duursma-Lee algorithm this simplifies to $R = S^{q^3-1}$. It is clear that given R, the value of S is not uniquely determined, but only up to a non-zero factor in $\mathbb{F}_{q^3}$. Indeed, for non-zero $c \in \mathbb{F}_{q^3}$ we have $c^{q^3-1} = 1$. Furthermore, given one solution $S \in \mathbb{F}_{q^6}$ to $X^{q^3-1} - R = 0$, all others are of the form $c \cdot S$ for non-zero $c \in \mathbb{F}_{q^3}$.

However, given the attack description above we are not trying to reverse the powering on the full product of factors computed by the Duursma-Lee algorithm, rather only one of these factors which has a fairly special form. That is, given

$$R = \frac{R_2}{R_1} = \frac{\overline{e}_{m\pm l+1}(P, Q)}{\overline{e}_{m\pm l}(P, Q)} = g_{(m\pm l+1)}^{q^3-1}$$

[2]Note that in situations where DPA protections that randomize the execution are implemented, determining the number of loop iterations will be more difficult.

we want to recover $g_{(m\pm l+1)}$ which will, in turn, allow us to recover the correct x_P and y_P for the secret point. We therefore need a method to compute one valid root of $R = g^{q^3-1}$ for some factor g, and then derive the correct value of g from among all possible solutions. The first problem can be solved very efficiently as follows. First multiply the equation $X^{q^3-1} - R = 0$ by X to obtain

$$X^{q^3} - R \cdot X = 0$$

and note that the operator $X^{q^3} - R \cdot X$ is a linear operator on the two dimensional vector space $\mathbb{F}_{q^6}/\mathbb{F}_{q^3}$. Since $\mathbb{F}_{q^6} \cong \mathbb{F}_{q^3}[\sigma]/(\sigma^2 - 1)$ we can write $X = x_0 + \sigma x_1$ and $R = r_0 + \sigma r_1$, with $x_0, x_1, r_0, r_1 \in \mathbb{F}_{q^3}$. Using this representation, we see that the above equation is equivalent to

$$M \cdot X = \begin{pmatrix} 1 - r_0 & r_1 \\ r_1 & 1 + r_0 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = 0 \,.$$

The kernel of the matrix M is a one-dimensional vector space over $\mathbb{F}_{q^3}$ and thus provides all the solutions to $X^{q^3-1} - R = 0$.

To choose the correct root among all $q^3 - 1$ possibilities, we use the specific form of the factors in the product computed in the Duursma-Lee algorithm. Indeed, each factor g is of the form

$$g = g_0 + g_1\rho - \rho^2 + g_2\sigma$$

with $g_0, g_1, g_2 \in \mathbb{F}_q$. To recover g from $R = g^{q^3-1}$, we first obtain $g' = c \cdot g$ for some $c \in \mathbb{F}_{q^3}$ using the root finding algorithm above, and then compute c^{-1} and thus g itself. Again, this boils down to a simple linear system of equations: by multiplying g' with an appropriate factor in $\mathbb{F}_{q^3}$ we can assume that g' is of the form $g' = 1 + (g'_0 + g'_1\rho + g'_2\rho^2)\sigma$. Let $d = c^{-1} = g/g' \in \mathbb{F}_{q^3}$, then d clearly is of the form $d = d_0 + d_1\rho - \rho^2$. To determine $d_0, d_1 \in \mathbb{F}_q$, we use the fact that the terms $\rho\sigma$ and $\rho^2\sigma$ do not appear in g. This finally gives the following linear system of equations

$$\begin{pmatrix} g'_1 & g'_0 + g'_2 \\ g'_2 & g'_1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \end{pmatrix} = \begin{pmatrix} g'_1 + g'_2 \\ g'_0 + g'_2 \end{pmatrix} \,.$$

4.2. Attacking the Miller Variable

Attacking the Miller variable in the computation involves corrupting either the Miller variable itself, which is multiplicatively accumulated and eventually returned from Miller's algorithm, or the line functions evaluated in each round that contribute to the Miller variable. This attack, proposed in [WS07], demonstrates that the final exponentiation is a feature of the pairing which inhibits attacks corrupting the Miller variable. Here we will examine two pairings, the η and Tate pairing.

For this section a faulty pairing will be denoted by $e(P, Q)'$. The notation $[a_0][a_1][...]$ $[a_{k-1}]$ will also be used to represent the storage of field elements $a \in \mathbb{F}_{q^k}$. For example, $a \in \mathbb{F}_{2^{4m}}$ will be represented as $a_0 + a_1x + a_2x^2 + a_3x^3$ with $a_i \in \mathbb{F}_{2^m}$. Each component of this representation will be stored in four different memory locations and referred to as a cell. A fault can target any of the cells in memory, $[a_0][a_1][a_2]$ or $[a_3]$. For example, a fault may corrupt either a bit, bits, byte or bytes of $[a_3]$ to produce $[a_3]'$.

4.2.1. Case study: The η pairing

The η pairing $\eta(P, Q)$ [BGhS07] is defined for supersingular curves over finite fields of small characteristic. The main distinction between the η pairing and its siblings is that it chooses the order of the Miller loop r as a multiple of the group order such that it divides $q^k - 1$ to give a small factor, resulting in a simple final exponentiation. For example, consider the η pairing on a supersingular curve of characteristic two, if $q^k - 1 = 2^{4m} - 1$ and $r = 2^{2m} + 1$, which is a multiple of the order $2^m \pm 2^{(m+1)/2} + 1$, then the final exponentiation basically involves a conjugation and division, i.e. $(2^{2m} - 1)$.

Galbraith et al. [GhS07] described a variant of the η pairing, we refer to as the η_G pairing, which required no final exponentiation, that is the result of the pairing is a unique element and a bilinear map without any final exponentiation. This is enabled by the additional evaluation of vertical line functions (which the original η pairing [BGhS07] did not require). The implementation described considers supersingular elliptic curves over the binary field $\mathbb{F}_{2^m}$ with $k = 4$. The heart of the Miller loop is the calculation of the Miller variable f. In each round the line functions u and v are calculated and then divided to produce g, which is multiplicatively incorporated into the Miller variable. The lines u and v to produce g, are calculated as

$$u = [y_Q + y_A + \lambda(x_Q + x_A + 1)][\lambda + x_Q + 1][\lambda + x_Q][0]$$

and

$$v = [x_Q + x_C + 1][1][1][0] \ .$$

The cells of u and v will be denoted by $[u_0]$, $[u_1]$, $[u_2]$, $[u_3]$ and $[v_0]$, $[v_1]$, $[v_2]$, $[v_3]$ respectively.

There are a number of possible locations in which the fault can be injected, and a number of different effects that this fault can have. A fault can be injected into any of the cells of u or v, or any of the coordinates x_A, y_A, x_C, x_Q or y_Q and the fault injected can corrupt a bit or byte or multiple bits or bytes of the target. If a fault is injected into any of the cells of u or v, then the effect will be local, only corrupting the cell in question. If the fault is injected into one of the coordinates, then the resulting erroneous coordinate can, depending on whether precomputation is implemented or not [WS07], have consequences for all subsequent operations in which the erroneous coordinate is used. Here we will discuss what happens when one of the cells of v is corrupted.

Let $\eta_G(P, Q)'$ denote a corrupted pairing, where the fault is injected into the cell v_0 in the last round of the Miller loop. Note that the final round of Miller's algorithm is the optimal round to inject the fault since the squaring of the Miller variable, which happens in each round of the Miller loop, will not have to be reversed [WS07]. Division of the valid pairing by the faulty pairing will isolate the round in which the fault was injected to yield

$$\frac{\eta_G(P, Q)}{\eta_G(P, Q)'} = \frac{f^{2^l} \cdot g}{f^{2^l} \cdot g'} = \frac{g}{g'} = \frac{(u/v)}{(u/v')} = \frac{v'}{v} = \frac{[v_0]'[v_1][v_2][v_3]}{[v_0][v_1][v_2][v_3]} \ .$$

This division will produce an element in $\mathbb{F}_{2^{4m}}$, and can be thought of as four different cell values N_0, N_1, N_2 and N_3, where $N_i \in \mathbb{F}_{2^m}$. Therefore,

$$[v_0]'[v_1][v_2][v_3] = ([N_0][N_1][N_2][N_3])([v_0][v_1][v_2][v_3]). \tag{1}$$

Given $\eta_G(P, Q)$ and $\eta_G(P, Q)'$, the adversary can compute N_0, N_1, N_2 and N_3. Using knowledge of how multiplication in $\mathbb{F}_{2^{4m}}$ is performed, it can be determined which cells on the right-hand-side of Equation (1) correspond to the cell on the left-hand-side of the equation. For instance, the following three equations can be derived.

$$v_0' = N_0v_0 + N_1v_1 + N_2v_2 + (N_1 + N_3)v_1$$

$$v_1 = N_0v_0 + 2N_1v_1 + 2N_2v_2 + (N_0 + N_1)(v_0 + v_1) + (N_1 + N_3)v_1 + (N_2 + N_3)v_2 \tag{2}$$

$$v_2 = N_0v_0 + N_1v_1 + 2N_2v_2 + (N_2 + N_3)v_2 + (N_0 + N_2)(v_0 + v_2) \tag{3}$$

Since $v_1 = 1$ and $v_2 = 1$, Equations (2) and (3) can be simplified to

$$v_0 = \frac{N_0 + 4N_1 + 3N_2 + 2N_3 + 1}{2N_0 + N_1}$$

or

$$v_0 = \frac{N_0 + N_1 + 4N_2 + N_3 + 1}{2N_0 + N_2},$$

where only one unknown piece of data remains, v_0. Since v_0 is equal to $[x_Q + x_C + 1]$, the coordinate x_C, which relates to the secret point P, can be extracted using the adversaries knowledge of x_Q.[3] Numerical examples of this and other fault attacks on the η pairing can be found in [WS07].

4.3. The Final Exponentiation

The Tate pairing $e(P, Q)$, has the most complex final exponent of $(q^k - 1)/r$. This not only ensures that the output of the pairing is unique, but it also serves as a defensive mechanism against fault attacks as will now be demonstrated. The implementation assessed considers non-supersingular elliptic curves over the prime field $\mathbb{F}_p$ with $k = 2$ and a final exponent of $(p^2 - 1)/r$. In each round, the line function

$$u = [y_A - \lambda(x_Q + x_A)][-y_Q] \ .$$

is multiplicatively incorporated into the Miller variable.

Similar to the attack of the η pairing, there are a number of different locations a fault can target. Here, we examine the effect of a sign change fault [BOS06], which is where the sign bit of a value is flipped. This is a more powerful and targeted attack, which was

[3]The coordinate x_C relates to one of the points iP calculated during the calculation of rP. Hence, once the adversary extracts x_C, they can extract iP, then derive P depending on the iteration of the Miller loop attacked.

shown to be successful on the Weil pairing [WS07]. In the case of the BKLS algorithm for the Tate pairing considered here, the fault targets the sign of $[u_1]$, flipping the sign bit of the coordinate y_Q, in the final round of the Miller loop.

Assuming that one valid pairing $e(P,Q)$ and one faulty pairing $e(P,Q)'$ with the desired fault can be calculated, division of the valid pairing by a faulty pairing will yield the following:

$$\frac{e(P,Q)}{e(P,Q)'} = \frac{(f^{2^l} \cdot g)^{\frac{p^2-1}{r}}}{(f^{2^l} \cdot g')^{\frac{p^2-1}{r}}} = \frac{(u)^{\frac{p^2-1}{r}}}{(u')^{\frac{p^2-1}{r}}} = \left(\frac{[y_A - \lambda(x_Q + x_A)][-y_Q]}{[y_A - \lambda(x_Q + x_A)][y_Q]}\right)^{\frac{p^2-1}{r}} .$$

Since the exponent can be broken up into $(p-1)[(p+1)/r]$, and performed in parts, we can also attempt to reverse it in parts. Raising an element in $\mathbb{F}_{p^2}$ to the power of $p-1$ is simply a conjugation and division,

$$\frac{e(P,Q)}{e(P,Q)'} = \left(\frac{\frac{([y_A - \lambda(x_Q + x_A][y_Q])}{([y_A - \lambda(x_Q + x_A][-y_Q])}}{\frac{([y_A - \lambda(x_Q + x_A][-y_Q])}{([y_A - \lambda(x_Q + x_A][y_Q])}}\right)^{\frac{p+1}{r}}$$

which is equivalent to

$$\frac{e(P,Q)}{e(P,Q)'} = \left(\left(\frac{[y_A - \lambda(x_Q + x_A)][y_Q]}{[y_A - \lambda(x_Q + x_A)][-y_Q]}\right)^2\right)^{\frac{p+1}{r}} .$$

Therefore,

$$\sqrt{\frac{e(P,Q)}{e(P,Q)'}} = \pm\left(\frac{[y_A - \lambda(x_Q + x_A)][y_Q]}{[y_A - \lambda(x_Q + x_A)][-y_Q]}\right)^{\frac{p+1}{r}} .$$

In order to reverse the remaining exponent $(p+1)/r$, access the factor

$$\frac{[y_A - \lambda(x_Q + x_A)][y_Q]}{[y_A - \lambda(x_Q + x_A)][-y_Q]},$$

and subsequently derive the secret elliptic curve point, a specific n-th root, where $n = (p+1)/r$, must be calculated. Since n divides the group order once, there exists a simple formula to compute a n-th root, i.e.

$$\left(\sqrt{\frac{e(P,Q)}{e(P,Q)'}}\right)^{n^{-1} \pmod{s}}$$

where $s = (p+1)/n$. However, the particular root of interest does not exhibit any special form, unlike the case of [PV06], and is a full quadratic element. Therefore, no extra information is available to aid in determining which is the root we are interested in. In addition, since there exists n n-th roots, the cost of computing and testing all n-th roots, renders this sign change fault attack of the Tate pairing infeasible.

5. Countermeasures

A number of suggestions have been made to secure pairings against power and fault type implementation attacks. Some of these approaches will be addressed here.

5.1. Power Analysis Countermeasures

In the context of power analysis attacks, the main approach to deter first-order power analysis is to break the link between known data and secret data so that $\beta \neq S(\alpha, k)$. This disables the adversary from predicting the output β. Masking and randomization are two well know techniques to achieving this effect [MOP07]. For pairings, the property of bilinearity easily allows a form of masking or point blinding to be incorporated into the pairing [PV06,Sco05a]. For instance, a pairing can be calculated as

$$e(P,Q) = \frac{e(P,Q+R)}{e(P,R)} \quad \text{or} \quad e(P,Q) = e(P,lQ)^{\frac{1}{l}}$$

where R is a random point in $E(\mathbb{F}_q)$ and l is a random value. These are simplified versions of the proposed countermeasures. They prevent the formation of suitable selection functions since the information (about the point Q) previously known is now unknown by the adversary. For instance, in practice, a random point R would not be chosen for each pairing as this would require two pairing computations. Instead, R and $S = e(P,R)^{-1}$ can be stored on the device with a new random l being introduced as

$$e(P,Q) = e(P,Q+lR) \cdot S^l \ . \tag{4}$$

In this approach, while there are still extra operations to perform, only one pairing computation is required [PV06].

Other approaches to deter first-order power analysis include masking the individual parts that make up the Miller variable [Sco05a,WS06] and the use of randomized projective coordinates [KTH$^+$06]. In terms of randomizing the Miller variable all components that make up the Miller variable are multiplied by random element in the base field. These extra multiplications will have no effect on the final pairing value since they will be eliminated by the final exponentiation. Using randomized projective coordinates involves rewriting the pairing algorithm in terms of $(X,Y,Z) = (lX,lY,lZ)$ for a random value l, which again will not affect the output of the pairing.

5.2. Fault Analysis Countermeasures

Apart from choosing a pairing algorithm which has a complex final exponentiation, a number of fault obfuscation and detection mechanisms can be used to prevent the described attacks. Fault detection mechanisms determine whether or not a fault has been injected. Numerous software and hardware mechanisms already exist to detect a fault attack that are not algorithm specific [BECN$^+$06]. For example, the simple act of executing the algorithm twice to check if the results are the same can be applied to any algorithm. In the context of pairings, the property of bilinearity can again be used to check whether

$$e(P,Q)^{st} = e(sP,tQ) \ .$$

If the result of the two pairings do not match, this will indicate that the computation has in some way been tampered with. The obvious drawback of this fault detection mechanism is that it is quite costly requiring two pairing computations, point scalar multiplication of two points and exponentiation of an element in the extension field $\mathbb{F}_{q^k}$.

Other more cost effective methods of fault detection involve checking whether the intermediate points used in the computation are still points on the prescribed elliptic curve. A check could be carried out in every round but this will require $2r$ evaluations. There is also a possibility that this check will not be reliable. For example, the sign change fault attack of the Weil pairing [WS07] returns a point that is still on the curve and so will pass this check.

Fault prevention mechanisms obfuscate the execution so that even if a fault has been injected, the effects of the fault are completely lost and so useless to an adversary. In the case of the attacks described in Section 4.1.1 and 4.2.1 information about the point Q is required to eventually derive information about the secret point P. Point blinding, like that described in Section 5.1, can also be used here to prevent the adversary from using Q to their advantage.

5.3. Cost Analysis of Countermeasures

The cost of implementing some of the suggested countermeasures can be expensive and hence will impact greatly on the efficiency of the pairing computation. Point blinding, as in Equation (4), requires an additional point scalar multiplication (lR), point addition ($Q+lR$) and exponentiation in the extension field $\mathbb{F}_{q^k}$ (S^l). Randomizing the Miller variable requires the multiplication of each of the components that make up the Miller variable by a random value. This requires nr extra multiplications in $\mathbb{F}_q$, where n is the number of components contributing to the Miller variable and r is the order of the Miller loop. The cost for randomized projective coordinates were presented in [KTH+06], which were demonstrated to be slightly more efficient than randomizing the Miller variable.

In terms of the implementation used for analysis in Section 3.2, point blinding (as in Equation (4)) was timed and tested for effectiveness in deterring first-order power analysis attacks. The computation time for the point blinding was 3.614s, which is an increase of 1.536s for a naive implementation of the η_T pairing. When examined using the same attack technique as that described in Section 3.2, the implementation demonstrated to be secure against first-order power analysis. A low correlation for the correct portion of $\sqrt{x_P}$, which was indistinguishable from other possibilities of $\sqrt{x_P}$, was witnessed. While this is advantageous from a security perspective, the increase in computation time may hinder the viability of pairings for resource constrained devices. Hence, either more cost effective countermeasures will have to be developed or efforts to improve the existing approaches will have to be made.

6. Concluding Remarks

Implementation attacks have implications for the security of pairings. If potential attacks are not identified in advance of the implementation and deployment of a particular pairing on a target device, there will be security consequences for the associated pairing-based scheme. Hence, it is necessary to scrutinize pairings in the context of real-world conditions so that attacks can be identified and efficient and effective countermeasures developed. We conclude this chapter by providing some recommendations for the secure implementation of pairings, which result from the existing research in the area. Since this is a relatively unexplored area to date, we also provide a list of topics for further research.

6.1. Recommendations and Further Research

We highlight a number of pairing implementation features that may introduce security weaknesses. Firstly, data related to the secret elliptic curve point should be prevented from interacting directly with data that is potentially known and computable by an adversary. This recommendation is related to the fundamental hypothesis of [GT02] and prevents the formulation of selection functions. In the context of pairings, these features will have to be re-examined depending on what exactly the secret is (i.e. it could be the second input elliptic curve point to the pairing [WS06]).

Secondly, the final exponentiation should be included and chosen such that reversal of it is difficult. A first analysis of the hardness of this problem is given in [Ver08a] and the current recommendation is to take r a proper divisor of $\Phi_k(q)$ with large cofactor. Thirdly, the Miller variable calculated in each round of the Miller loop should utilize the full field representation of this element. One feature that the attack of [PV06] exploited was the fact that some of the coefficients of the Miller variable were zero. This is additional information that should not be available to an adversary.

Finally, the intermediate points required for the computation of rP should not be precomputed, stored and looked up as part of the pairing algorithm. Instead, it is recommended that they are dynamically computed. Precomputation is a method often used to increase efficiency. However, it was demonstrated in [WS07] that it can actually facilitate attacks which tamper with the Miller variable. This is because if a fault is injected into a precomputed value, the effect of the fault will be contained within that value and can be isolated by an adversary.

6.2. Open Problems

6.2.1. Curve parametrization attacks

Ciet and Joye [CJ05] described two different fault attacks on ECC which work by computing point multiplications with invalid points and erroneous field arithmetic. Since Miller's algorithm is essentially a point multiplication with some auxiliary operations, translating these concepts to the context of pairings seems interesting. However, such translation is difficult as a result of the fragility described earlier: using invalid points or curves will typically cause the pairing to become degenerate or produce garbage results.

6.2.2. Special point attacks

Goubin [Gou03] proposed an attack on ECC point multiplication, later enhanced by Akishita and Takagi [AT03], where so-called special points are used as input. These points trigger features in execution that, when carried through to the result, reveal information about the scalar multiplier. Clearly the attacks do not apply directly to the context of pairings since there is no scalar multiplier to reveal. However, it is not clear what happens if either input to the pairing is a special point in some more pairing-specific sense.

6.2.3. High-order power attacks

Second-order power analysis [Mes00,WW04] and template attacks [ARRS05,OM07] are types of side-channel attacks more powerful than first-order power analysis. In particular, these attacks have shown to bypass randomization and masking countermeasures mechanisms developed to deter first-order attacks. Therefore, further examination of the proposed countermeasures must be performed to determine whether they succumb to or withstand other forms of power analysis attacks.

Identity-Based Cryptography
M. Joye and G. Neven (Eds.)
IOS Press, 2009
© 2009 The authors and IOS Press. All rights reserved.

Bibliography

[AAB+97] Hal Abelson, Ross Anderson, Steven M. Bellovin, Josh Benaloh, Matt Blaze, Whitfield Diffie, John Gilmore, Peter G. Neumann, Ronald L. Rivest, Jeffrey I. Schiller, and Bruce Schneier. The risks of key recovery, key escrow, and trusted third-party encryption. *World Wide Web Journal*, 2(3):241–257, 1997.

[AABN02] Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 418–433. Springer-Verlag, 2002.

[ACD+06] Michel Abdalla, Dario Catalano, Alex Dent, John Malone-Lee, Gregory Neven, and Nigel Smart. Identity-based encryption gone wild. In Michele Bugliesi et al., editors, *Automata, Languages and Programming (ICALP 2006), Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 300–311. Springer-Verlag, 2006.

[ACJT06] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. Remarks on "Analysis of one popular group signature scheme" in Asiacrypt 2006. Cryptology ePrint Archive, Report 2006/464, 2006.

[ACL+07] Man Ho Au, Jing Chen, Joseph K. Liu, Yi Mu, Duncan S. Wong, and Guomin Yang. Malicious KGC attacks in certificateless cryptography. In Feng Bao and Steven Miller, editors, *2nd ACM Symposium on Information, Computer and Communications Security (ASIACCS 2007)*, pages 302–311. ACM Press, 2007.

[AF07] Masayuki Abe and Serge Fehr. Perfect NIZK with adaptive soundness. In Salil P. Vadhan, editor, *Theory of Cryptography Conference (TCC 2007)*, volume 4392 of *Lecture Notes in Computer Science*, pages 118–136. Springer-Verlag, 2007.

[AFG+06] Nuttapong Attrapadung, Jun Furukawa, Takeshi Gomi, Goichiro Hanaoka, Hideki Imai, and Rui Zhang. Efficient identity-based encryption with tight security reduction. In David Pointcheval, Yi Mu, and Kefei Chen, editors, *Cryptology and Network Security (CANS 2006)*, volume 4301 of *Lecture Notes in Computer Science*, pages 19–36. Springer-Verlag, 2006.

[AGKS05] Masayuki Abe, Rosario Gennaro, Kaoru Kurosawa, and Victor Shoup. Tag-KEM/DEM: A new framework for hybrid encryption and a new analysis of Kurosawa-Desmedt KEM. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 128–146. Springer-Verlag, 2005.

[AHM07] Omran Ahmadi, Darrel Hankerson, and Alfred Menezes. Software implementation of arithmetic in $\mathbb{F}_{3^m}$. In Claude Carlet and Berk Sunar, editors, *Arithmetic of Finite Fields (WAIFI 2007)*, volume 4547 of *Lecture Notes in Computer Science*, pages 85–102. Springer-Verlag, 2007.

[AM93] Arthur O. L. Atkin and François Morain. Elliptic curves and primality proving. *Mathematics of Computation*, 61(203):29–68, 1993.

[AMN01] Michel Abdalla, Sara K. Miner, and Chanathip Namprempre. Forward-secure threshold signature schemes. In David Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 441–456. Springer-Verlag, 2001.

[And00] Ross Anderson. Two remarks on public-key cryptology. Manuscript. Relevant material presented by the author in an invited lecture at ACM CCS 97: 4th ACM Conference on Computer and Communications Security, 2000.

[AR04] Sattam S. Al-Riyami. *Cryptographic Schemes Based on Elliptic Curve Pairings*. PhD thesis, Royal Holloway, University of London, 2004.

[ARP03] Sattam S. Al-Riyami and Kenneth G. Paterson. Certificateless public key cryptography. In Chi-Sung Laih, editor, *Advances in Cryptology – ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 452–473. Springer-Verlag, 2003.

[ARP05] Sattam S. Al-Riyami and Kenneth G. Paterson. CBE from CL-PKE: A generic construction and efficient schemes. In Serge Vaudenay, editor, *Public Key Cryptography – PKC 2005*, volume 3386 of *Lecture Notes in Computer Science*, pages 398–415. Springer-Verlag, 2005.

[ARRS05] Dakshi Agrawal, Josyula R. Rao, Pankaj Rohatgi, and Kai Schramm. Templates as master keys. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2005*, volume 3659 of *Lecture Notes in Computer Science*, pages 15–29. Springer-Verlag, 2005.

[AT83] Selim G. Akl and Peter D. Taylor. Cryptographic solution to a multilevel security problem. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *Advances in Cryptology – CRYPTO '82*, pages 237–249. Plenum Press, 1983.

[AT03] Toru Akishita and Tsuyoshi Takagi. Zero-value point attacks on elliptic curve cryptosystem. In Colin Boyd and Wenbo Mao, editors, *Information Security (ISC 2003)*, volume 2851 of *Lecture Notes in Computer Science*, pages 218–233. Springer-Verlag, 2003.

[AT07] Roberto Avanzi and Nicolas Thériault. Effects of optimizations for software implementations of small binary field arithmetic. In Claude Carlet and Berk Sunar, editors, *Arithmetic of Finite Fields (WAIFI 2007)*, volume 4547 of *Lecture Notes in Computer Science*, pages 69–84. Springer-Verlag, 2007.

[BB04a] Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer-Verlag, 2004.

[BB04b] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 443–459. Springer-Verlag, 2004.

[BB04c] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73. Springer-Verlag, 2004.

[BB08] Dan Boneh and Xavier Boyen. Short signatures without random oracles and the SDH assumption in bilinear groups. *Journal of Cryptology*, 21(2):149–177, 2008.

[BBBP08] Alessandro Barenghi, Guido Bertoni, Luca Breveglieri, and Gerardo Pelosi. A FPGA coprocessor for the cryptographic Tate pairing over $\mathbb{F}_p$. In *5th International Conference on Information Technology: New Generations (ITNG 2008)*, pages 112–119. IEEE Computer Society, 2008.

[BBD$^+$07] Jean-Luc Beuchat, Nicolas Brisebarre, Jérémie Detrey, Eiji Okamoto, Masaaki Shirase, and Tsuyoshi Takagi. Algorithms and arithmetic operators for computing the η_T pairing in characteristic three. Cryptology ePrint Archive, Report 2007/417, 2007.

[BBD$^+$08] Jean-Luc Beuchat, Nicolas Brisebarre, Jérémie Detrey, Eiji Okamoto, and Francisco Rodríguez-Henríquez. A comparison between hardware accelerators for the modified Tate pairing over $\mathbb{F}_{2^m}$ and $\mathbb{F}_{3^m}$. Cryptology ePrint Archive, Report 2008/115, 2008.

[BBDO07] Jean-Luc Beuchat, Nicolas Brisebarre, Jeremie Detrey, and Eiji Okamoto. Arithmetic operators for pairing-based cryptography. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems – CHES 2007*, volume 4727 of *Lecture Notes in Computer Science*, pages 239–255. Springer-Verlag, 2007.

[BBG05] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer-Verlag, 2005.

[BBH06] Dan Boneh, Xavier Boyen, and Shai Halevi. Chosen ciphertext secure public key threshold encryption without random oracles. In David Pointcheval, editor, *Topics in Cryptology – CT-RSA 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 226–243. Springer-Verlag, 2006.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer*

Science, pages 41–55. Springer-Verlag, 2004.

[BBS$^+$08] Jean-Luc Beuchat, Nicolas Brisebarre, Masaaki Shirase, Tsuyoshi Takagi, and Eiji Okamoto. A coprocessor for the final exponentiation of the η_T pairing in characteristic three. Cryptology ePrint Archive, Report 2007/045, 2008.

[BCH93] Hannes Brunner, Andreas Curiger, and Max Hofstetter. On computing multiplicative inverses in $\mathrm{GF}(2^m)$. *IEEE Transactions on Computers*, 42(8):1010–1015, 1993.

[BCHK06] Dan Boneh, Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 36(5):915–942, 2006.

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In Marc Joye and Jean-Jacques Quisquater, editors, *Cryptographic Hardware and Embedded Systems – CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 16–29. Springer-Verlag, 2004.

[BDL97] Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the importance of checking cryptographic protocols for faults (extended abstract). In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 37–51. Springer-Verlag, 1997.

[BDOP04] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 506–522. Springer-Verlag, 2004.

[BDPR98] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45. Springer-Verlag, 1998.

[BDT04] Dan Boneh, Xuhua Ding, and Gene Tsudik. Fine-grained control of security capabilities. *ACM Transactions on Internet Technology*, 4(1):60–82, 2004.

[BECN$^+$06] Hagai Bar-El, Hamid Choukri, David Naccache, Michael Tunstall, and Claire Whelan. The sorcerers apprentice guide to fault attacks. *Proceedings of the IEEE*, 94(2):370–382, 2006.

[Bei96] Amos Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. PhD thesis, Israel Institute of Technology, Technion, 1996.

[Ber68] Elwyn R. Berlekamp. *Algebraic Coding Theory*. McGraw-Hill, 1968.

[Ber01] Daniel J. Bernstein. A software implementation of NIST P-224. In *5th Workshop on Elliptic Curve Cryptography (ECC 2001)*, Waterloo, October 29–31, 2001. Slides and software available via `http://cr.yp.to/nistp224.html`.

[Ber06] Daniel J. Bernstein. Curve25519: New Diffie-Hellman speed records. In Moti Yung et al., editors, *Public Key Cryptography – PKC 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 207–228. Springer-Verlag, 2006.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer-Verlag, 2001.

[BF03] Dan Boneh and Matthew K. Franklin. Identity based encryption from the Weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.

[BFMLS08] Kamel Bentahar, Pooya Farshim, John Malone-Lee, and Nigel P. Smart. Generic constructions of identity-based and certificateless KEMs. *Journal of Cryptology*, 21(2):178–199, 2008.

[BGH07a] Dan Boneh, Craig Gentry, and Michael Hamburg. Space-efficient identity based encryption without pairings. In *48th Annual Symposium on Foundations of Computer Science (FOCS 2007)*, pages 647–657. IEEE Computer Society Press, 2007.

[BGH07b] Dan Boneh, Craig Gentry, and Michael Hamburg. Space-efficient identity based encryption without pairings. Cryptology ePrint Archive, Report 2007/177, 2007.

[BGhS07] Paulo S. L. M. Barreto, Steven D. Galbraith, Colm Ó' hÉigeartaigh, and Michael Scott. Efficient pairing computation on supersingular abelian varieties. *Designs, Codes and Cryptography*, 42(3):239–271, 2007.

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 416–432. Springer-Verlag, 2003.

[BGW05] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275. Springer-Verlag, 2005.

[BHS04] Robert W. Bradshaw, Jason E. Holt, and Kent E. Seamons. Concealing complex policies with hidden credentials. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick McDaniel, editors, *11th ACM Conference on Computer and Communications Security (CCS 2004)*, pages 146–157. ACM Press, 2004.

[BJN00] Dan Boneh, Antoine Joux, and Phong Q. Nguyen. Why textbook ElGamal and RSA encryption are insecure. In Tatsuaki Okamoto, editor, *Advances in Cryptology – ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 30–43. Springer-Verlag, 2000.

[BK05] Dan Boneh and Jonathan Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 87–103. Springer-Verlag, 2005.

[BKLS02] Paulo S. L. M. Barreto, Hae Yong Kim, Ben Lynn, and Michael Scott. Efficient algorithms for pairing-based cryptosystems. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 354–368. Springer-Verlag, 2002.

[BL90] Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In Shafi Goldwasser, editor, *Advances in Cryptology – CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 27–35. Springer-Verlag, 1990.

[BL07] Daniel J. Bernstein and Tanja Lange. Faster addition and doubling on elliptic curves. In Kaoru Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 29–50. Springer-Verlag, 2007.

[Bla79] G. R. Blakley. Safeguarding cryptographic keys. *Proceedings of AFIPS 1979 National Computer Conference*, 48:313–317, 1979.

[BLMQ05] Paulo S. L. M. Barreto, Benoît Libert, Noel McCullagh, and Jean-Jacques Quisquater. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In Bimal K. Roy, editor, *Advances in Cryptology – ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 515–532. Springer-Verlag, 2005.

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer-Verlag, 2001.

[BLS04] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. *Journal of Cryptology*, 17(4):297–319, 2004.

[BM99] Mihir Bellare and Sara K. Miner. A forward-secure digital signature scheme. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 431–448. Springer-Verlag, 1999.

[BM07] Xavier Boyen and Luther Martin. Identity-based cryptography standard (IBCS) #1: Supersingular curve implementations of the BF and BB1 cryptosystems. IETF RFC 5091, December 2007.

[BMvT78] Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. On the intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.

[BMW05] Xavier Boyen, Qixiang Mei, and Brent Waters. Direct chosen ciphertext security from identity-based techniques. In Vijayalakshmi Atluri, Catherine Meadows, and Ari Juels, editors, *12th ACM Conference on Computer and Communications Security (CCS 2005)*, pages 320–329. ACM Press, 2005.

[BN05] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In Bart Preneel and Stafford Tavares, editors, *Selected Areas in Cryptography (SAC 2005)*, volume 3897 of *Lecture Notes in Computer Science*, pages 319–331. Springer-Verlag, 2005.

[BNN04] Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Security proofs for identity-based identification and signature schemes. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 268–286. Springer-Verlag, 2004. Full version available from Cryptology ePrint Archive, Report 2004/252.

[BOS06] Johannes Blömer, Martin Otto, and Jean-Pierre Seifert. Sign change fault attacks on elliptic curve cryptosystems. In Luca Breveglieri et al., editors, *Fault Diagnosis and Tolerance in Cryptography (FDTC 2006)*, volume 4236 of *Lecture Notes in Computer Science*, pages 36–52.

Springer-Verlag, 2006.

[Boy03] Xavier Boyen. Multipurpose identity-based signcryption (a swiss army knife for identity-based cryptography). In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 383–399. Springer-Verlag, 2003.

[Boy07] Xavier Boyen. General ad hoc encryption from exponent inversion IBE. In Moni Naor, editor, *Advances in Cryptology – EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 394–411. Springer-Verlag, 2007.

[Boy08] Xavier Boyen. A tapestry of identity-based encryption: Practical frameworks compared. *International Journal of Applied Cryptography*, 1(1):3–21, 2008.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *1st ACM Conference on Computer and Communications Security (CCS '93)*, pages 62–73. ACM Press, 1993.

[Bri89] Ernest F. Brickell. Some ideal secret sharing schemes. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 6:105–113, 1989.

[Bro02] Daniel R. L. Brown. Generic groups, collision resistance, and ECDSA. Contributions to IEEE P1363a, February 2002. Updated version for "The Exact Security of ECDSA." Available from `http://grouper.ieee.org/groups/1363/`.

[BSNS05] Joonsang Baek, Reihaneh Safavi-Naini, and Willy Susilo. Certificateless public key encryption without pairing. In Jianying Zhou et al., editors, *Information Security (ISC 2005)*, volume 3650 of *Lecture Notes in Computer Science*, pages 134–148. Springer-Verlag, 2005.

[BSS99] Ian F. Blake, Gadiel Seroussi, and Nigel P. Smart. *Elliptic Curves in Cryptography*, volume 265 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 1999.

[BSS05] Ian F. Blake, Gadiel Seroussi, and Nigel P. Smart, editors. *Advances in Elliptic Curve Cryptography*, volume 317 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 2005.

[BSTO07a] Jean-Luc Beuchat, Masaaki Shirase, Tsuyoshi Takagi, and Eiji Okamoto. An algorithm for the η_T pairing calculation in characteristic three and its hardware implementation. In *18th IEEE Symposium on Computer Arithmetic (ARITH 2007)*, pages 97–104. IEEE Computer Society, 2007.

[BSTO07b] Jean-Luc Beuchat, Masaaki Shirase, Tsuyoshi Takagi, and Eiji Okamoto. A refined algorithm for the η_T pairing calculation in characteristic three. Cryptology ePrint Archive, Report 2007/311, 2007.

[BSW07] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *2007 IEEE Symposium on Security and Privacy*, pages 321–334. IEEE Computer Society Press, 2007.

[BW06] Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In Cynthia Dwork, editor, *Advances in Cryptology – CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 290–307. Springer-Verlag, 2006.

[BW07] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In Salil P. Vadhan, editor, *Theory of Cryptography Conference (TCC 2007)*, volume 4392 of *Lecture Notes in Computer Science*, pages 535–554. Springer-Verlag, 2007.

[BY03] Mihir Bellare and Bennet S. Yee. Forward-security in private-key cryptography. In Marc Joye, editor, *Topics in Cryptology – CT-RSA 2003*, volume 2612 of *Lecture Notes in Computer Science*, pages 1–18. Springer-Verlag, 2003.

[CBG06] Sherman S. M. Chow, Colin Boyd, and Juan Manuel González Nieto. Security-mediated certificateless cryptography. In Moti Yung et al., editors, *Public Key Cryptography – PKC 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 508–524. Springer-Verlag, 2006.

[CC98] Anne Canteaut and Florent Chabaud. A new algorithm for finding minimum-weight words in a linear code application to primitive narrow-sense BCH codes of length 511. *IEEE Transactions on Information Theory*, 44(1):367–378, 1998.

[CC03] Jae Choon Cha and Jung Hee Cheon. An identity-based signature from gap Diffie-Hellman groups. In Yvo Desmedt, editor, *Public Key Cryptography – PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 18–30. Springer-Verlag, 2003.

[CC05] Liqun Chen and Zhaohui Cheng. Security proof of Sakai-Kasahara's identity-based encryption scheme. Cryptology ePrint Archive, Report 2005/226, 2005.

[CCLC07] Zhaohui Cheng, Liqun Chen, Li Ling, and Richard Comley. General and efficient certificateless public key encryption constructions. In Tsuyoshi Takagi et al., editors, *Pairing-Based Cryptography – Pairing 2007*, volume 4575 of *Lecture Notes in Computer Science*, pages 83–107. Springer-Verlag, 2007.

[CCMLS05] Liqun Chen, Zhaohui Cheng, John Malone-Lee, and Nigel P. Smart. An efficient ID-KEM based on the Sakai-Kasahara key construction. Cryptology ePrint Archive, Report 2005/224, 2005.

[CDM05] Francis Crowe, Alan Daly, and William Marnane. Optimised Montgomery domain inversion on FPGA. In Finbarr O'Regan and Carsten Wegener, editors, *European Conference on Circuit Theory and Design (ECCTD 2005)*, volume I, pages 277–280. IEEE Press, 2005.

[CF05] Henri Cohen and Gerhard Frey, editors. *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. Chapman & Hall/CRC, 2005.

[CFH$^+$07] Yang Cui, Eiichiro Fujisaki, Goichiro Hanaoka, Hideki Imai, and Rui Zhang. Formal security treatments for signatures from identity-based encryption. In Willy Susilo, Joseph K. Liu, and Yi Mu, editors, *Provable Security*, volume 4786 of *Lecture Notes in Computer Science*, pages 218–227. Springer-Verlag, 2007.

[CFS01] Nicolas Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a McEliece-based digital signature scheme. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 157–174. Springer-Verlag, 2001.

[CGG07] Pierre-Louis Cayrel, Philippe Gaborit, and Marc Girault. Identity-based identification and signature schemes using correcting codes. In Daniel Augot, Nicolas Sendrier, and Jean-Pierre Tillich, editors, *International Workshop on Coding and Cryptography – WCC 2007*, pages 69–78. INRIA, 2007.

[CGGG08] Pierre-Louis Cayrel, Philippe Gaborit, David Galindo, and Marc Girault. Improved identity-based identification and signature scheme using correcting codes. Unpublished manuscript, 2008.

[CGP08] Pierre-Louis Cayrel, Philippe Gaborit, and Emmanuel Prouff. Secure implementation of the Stern authentication and signature schemes for low-resource devices. In Gilles Grimaud and François-Xavier Standaert, editors, *Smart Card Research and Advanced Applications (CARDIS 2008)*, volume 5189 of *Lecture Notes in Computer Science*, pages 191–205. Springer-Verlag, 2008.

[CH07] Jaewook Chung and M. Anwar Hasan. Asymmetric squaring formulae. In *18th IEEE Symposium on Computer Arithmetic (ARITH 2007)*, pages 113–122. IEEE Computer Society, 2007.

[Cha83] David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *Advances in Cryptology – CRYPTO '82*, pages 199–203. Plenum Press, 1983.

[Che06] Jung Hee Cheon. Security analysis of the strong Diffie-Hellman problem. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 1–11. Springer-Verlag, 2006.

[CHK03] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 255–271. Springer-Verlag, 2003.

[CHK04] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222. Springer-Verlag, 2004.

[CIKL05] Debra L. Cook, John Ioannidis, Angelos D. Keromytis, and Jake Luck. Cryptographics: Secret key cryptography using graphics cards. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 334–350. Springer-Verlag, 2005.

[CJ05] Mathieu Ciet and Marc Joye. Elliptic curve cryptosystems in the presence of permanent and transient faults. *Designs, Codes and Cryptography*, 36(1):33–43, 2005.

[CM07] Jean-Sébastien Coron and Alexander May. Deterministic polynomial-time equivalence of computing the RSA secret key and factoring. *Journal of Cryptology*, 20(1):39–50, 2007.

[CN07] Ling Cheung and Calvin Newport. Provably secure ciphertext policy ABE. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *14th ACM Conference on Computer and Communications Security (CCS 2007)*, pages 456–465. ACM Press, 2007.

[Coc01] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In Bahram Honary, editor, *Cryptography and Coding*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363. Springer-Verlag, 2001.

[Com90] Paul G. Comba. Exponentiation cryptosystems on the IBM PC. *IBM Systems Journal*, 29(4):526–538, 1990.

[COV07] Pierre-Louis Cayrel, Ayoub Otmani, and Damien Vergnaud. On Kabatianskii-Krouk-Smeets signatures. In Claude Carlet and Berk Sunar, editors, *Arithmetic of Finite Fields (WAIFI 2007)*, volume 4547 of *Lecture Notes in Computer Science*, pages 237–251. Springer-Verlag, 2007.

[CP01] Clifford Cocks and Richard G. E. Pinch. ID-based cryptosystems based on the Weil pairing. Unpublished manuscript, 2001.

[CRR08] Sherman S. M. Chow, Volker Roth, and Eleanor Rieffel. General certificateless encryption and timed-release encryption. In Rafail Ostrovsky, Roberto De Prisco, and Ivan Visconti, editors, *Security in Communication Networks (SCN 2008)*, volume 5229 of *Lecture Notes in Computer Science*, pages 126–143. Springer-Verlag, 2008.

[CS99] Ronald Cramer and Victor Shoup. Signature schemes based on the strong RSA assumption. In *6th ACM Conference on Computer and Communications Security (CCS '99)*, pages 46–51. ACM Press, 1999.

[CS03] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.

[CS05] Sanjit Chatterjee and Palash Sarkar. Trading time for space: Towards an efficient IBE scheme with short(er) public parameters in the standard model. In Dongho Won and Seungjoo Kim, editors, *Information Security and Cryptology – ICISC 2005*, volume 3935 of *Lecture Notes in Computer Science*, pages 424–440. Springer-Verlag, 2005.

[CS06a] Sanjit Chatterjee and Palash Sarkar. Generalization of the selective-ID security model for HIBE protocols. In Moti Yung et al., editors, *Public Key Cryptography – PKC 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 241–256. Springer-Verlag, 2006.

[CS06b] Sanjit Chatterjee and Palash Sarkar. HIBE with short public parameters without random oracle. In Xuejia Lai and Kefei Chen, editors, *Advances in Cryptology – ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 145–160. Springer-Verlag, 2006.

[CS06c] Sanjit Chatterjee and Palash Sarkar. New constructions of constant size ciphertext HIBE without random oracle. In Min Surp Rhee and Byoungcheon Lee, editors, *Information Security and Cryptology – ICISC 2006*, volume 4296 of *Lecture Notes in Computer Science*, pages 310–327. Springer-Verlag, 2006.

[CS07] Sanjit Chatterjee and Palash Sarkar. Constant size ciphertext HIBE in the augmented selective-ID model and its extensions. *Journal of Universal Computer Science*, 13(10):1367–1395, 2007.

[CSB04] Sanjit Chatterjee, Palash Sarkar, and Rana Barua. Efficient computation of Tate pairing in projective coordinate over general characteristic fields. In Choonsik Park and Seongtaek Chee, editors, *Information Security and Cryptology – ICISC 2004*, volume 3506 of *Lecture Notes in Computer Science*, pages 168–181. Springer-Verlag, 2004.

[Dal07] Léonard Dallot. Towards a concrete security proof of Courtois, Finiasz and Sendrier signature scheme. In *Western European Workshop on Research in Cryptology (WEWorc 2007)*, Bochum, July 4–6, 2007.

[DEM05] Régis Dupont, Andreas Enge, and François Morain. Building curves with arbitrary small MOV degree over finite prime fields. *Journal of Cryptology*, 18(2):79–89, 2005.

[Den02] Alexander W. Dent. Adapting the weaknesses of the random oracle model to the generic group model. In Yuliang Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 100–109. Springer-Verlag, 2002.

[Den08] Alexander W. Dent. A survey of certificateless encryption schemes and security models. *International Journal of Information Security*, 7(5):349–377, 2008.

[Des88] Yvo Desmedt. Society and group oriented cryptography: A new concept. In Carl Pomerance, editor, *Advances in Cryptology – CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*, pages 120–127. Springer-Verlag, 1988.

[Deu41] Max Deuring. Die Typen der Multiplikatorenringe elliptischer Funktionenkörper. *Abhandlungen aus dem Mathematischen Seminar der Hansischen Universität*, 14:197–272, 1941.

[DF03a] Yevgeniy Dodis and Nelly Fazio. Public key broadcast encryption for stateless receivers. In

Joan Feigenbaum, editor, *Digital Rights Management (DRM 2002)*, volume 2696 of *Lecture Notes in Computer Science*, pages 61–80. Springer-Verlag, 2003.

[DF03b] Yevgeniy Dodis and Nelly Fazio. Public key trace and revoke scheme secure against adaptive chosen ciphertext attack. In Yvo Desmedt, editor, *Public Key Cryptography – PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 100–115. Springer-Verlag, 2003.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[DhSD06] Augusto Jun Devegili, Colm Ó hÉigeartaigh, Michael Scott, and Ricardo Dahab. Multiplication and squaring on pairing-friendly fields. Cryptology ePrint Archive, Report 2006/471, 2006.

[DK05] Yevgeniy Dodis and Jonathan Katz. Chosen-ciphertext security of multiple encryption. In Joe Kilian, editor, *Theory of Cryptography Conference (TCC 2005)*, volume 3378 of *Lecture Notes in Computer Science*, pages 188–209. Springer-Verlag, 2005.

[DKXY03] Yevgeniy Dodis, Jonathan Katz, Shouhuai Xu, and Moti Yung. Strong key-insulated signature schemes. In Yvo Desmedt, editor, *Public Key Cryptography – PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 130–144. Springer-Verlag, 2003.

[DL03] Iwan M. Duursma and Hyang-Sook Lee. Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$. In Chi-Sung Laih, editor, *Advances in Cryptology – ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 111–123. Springer-Verlag, 2003.

[DLP08] Alexander W. Dent, Benoit Libert, and Kenneth G. Paterson. Certificateless encryption schemes strongly secure in the standard model. In Ronald Cramer, editor, *Public Key Cryptography – PKC 2008*, volume 4939 of *Lecture Notes in Computer Science*, pages 344–359. Springer-Verlag, 2008.

[DSD07] Augusto Jun Devegili, Michael Scott, and Ricardo Dahab. Implementing cryptographic pairings over Barreto-Naehrig curves. In Tsuyoshi Takagi et al., editors, *Pairing-Based Cryptography – Pairing 2007*, volume 4575 of *Lecture Notes in Computer Science*, pages 197–207. Springer-Verlag, 2007.

[DSS94] Digital signature standard (DSS). National Institute of Standards and Technology (NIST), FIPS PUB 186, U.S. Department of Commerce, 1994.

[DvW92] Whitfield Diffie, Paul C. van Oorschot, and Michael J. Wiener. Authentication and authenticated key exchanges. *Designs, Codes and Cryptography*, 2(2):107–125, June 1992.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[EOS06] D. Engelbert, R. Overbeck, and A. Schmidt. A summary of McEliece-type cryptosystems and their security. Cryptology ePrint Archive, Report 2006/162, 2006.

[FFS88] Uriel Feige, Amos Fiat, and Adi Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1(2):77–94, 1988.

[Fin04] Matthieu Finiasz. *Nouvelles constructions utilisant des codes correcteurs d'erreurs en cryptographie à clef publique*. PhD thesis, INRIA – Ecole Polytechnique, 2004.

[FMR99] Gerhard Frey, Michael Müller, and Hans-Georg Rück. The Tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. *IEEE Transactions on Information Theory*, 45(5):1717–1719, 1999.

[FN93] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491. Springer-Verlag, 1993.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554. Springer-Verlag, 1999.

[FO00] Eiichiro Fujisaki and Tatsuaki Okamoto. How to enhance the security of public-key encryption at minimum cost. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer*, E83-A(1):24–32, 2000.

[FR94] Gerhard Frey and Hans-Georg Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Mathematics of Computation*, 62(206):865–874, 1994.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer-Verlag, 1987.

[FST06] David Freeman, Michael Scott, and Edlyn Teske. A taxonomy of pairing-friendly elliptic curves. Cryptology ePrint Archive, Report 2006/372, 2006.

[Gal05a] Steven D. Galbraith. Pairings. In Ian F. Blake, Gadiel Seroussi, and Nigel P. Smart, editors, *Advances in Elliptic Curve Cryptography*, volume 317 of *London Mathematical Society Lecture Note Series*, pages 183–214. Cambridge University Press, 2005.

[Gal05b] David Galindo. Boneh-Franklin identity based encryption revisited. In Luís Caires et al., editors, *Automata, Languages and Programming (ICALP 2005)*, volume 3580 of *Lecture Notes in Computer Science*, pages 791–802. Springer-Verlag, 2005.

[Gau07] Pierrick Gaudry. Fast genus 2 arithmetic based on Theta functions. *Journal of Cryptology*, 1(3):243–265, 2007.

[Gen03] Craig Gentry. Certificate-based encryption and the certificate revocation problem. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 272–293. Springer-Verlag, 2003.

[Gen06] Craig Gentry. Practical identity-based encryption without random oracles. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464. Springer-Verlag, 2006.

[GG07] Philippe Gaborit and Marc Girault. Lightweight code-based identification and signature. In *IEEE International Symposium on Information Theory (ISIT 2007)*, pages 191–195. IEEE Press, 2007.

[GHK06] David Galindo, Javier Herranz, and Eike Kiltz. On the generic construction of identity-based signatures with additional properties. In Xuejia Lai and Kefei Chen, editors, *Advances in Cryptology – ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 178–193. Springer-Verlag, 2006.

[GHO$^+$07] Robert Granger, Florian Hess, Roger Oyono, Nicolas Thériault, and Frederik Vercauteren. Ate pairing on hyperelliptic curves. In Moni Naor, editor, *Advances in Cryptology – EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 430–447. Springer-Verlag, 2007.

[GHR99] Rosario Gennaro, Shai Halevi, and Tal Rabin. Secure hash-and-sign signatures without the random oracle. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 123–139. Springer-Verlag, 1999.

[GHS02] Steven D. Galbraith, Keith Harrison, and David Soldera. Implementing the Tate pairing. In Claus Fieker and David R. Kohel, editors, *Algorithmic Number Theory (ANTS V)*, volume 2369 of *Lecture Notes in Computer Science*, pages 324–337. Springer-Verlag, 2002.

[GhS07] Steven Galbraith, Colm Ó hÉigeartaigh, and Caroline Sheedy. Simplified pairing computation and security implications. *Journal of Mathematical Cryptology*, 1(3):267–282, 2007.

[GHV07] Steven Galbraith, Florian Hess, and Frederik Vercauteren. Aspects of pairing inversion. Cryptology ePrint Archive, Report 2007/256, 2007.

[Gir91] Marc Girault. Self-certified public keys. In Donald W. Davies, editor, *Advances in Cryptology – EUROCRYPT '91*, volume 547 of *Lecture Notes in Computer Science*, pages 490–497. Springer-Verlag, 1991.

[GJPS08] Vipul Goyal, Abhishek Jain, Omkant Pandey, and Amit Sahai. Bounded ciphertext policy attribute based encryption. In Luca Aceto et al., editors, *Automata, Languages and Programming (ICALP 2008), Part II*, volume 5126 of *Lecture Notes in Computer Science*, pages 579–591. Springer-Verlag, 2008.

[GK08] Shay Gueron and Michael E. Kounavis. Carry-less multiplication and its usage for computing the GCM mode. White paper, Intel Corporation, 2008.

[GMO01] Karine Gandolfi, Christophe Mourtel, and Francis Olivier. Electromagnetic analysis: Concrete results. In Çetin Kaya Koç, David Naccache, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*, pages 251–261. Springer-Verlag, 2001.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.

[GMV07] Steven D. Galbraith, James F. McKee, and Paula C. Valença. Ordinary abelian varieties having small embedding degree. *Finite Fields and their Applications*, 13(4):800–814, 2007.

[GNO$^+$04] Rita Gavriloaie, Wolfgang Nejdl, Daniel Olmedilla, Kent E. Seamons, and Marianne Winslett. No registration needed: How to use declarative policies and negotiation to access sensitive resources on the semantic web. In Christoph Bussler et al., editors, *The Semantic Web: Research and Applications (ESWS 2004)*, volume 3053 of *Lecture Notes in Computer Science*, pages 342–356. Springer-Verlag, 2004.

[Gou03] Louis Goubin. A refined power-analysis attack on elliptic curve cryptosystems. In Yvo Desmedt, editor, *Public Key Cryptography – PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 199–210. Springer-Verlag, 2003.

[Goy07] Vipul Goyal. Reducing trust in the PKG in identity based cryptosystems. In Alfred Menezes, editor, *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 430–447. Springer-Verlag, 2007.

[GP05] Philipp Grabher and Dan Page. Hardware acceleration of the Tate pairing in characteristic three. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2005*, volume 3659 of *Lecture Notes in Computer Science*, pages 398–411. Springer-Verlag, 2005.

[GPS06] Robert Granger, Dan Page, and Martijn Stam. On small characteristic algebraic tori in pairing-based cryptography. *LMS Journal of Computation and Mathematics*, 9:64–85, 2006.

[GPS07] Elisa Gorla, Christoph Puttmann, and Jamshid Shokrollahi. Explicit formulas for efficient multiplication in $\mathrm{GF}(3^{6m})$. In Carlisle Adams, Ali Miri, and Michael Wiener, editors, *Selected Areas in Cryptography (SAC 2007)*, volume 4876 of *Lecture Notes in Computer Science*, pages 173–183. Springer-Verlag, 2007.

[GPS08] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *13th ACM Conference on Computer and Communications Security (CCS 2006)*, pages 89–98. ACM Press, 2006. Available as Cryptology ePrint Archive Report 2006/309.

[GPV07] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. Cryptology ePrint Archive, Report 2007/432, 2007.

[GQ90] Louis C. Guillou and Jean-Jacques Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In Shafi Goldwasser, editor, *Advances in Cryptology – CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 216–231. Springer-Verlag, 1990.

[Gra07] Torbjörn Granlund. Instruction latencies and throughput for AMD and Intel x86 processors. `http://swox.com/doc/x86-timing.pdf`, 2007.

[GS02] Craig Gentry and Alice Silverberg. Hierarchical ID-based cryptography. In Yuliang Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer-Verlag, 2002.

[GST04] Michael T. Goodrich, Jonathan Z. Sun, and Roberto Tamassia. Efficient tree-based revocation in groups of low-state devices. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 511–527. Springer-Verlag, 2004.

[GSW00] Juan A. Garay, Jessica Staddon, and Avishai Wool. Long-lived broadcast encryption. In Mihir Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 333–352. Springer-Verlag, 2000.

[GT02] Jovan Dj. Golic and Christophe Tymen. Multiplicative masking and power analysis of AES. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 198–212. Springer-Verlag, 2002.

[Gün90] Christoph G. Günther. An identity-based key-exchange protocol. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *Advances in Cryptology – EUROCRYPT '89*, volume 434 of *Lecture Notes in Computer Science*, pages 29–37. Springer-Verlag, 1990.

[GZ05] Chunxiang Gu and Yuefei Zhu. An ID-based verifiable encrypted signature scheme based on Hess's scheme. In Dengguo Feng, Dongdai Lin, and Moti Yung, editors, *Information Security and Cryptology (CISC 2005)*, volume 3822 of *Lecture Notes in Computer Science*, pages 42–52. Springer-Verlag, 2005.

[HCM01] Hugh Harney, Andrea Colgrove, and Patrick McDaniel. Principles of policy in secure groups. In

ISOC Network and Distributed System Security Symposium – NDSS 2001. The Internet Society, 2001.

[Hes04] Florian Hess. A note on the Tate pairing of curves over finite fields. *Archiv der Mathematik*, 82(1):28–32, 2004.

[Hes08] Florian Hess. Pairing lattices. Cryptology ePrint Archive, Report 2008/125, 2008.

[HL02] Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 466–481. Springer-Verlag, 2002.

[HLC08] Yong Ho Hwang, Joseph K. Liu, and Sherman S.M. Chow. Certificateless public key encryption secure against malicious KGC attacks in the standard model. *Journal of Universal Computer Science*, 14(3):463–480, 2008.

[HMV03] Darrel Hankerson, Alfred Menezes, and Scott Vanstone. *Guide to Elliptic Curve Cryptography*. Springer-Verlag, 2003.

[How96] Everett W. Howe. The Weil pairing and the Hilbert symbol. *Mathematische Annalen*, 305(2):387–392, 1996.

[HPS02] Keith Harrison, Dan Page, and Nigel P. Smart. Software implementation of finite fields of characteristic three, for use in pairing-based cryptosystems. *LMS Journal of Computation and Mathematics*, 5:181–193, 2002.

[HS02] Dani Halevy and Adi Shamir. The LSD broadcast encryption scheme. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 47–60. Springer-Verlag, 2002.

[HSV06] Florian Hess, Nigel P. Smart, and Frederik Vercauteren. The eta pairing revisited. *IEEE Transactions on Information Theory*, 52(10):4595–4602, 2006.

[HW07a] Qiong Huang and Duncan S. Wong. Generic certificateless encryption in the standard model. In Atsuko Miyaji, Hiroaki Kikuchi, and Kai Rannenberg, editors, *Advances in Information and Computer Security (IWSEC 2007)*, volume 4752 of *Lecture Notes in Computer Science*, pages 278–291. Springer-Verlag, 2007.

[HW07b] Qiong Huang and Duncan S. Wong. Generic certificateless key encapsulation mechanism. In Josef Pieprzyk, Hossein Ghodosi, and Ed Dawson, editors, *Information Security and Privacy (ACISP 2007)*, volume 4586 of *Lecture Notes in Computer Science*, pages 215–229. Springer-Verlag, 2007.

[HWZD07] Bessie C. Hu, Duncan S. Wong, Zhenfeng Zhang, and Xiaotie Deng. Certificateless signature: A new security model and an improved generic construction. *Designs, Codes and Cryptography*, 42(2):109–126, 2007.

[Int01] Intel Corporation. *Intel Pentium 4 and Intel Xeon Processor Optimization Reference Manual*, 2001. Number 248966-04, `http://developer.intel.com`.

[ISN87] Mitsuru Ito, Akira Saito, and Takao Nishizeki. Secret sharing schemes realizing general access structure. In *IEEE Global Telecommunication Conf. (Globecom '87)*, pages 99–102, 1987.

[IT88] Toshiya Itoh and Shigeo Tsujii. A fast algorithm for computing multiplicative inverses in $\mathrm{GF}(2^m)$ using normal bases. *Information and Computation*, 78(3):171–177, 1988.

[Jia07] Junjie Jiang. Bilinear Pairing (Eta_T Pairing) IP Core. Data sheet, City University of Hong Kong, Available from `http://www.cs.cityu.edu.hk/~ecc/doc/etat_datasheet_v2.pdf`, 2007.

[Jou00] Antoine Joux. A one round protocol for tripartite Diffe-Hellman. In *Algorithmic Number Theory Symposium (ANTS-IV)*, volume 1838 of *Lecture Notes in Computer Science*, pages 385–394. Springer-Verlag, 2000.

[Jou04] Antoine Joux. A one round protocol for tripartite Diffie-Hellman. *Journal of Cryptology*, 17(4):263–276, 2004.

[KAK96] Çetin Kaya Koç, Toga Acar, and Burton S. Kaliski Jr. Analyzing and comparing Montgomery multiplication algorithms. *IEEE Micro*, 16(3):26–33, 1996.

[Kal95] Burton S. Kaliski Jr. The Montgomery inverse and its applications. *IEEE Transactions on Computers*, 44(8):1064–1065, 1995.

[Kat02] Jonathan Katz. A forward-secure public-key encryption scheme. Cryptology ePrint Archive, Report 2002/060, 2002.

[KG06] Eike Kiltz and David Galindo. Direct chosen-ciphertext secure identity-based key encapsulation without random oracles. In Lynn Margaret Batten and Reihaneh Safavi-Naini, editors, *Information Security and Privacy (ACISP 2006)*, volume 4058 of *Lecture Notes in Computer Science*, pages 336–347. Springer-Verlag, 2006.

[KHL03] Chong Hee Kim, Yong Ho Hwang, and Pil Joong Lee. An efficient public key trace and revoke scheme secure against adaptive chosen ciphertext attack. In Chi-Sung Laih, editor, *Advances in Cryptology – ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 359–373. Springer-Verlag, 2003.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer-Verlag, 1999.

[KKS97] Gregory Kabatianskii, E. Krouk, and Ben J. M. Smeets. A digital signature scheme based on random error-correcting codes. In Michael Darnell, editor, *Cryptography and Coding*, volume 1355 of *Lecture Notes in Computer Science*, pages 161–167. Springer-Verlag, 1997.

[Kle07] Thorsten Kleinjung. Discrete logarithms in $\mathrm{GF}(p)$ – 160 digits. Posted on the Number Theory List, February 2007.

[KM07] Neal Koblitz and Alfred Menezes. Another look at generic groups. *Advances in Mathematics of Communications*, 1(1):13–28, 2007.

[KMPB05] Tim Kerins, William P. Marnane, Emmanuel M. Popovici, and Paulo S. L. M. Barreto. Hardware accelerators for pairing based cryptosystems. *IEE Proceedings on Information Security*, 152(1):47–56, 2005.

[KMPR05] Eike Kiltz, Anton Mityagin, Saurabh Panjwani, and Barath Raghavan. Append-only signatures. In Luís Caires et al., editors, *Automata, Languages and Programming (ICALP 2005)*, volume 3580 of *Lecture Notes in Computer Science*, pages 434–445. Springer-Verlag, 2005.

[Koc96] Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer-Verlag, 1996.

[KPF01] Myong H. Kang, Joon S. Park, and Judith N. Froscher. Access control mechanisms for inter-organizational workflow. In *6th ACM Symposium on Access Control Models and Technologies (SACMAT 2001)*, pages 66–74. ACM Press, 2001.

[KRMM06] Maurice Keller, Robert Ronan, William P. Marnane, and Colin Murphy. A $\mathrm{GF}(2^{4m})$ inverter and its application in a reconfigurable Tate pairing processor. In *2006 IEEE International Conference on Reconfigurable Computing and FPGA's (ReConFig 2006)*, pages 158–167. IEEE Press, 2006.

[KRMM07] Maurice Keller, Robert Ronan, William P. Marnane, and Colin Murphy. Hardware architectures for the Tate pairing over $\mathrm{GF}(2^m)$. *Journal of Computers & Electrical Engineering*, 33(5/6):392–406, 2007.

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *Advances in Cryptology – EUROCRYPT 2008*, Lecture Notes in Computer Science, pages 146–162. Springer-Verlag, 2008.

[KTH$^+$06] Tae-Hyun Kim, Tsuyoshi Takagi, Dong-Guk Han, Ho Won Kim, and Jongin Lim. Side channel attacks and countermeasures on pairing based cryptosystems over binary fields. In David Pointcheval, Yi Mu, and Kefei Chen, editors, *Cryptology and Network Security (CANS 2006)*, volume 4301 of *Lecture Notes in Computer Science*, pages 168–181. Springer-Verlag, 2006.

[KV08] Eike Kiltz and Yevgeniy Vahlis. CCA2 secure IBE: Standard model efficiency through authenticated symmetric encryption. In Tal Malkin, editor, *Topics in Cryptology – CT-RSA 2008*, volume 4964 of *Lecture Notes in Computer Science*, pages 239–255. Springer-Verlag, 2008.

[KW93] Mauricio Karchmer and Avi Wigderson. On span programs. In *8th Annual Conference on Structure in Complexity Theory*, pages 102–111. IEEE Computer Society Press, 1993.

[KW03] Jonathan Katz and Nan Wang. Efficiency improvements for signature schemes with tight security reductions. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *10th ACM Conference on Computer and Communications Security (CCS 2003)*, pages 155–164. ACM Press, 2003.

[LAS07] Joseph K. Liu, Man Ho Au, and Willy Susilo. Self-generated-certificate public key cryptography and certificateless signature / encryption scheme in the standard model. In Feng Bao and Steven Miller, editors, *2nd ACM Symposium on Information, Computer and Communications Security (ASIACCS 2007)*, pages 273–283. ACM Press, 2007.

[LD00] Julio López and Ricardo Dahab. High-speed software multiplication in F_{2^m}. In Bimal K.

Roy and Eiji Okamoto, editors, *Progress in Cryptology – INDOCRYPT 2000*, volume 1977 of *Lecture Notes in Computer Science*, pages 203–212. Springer-Verlag, 2000.

[Len87] Hendrick W. Lenstra, Jr. Factoring integers with elliptic curves. *Annals of Mathematics*, 126:649–673, 1987.

[Len01] Arjen K. Lenstra. Unbelievable security. Matching AES security using public key systems (invited talk). In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 67–86. Springer-Verlag, 2001.

[Lic69] Stephen Lichtenbaum. Duality theorems for curves over p-adic fields. *Inventiones Mathematicae*, 7:120–136, 1969.

[LK07] Junzuo Lai and Weidong Kou. Self-generated-certificate public key encryption without pairing. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography – PKC 2007*, volume 4450 of *Lecture Notes in Computer Science*, pages 476–489. Springer-Verlag, 2007.

[LL93] Arjen K. Lenstra and Hendrick W. Lenstra, Jr., editors. *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*. Springer-Verlag, 1993.

[LLP08] Eunjeong Lee, Hyang-Sook Lee, and Cheol-Min Park. Efficient and generalized pairing computation on abelian varieties. Cryptology ePrint Archive, Report 2008/040, 2008.

[LLW05] Jiangtao Li, Ninghui Li, and William H. Winsborough. Automated trust negotiation using cryptographic credentials. In Vijayalakshmi Atluri, Catherine Meadows, and Ari Juels, editors, *12th ACM Conference on Computer and Communications Security (CCS 2005)*, pages 46–57. ACM Press, 2005.

[LOS$^+$06] Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. Sequential aggregate signatures and multisignatures without random oracles. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 465–485. Springer-Verlag, 2006.

[LQ03] Benoît Libert and Jean-Jacques Quisquater. Efficient revocation and threshold pairing based cryptosystems. In *22nd ACM Symposium Annual on Principles of Distributed Computing*, pages 163–171. ACM Press, 2003.

[LQ06] Benoît Libert and Jean-Jacques Quisquater. On constructing certificateless cryptosystems from identity based encryption. In Moti Yung et al., editors, *Public Key Cryptography – PKC 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 474–490. Springer-Verlag, 2006.

[LR88] Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 17(2):373–386, 1988.

[LS06] Florian Luca and Igor E. Shparlinski. Elliptic curves with low embedding degree. *Journal of Cryptology*, 19(4):553–562, 2006.

[Mas91] E. D. Mastrovito. *VLSI Architectures for Computation in Galois Fields*. PhD thesis, Linköping University, 1991.

[May04] Alexander May. Computing the RSA secret key is deterministic polynomial time equivalent to factoring. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 213–219. Springer-Verlag, 2004.

[MBG$^+$93] Alfred J. Menezes, Ian F. Blake, Shuhong Gao, Ronald C. Mullin, Scott A. Vanstone, and Tomik Yaghoobian. *Applications of Finite Fields*. Kluwer Academic Publishers, 1993.

[McE78] Robert J. McEliece. A public-key cryptosystem based on algebraic coding theory. *JPL DSN Progress Report*, pages 114–116, 1978.

[Mes00] Thomas Messerges. Using second order power analysis to attack DPA resistant software. In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2000*, volume 1965 of *Lecture Notes in Computer Science*, pages 238–252. Springer-Verlag, 2000.

[Mil86] Victor S. Miller. Short programs for functions on curves. IBM, Thomas J. Watson Research Center, 1986. Available at `http://crypto.stanford.edu/miller/`.

[Mil04] Victor S. Miller. The Weil pairing, and its efficient calculation. *Journal of Cryptology*, 17(4):235–261, 2004.

[MK90] Yasuyuki Murakami and Masao Kasahara. An ID-based key distribution system. Technical Report ISEC90-42, IEICE, 1990.

[MK05] Yasuyuki Murakami and Masao Kasahara. Murakami-Kasahara ID-based key sharing scheme revisited — in comparison with Maurer-Yacobi schemes. Cryptology ePrint Archive, Report 2005/306, 2005.

[MKHO07] Seiichi Matsuda, Naoki Kanayama, Florian Hess, and Eiji Okamoto. Optimised versions of the ate and twisted ate pairings. In Steven D. Galbraith, editor, *Cryptography and Coding*, volume 4887 of *Lecture Notes in Computer Science*, pages 302–312. Springer-Verlag, 2007.

[MMM02] Tal Malkin, Daniele Micciancio, and Sara K. Miner. Efficient generic forward-secure signatures with an unbounded number of time periods. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 400–417. Springer-Verlag, 2002.

[MNT01] Atsuko Miyaji, Masaki Nakabayashi, and Shunzo Takano. New explicit conditions of elliptic curve traces for FR-reduction. *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, E84-A(5):1234–1243, 2001.

[Mon85] Peter L. Montgomery. Modular multiplication without trial division. *Mathematics of Computation*, 44(170):519–521, 1985.

[MOP07] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer-Verlag, 2007.

[MP02] Patrick Drew McDaniel and Atul Prakash. Methods and limitations of security policy reconciliation. In *2002 IEEE Symposium on Security and Privacy*, pages 73–87. IEEE Computer Society Press, 2002.

[MPS07] Andrew Moss, Daniel Page, and Nigel P. Smart. Toward acceleration of RSA using 3D graphics hardware. In Steven D. Galbraith, editor, *Cryptography and Coding*, volume 4887 of *Lecture Notes in Computer Science*, pages 364–383. Springer-Verlag, 2007.

[MR02] Douglas C. Montgomery and George C. Runger. *Applied Statistic and Probability for Engineers*. John Wiley & Sons, 2002.

[MS77] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error Correcting Codes*. North-Holland, 1977.

[MSK02] Shigeo Mitsunari, Ryuichi Saka, and Masao Kasahara. A new traitor tracing. *IEICE Transactions*, E85-A(2):481–484, 2002.

[MTI86] Tsutomu Matsumoto, Youichi Takashima, and Hideki Imai. On seeking smart public-key-distribution systems. *The Transactions of the IECE of Japan*, E69:99–106, 1986.

[MvV97] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.

[MY91] Ueli M. Maurer and Yacov Yacobi. Non-interactive public-key cryptography. In Donald W. Davies, editor, *Advances in Cryptology – EUROCRYPT '91*, volume 547 of *Lecture Notes in Computer Science*, pages 498–507. Springer-Verlag, 1991.

[MY92] Ueli M. Maurer and Yacov Yacobi. A remark on a non-interactive public-key distribution system (rump session). In Rainer A. Rueppel, editor, *Advances in Cryptology – EUROCRYPT '92*, volume 658 of *Lecture Notes in Computer Science*, pages 458–460. Springer-Verlag, 1992.

[Nac05] David Naccache. Secure and practical identity-based encryption. Cryptology ePrint Archive, Report 2005/369, 2005.

[Nec94] V. I. Nechaev. Complexity of a determinate algorithm for the discrete logarithm. *Mathematical Notes*, 55(2):165–172, 1994.

[Net] Cisco Networks. Netflow monitor. `http://netflow.cesnet.cz/n_netflow.php`.

[Nie86] Harald Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15(2):159–166, 1986.

[NNL01] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62. Springer-Verlag, 2001.

[NP00] Moni Naor and Benny Pinkas. Efficient trace and revoke schemes. In Yair Frankel, editor, *Financial Cryptography (FC 2000)*, volume 1962 of *Lecture Notes in Computer Science*, pages 1–20. Springer-Verlag, 2000.

[NSS04] David Naccache, Nigel Smart, and Jacques Stern. Projective coordinates leak. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 257–267. Springer-Verlag, 2004.

[OM07] Elisabeth Oswald and Stefan Mangard. Template attacks on masking - Resistance is futile. In Masayuki Abe, editor, *Topics in Cryptology – CT-RSA 2007*, volume 4377 of *Lecture Notes in Computer Science*, pages 243–256. Springer-Verlag, 2007.

[OP00] Gerardo Orlando and Christof Paar. A high performance reconfigurable elliptic curve processor for GF(2^m). In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2000*, volume 1965 of *Lecture Notes in Computer Science*, pages 41–56. Springer-Verlag, 2000.

[OS90] H. Ong and Claus-Peter Schnorr. Fast signature generation with a Fiat-Shamir-like scheme. In Ivan Damgård, editor, *Advances in Cryptology – EUROCRYPT '90*, volume 473 of *Lecture Notes in Computer Science*, pages 432–440. Springer-Verlag, 1990.

[OSW07] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *14th ACM Conference on Computer and Communications Security (CCS 2007)*, pages 195–203. ACM Press, 2007.

[Paa94] Christof Paar. *Efficient VLSI Architectures for Bit-Parallel Computation in Galois Fields*. PhD thesis, Institute for Experimental Mathematics, University of Essen, 1994.

[Pap94] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley Publishing Company, 1994.

[PCHL07] Jong Hwan Park, Kyu Young Choi, Jung Yeon Hwang, and Dong Hoon Lee. Certificateless public key encryption in the selective-ID security model (without random oracles). In Tsuyoshi Takagi et al., editors, *Pairing-Based Cryptography – Pairing 2007*, volume 4575 of *Lecture Notes in Computer Science*, pages 60–82. Springer-Verlag, 2007.

[PH78] Stephen C. Pohlig and Martin E. Hellman. An improved algorithm for computing logarithms over $\mathrm{GF}(p)$ and its cryptographic significance. *IEEE Transactions on Information Theory*, 24(1):106–110, 1978.

[Pol78] John M. Pollard. Monte Carlo methods for index computation $(\mathrm{mod}\ p)$. *Mathematics of Computation*, 32(143):918–924, 1978.

[PS06] Kenneth G. Paterson and Jacob C. N. Schuldt. Efficient identity-based signatures secure in the standard model. In Lynn Margaret Batten and Reihaneh Safavi-Naini, editors, *Information Security and Privacy (ACISP 2006)*, volume 4058 of *Lecture Notes in Computer Science*, pages 207–222. Springer-Verlag, 2006.

[PSV06] Dan Page, Nigel P. Smart, and Frederik Vercauteren. A comparison of MNT curves and supersingular curves. *Applicable Algebra in Engineering, Communication and Computing*, 17(5):379–392, 2006.

[PV04] Daniel Page and Frederik Vercauteren. Fault and side-channel attacks on pairing based cryptography. Cryptology ePrint Archive, Report 2004/283, 2004.

[PV06] Daniel Page and Frederik Vercauteren. A fault attack on pairing based cryptography. *IEEE Transactions on Computers*, 55(9):1075–1080, 2006.

[RhM$^+$06] Robert Ronan, Colm Ó hÉigeartaigh, Colin Murphy, Michael Scott, and Tim Kerins. FPGA acceleration of the Tate pairing in characteristic 2. In *IEEE International Conference on Field Programmable Technology (FPT 2006)*, pages 213–220. IEEE Press, 2006.

[RhM$^+$07a] Robert Ronan, Colm Ó hÉigeartaigh, Colin Murphy, Tim Kerins, and P. S. L. M. Barreto. A flexible processor for the characteristic 3 η_T pairing. *International Journal of High Performance Systems Architecture*, 1(2):79–88, 2007.

[RhM$^+$07b] Robert Ronan, Colm Ó hÉigeartaigh, Colin Murphy, Michael Scott, and Tim Kerins. Hardware acceleration of the Tate pairing on a genus 2 hyperelliptic curve. *Journal of Systems Architecture*, 53(2/3):85–98, 2007.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd Annual ACM Symposium on Theory of Computing*, pages 387–394. ACM Press, 1990.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[SA02] Sergei P. Skorogogatov and Ross A. Anderson. Optical fault induction attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 2–12. Springer-Verlag, 2002.

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th Annual Symposium on Foundations of Computer Science*, pages 543–553. IEEE Computer Society Press, 1999.

[SB04] Michael Scott and Paulo S. L. M. Barreto. Compressed pairings. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 140–156. Springer-Verlag, 2004.

[SC07] Palash Sarkar and Sanjit Chatterjee. Construction of a hybrid HIBE protocol secure against adaptive attacks. In Willy Susilo, Joseph K. Liu, and Yi Mu, editors, *Provable Security*, volume 4786 of *Lecture Notes in Computer Science*, pages 51–67. Springer-Verlag, 2007.

[SCA06] Michael Scott, Neil Costigan, and Wesam Abdulwahab. Implementing cryptographic pairings on smartcards. In Louis Goubin and Mitsuru Matsui, editors, *Cryptographic Hardware and Embedded Systems – CHES 2006*, volume 4249 of *Lecture Notes in Computer Science*, pages 134–147. Springer-Verlag, 2006.

[SCFY96] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.

[Sch80] Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[Sch98] René Schoof. Counting points on elliptic curves over finite fields. *Journal de Théorie des Nombres de Bordeaux*, 7:483–494, 1998.

[Sco] Michael Scott. Multiprecision Integer and Rational Arithmetic C/C++ Library - MIRACL. URL: `http://www.shamus.ie/`.

[Sco05a] Michael Scott. Computing the Tate pairing. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 293–304. Springer-Verlag, 2005.

[Sco05b] Michael Scott. Faster pairings using an elliptic curve with an efficient endomorphism. In Subhamoy Maitra, C. E. Veni Madhavan, and Ramarathnam Venkatesan, editors, *Progress in Cryptology – INDOCRYPT 2005*, volume 3797 of *Lecture Notes in Computer Science*, pages 258–269. Springer-Verlag, 2005.

[Sco05c] Michael Scott. Scaling security in pairing-based protocols. Cryptology ePrint Archive, Report 2005/139, 2005.

[Sco07a] Michael Scott. Implementing cryptographic pairings. In Tsuyoshi Takagi et al., editors, *Pairing-Based Cryptography – Pairing 2007*, volume 4575 of *Lecture Notes in Computer Science*, pages 177–196. Springer-Verlag, 2007.

[Sco07b] Michael Scott. Optimal irreducible polynomials for $\mathrm{GF}(2^m)$ arithmetic. Cryptology ePrint Archive, Report 2007/192, 2007.

[Sen02] Nicolas Sendrier. On the security of the McEliece public-key cryptosystem. In Mario Blaum, Patrick G. Farrell, and Henk C. A. van Tilborg, editors, *Information Coding and Mathematics*, volume 687 of *Kluwer International Series in Engineering and Computer Science*, pages 141–163. Kluwer, 2002.

[Sha79] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[Sha85] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO '84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer-Verlag, 1985.

[Sha01] Sheueling Chang Shantz. From Euclid's GCD to Montgomery multiplication to the great divide. Technical Report TR-2001-95, Sun Microsystems, 2001.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266. Springer-Verlag, 1997.

[Sho01] Victor Shoup. A proposal for an ISO standard for public key encryption (version 2.1). Cryptology ePrint Archive, Report 2001/112, 2001.

[Sil92] Joseph H. Silverman. *The Arithmetic of Elliptic Curves*, volume 106 of *Graduate Texts in Mathematics*. Springer-Verlag, 1992. Corrected reprint of the 1986 original.

[SK03] Ryuichi Sakai and Masao Kasahara. ID based cryptosystems with pairing on elliptic curve. Cryptology ePrint Archive, Report 2003/054, 2003.

[SKG06] Chang Shu, Soonhak Kwon, and Kris Gaj. FPGA accelerated Tate pairing based cryptosystems over binary fields. In *IEEE International Conference on Field Programmable Technology (FPT 2006)*, pages 173–180. IEEE Press, 2006.

[SL02] Martijn Stam and Arjen K. Lenstra. Efficient subgroup exponentiation in quadratic and sixth degree extensions. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *Cryp-*

tographic Hardware and Embedded Systems – CHES 2002, volume 2523 of *Lecture Notes in Computer Science*, pages 159–174. Springer-Verlag, 2002.

[Sma03] Nigel P. Smart. Access control using pairing based cryptography. In Marc Joye, editor, *Topics in Cryptology – CT-RSA 2003*, volume 2612 of *Lecture Notes in Computer Science*, pages 111–121. Springer-Verlag, 2003.

[SOK00] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *2000 Symposium on Cryptography and Information Security (SCIS 2000)*, Okinawa, January 2000.

[SOS$^+$08] Piotr Szczechowiak, Leonardo B. Oliveira, Michael Scott, Martin Collier, and Ricardo Dahab. NanoECC: Testing the limits of elliptic curve cryptography in sensor networks. In Roberto Verdone, editor, *Wireless Sensor Networks (EWSN 2008)*, volume 4913 of *Lecture Notes in Computer Science*, pages 305–320. Springer-Verlag, 2008.

[SP98] Leilei Song and Keshab K. Parhi. Low energy digit-serial/parallel finite field multipliers. *Journal of VLSI Signal Processing Systems*, 19(2):149–166, 1998.

[SPMLS02] Jacques Stern, David Pointcheval, John Malone-Lee, and Nigel P. Smart. Flaws in applying proof methodologies to signature schemes. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 93–110. Springer-Verlag, 2002.

[Ste94] Jacques Stern. A new identification scheme based on syndrome decoding. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 13–21. Springer-Verlag, 1994.

[STO07] Masaaki Shirase, Tsuyoshi Takagi, and Eiji Okamoto. Some efficient algorithms for the final exponentiation of η_T pairing. In Ed Dawson and Duncan S.Wong, editors, *Information Security Practice and Experience Conference (ISPEC 2007)*, volume 4464 of *Lecture Notes in Computer Science*, pages 254–268. Springer-Verlag, 2007.

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer-Verlag, 2005.

[SZB07] Yinxia Sun, Futai Zhang, and Joonsang Baek. Strongly secure certificateless public key encryption without pairing. In Feng Bao et al., editors, *Cryptology and Network Security (CANS 2007)*, volume 4856 of *Lecture Notes in Computer Science*, pages 194–208. Springer-Verlag, 2007.

[Tan88] Hatsukazu Tanaka. A realization scheme for the identity-based cryptosystem. In Carl Pomerance, editor, *Advances in Cryptology – CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*, pages 340–349. Springer-Verlag, 1988.

[Tat95] John Tate. WC-groups over $\mathfrak{p}$-adic fields. In *Séminaire Bourbaki*, volume 4, pages 265–277. Société Mathématique de France, 1995. Exposé no. 156 (1957/58).

[THK07] Gen Takahashi, Fumitaka Hoshino, and Tetsutaro Kobayashi. Efficient $\mathrm{GF}(3^m)$ multiplication algorithm for ηT pairing. Cryptology ePrint Archive, Report 2007/463, 2007.

[TSP86] Stafford E. Tavares, P. A. Scott, and Lloyd E. Peppard. A fast VLSI multiplier for $\mathrm{GF}(2^m)$. *IEEE Journal Selected Areas in Communications*, 4(1):62–66, 1986.

[TT01] Wen-Guey Tzeng and Zhi-Jia Tzeng. A public-key traitor tracing scheme with revocation using dynamic shares. In Kwangjo Kim, editor, *Public Key Cryptography (PKC 2001)*, volume 1992 of *Lecture Notes in Computer Science*, pages 207–224. Springer-Verlag, 2001.

[TYW04] Roberto Tamassia, Danfeng Yao, and William H. Winsborough. Role-based cascaded delegation. In *9th ACM Symposium on Access Control Models and Technologies (SACMAT 2004)*, pages 146–155. ACM Press, 2004.

[Vér95] Pascal Véron. A fast identification scheme. In *1995 IEEE International Symposium on Information Theory*, page 359. IEEE Press, 1995.

[Ver08a] Frederik Vercauteren. The hidden root problem. In Steven D. Galbraith and Kenny G. Paterson, editors, *Pairing-Based Cryptography – Pairing 2008*, volume 5209 of *Lecture Notes in Computer Science*, pages 89–99. Springer-Verlag, 2008.

[Ver08b] Frederik Vercauteren. Optimal pairings. Cryptology ePrint Archive, Report 2008/096, 2008.

[Wat05] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127. Springer-Verlag, 2005.

[Wat08] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. Cryptology ePrint Archive, Report 2008/290, 2008.

[Wei40] André Weil. Sur les fonctions algébriques à corps de constantes fini. *C. R. Acad. Sci. Paris*, 210:592–594, 1940.

[WG94] David L. Weaver and Tom Germond, editors. *The SPARC Architecture Manual, Version 9*. Prentice Hall, 1994.

[WS06] Claire Whelan and Michael Scott. Side channel analysis of practical pairings: Which path is more secure? In Phong Q. Nguyen, editor, *Progress in Cryptology – VIETCRYPT 2006*, volume 4341 of *Lecture Notes in Computer Science*, pages 99–114. Springer-Verlag, 2006.

[WS07] Claire Whelan and Michael Scott. The importance of the final exponentiation in pairings when considering fault attacks. In Tsuyoshi Takagi et al., editors, *Pairing-Based Cryptography – Pairing 2007*, volume 4575 of *Lecture Notes in Computer Science*, pages 225–246. Springer-Verlag, 2007.

[WW04] Jason Waddle and David Wagner. Towards efficient second-order power analysis. In Marc Joye and Jean-Jacques Quisquater, editors, *Cryptographic Hardware and Embedded Systems – CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 1–15. Springer-Verlag, 2004.

[Xil] *Xilinx Virtex-II FPGA Datasheet*. Available from `http://www.xilinx.com/support/documentation/data_sheets/ds_031.pdf`.

[YCHG07] Wun-She Yap, Sherman S. M. Chow, Swee-Huay Heng, and Bok-Min Goi. Security mediated certificateless signatures. In Jonathan Katz and Moti Yung, editors, *Applied Cryptography and Network Security (ACNS 2007)*, volume 4521 of *Lecture Notes in Computer Science*, pages 459–477. Springer-Verlag, 2007.

[YFDL04] Danfeng Yao, Nelly Fazio, Yevgeniy Dodis, and Anna Lysyanskaya. ID-based encryption for complex hierarchies with applications to forward security and broadcast encryption. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick McDaniel, editors, *11th ACM Conference on Computer and Communications Security (CCS 2004)*, pages 354–363. ACM Press, 2004.

[Yi03] Xun Yi. An identity-based signature scheme from the Weil pairing. *IEEE Communications Letters*, 7(2):76–78, February 2003.

[YW03] Ting Yu and Marianne Winslett. A unified scheme for resource protection in automated trust negotiation. In *2003 IEEE Symposium on Security and Privacy*, pages 110–122. IEEE Computer Society Press, 2003.

[YW05] Tsz Hon Yuen and Victor K. Wei. Fast and proven secure blind identity-based signcryption from pairings. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 305–322. Springer-Verlag, 2005.

[Zim] Paul Zimmerman. Integer factoring records. `http://www.loria.fr/~zimmerma/records/factor.html`.

[ZZ08] Chang-An Zhao and Fangguo Zhang. Reducing the complexity of the Weil pairing computation. Cryptology ePrint Archive, Report 2008/212, 2008.

[ZZH06] Chang-An Zhao, Fangguo Zhang, and Jiwu Huang. Speeding up the bilinear pairings computation on curves with automorphisms. Cryptology ePrint Archive, Report 2006/474, 2006.

[ZZH07] Chang-An Zhao, Fangguo Zhang, and Jiwu Huang. A note on the ate pairing. Cryptology ePrint Archive, Report 2007/247, 2007.

[ZZH08] Chang-An Zhao, Fangguo Zhang, and Jiwu Huang. All pairings are in a group. Cryptology ePrint Archive, Report 2008/085, 2008.

Author Index