KU LEUVEN

FACULTEIT INGENIEURSWETENSCHAPPEN

Practical Identity-Based Encryption for Online Social Networks

Stijn Meul

Thesis submitted for the degree of
Master of Science in
Electrical Engineering, option
Embedded Systems and Multimedia

Thesis supervisors:
Prof. dr. ir. Bart Preneel
Prof. dr. ir. Vincent Rijmen

Assessors:
Prof. dr. ir. Claudia Diaz
Prof. dr. ir. Frank Piessens

Mentor:
Filipe Beato

Academic year 2013 – 2014

Preface

I would like to thank everybody who kept me busy the last year, especially my promotor and my assistants. I would also like to thank the jury for reading the text. My sincere gratitude also goes to my wive and the rest of my family.

Stijn Meul

Contents

Preface i

Abstract iv

List of Figures and Tables v

List of Abbreviations and Symbols vi

1 Introduction 1

1.1 Problem Statement . . . 1
1.2 Previous Work . . . 1
1.3 Goals of this Thesis . . . 1
1.4 Structure of this Thesis . . . 1

2 Background 3

2.1 Negligible Function . . . 3
2.2 Abstract Algebra . . . 4
2.3 Number Theoretic Assumptions . . . 7
2.4 Bilinear Maps . . . 9
2.5 Hash Functions . . . 10
2.6 Cryptology - this section is temporary moved from chapter 3 . . . 12
2.7 Conclusion . . . 16

3 Literature Review 17

3.1 Identity-Based Encryption . . . 17
3.2 Broadcast Encryption . . . 18
3.3 Secret Sharing . . . 18
3.4 Distributed Key Generation . . . 20
3.5 Conclusion . . . 20

4 Outsider Anonymous Identity-Based Broadcast Encryption 21

4.1 Security Model . . . 21
4.2 Proposed Scheme . . . 21
4.3 Conclusion . . . 21

5 Implementation 23

5.1 Existing Solutions . . . 23

5.2 Anonymous Identity-Based Broadcasting Implementation 23
5.3 Distributed Key Generation Implementation 23
5.4 Evaluation 23
5.5 Performance Analysis 23
5.6 Conclusion 23

6 Conclusion 25

A Installing and Executing the Code 29
A.1 Setting up the DKG 29
A.2 Setting up Scramble 29

B The Last Appendix 31
B.1 Lorem 20-24 31
B.2 Lorem 25-27 31

Bibliography 33

Abstract

The `abstract` environment contains a more extensive overview of the work. But it should be limited to one page.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

List of Figures and Tables

List of Figures

2.1 A cryptosystem [2] 12

3.1 Multiple (n,t)-PKG IBE for OSNs overview, for a message m published for the set $\mathcal{S}$ for $t = 3$. 18

List of Tables

List of Abbreviations and Symbols

Abbreviations

LoG	Laplacian-of-Gaussian
MSE	Mean Square error
PSNR	Peak Signal-to-Noise ratio

Symbols

G	A group $(G, *)$
$S_A(m)$	Signature of entity A on message m
$e : G_1 \times G_2 \to G_T$	Admissible bilinear map
$e(P, Q)$	Admissible bilinear map $P \in G_1, Q \in G_2, e(P, Q) \in G_T$

Chapter 1

Introduction

The newest internet trend at the dawn of the 21st century certainly is the Online Social Network (OSN). Words like tweeting, sharing, liking, trending and tagging have found common acceptance in the vocabulary of today's internet savvy users while services like Facebook, Google+, LinkedIn and Twitter have become part of everyday life.

The far reaching influence of today's most popular OSNs is best illustrated with the help of some statistics. In May 2013, 72% of all internet users were active on a social network [26]. At the time of writing, Facebook has 1.23 Billion monthly active users which corresponds to 17% of the global population [13, 38]. Furthermore, the average Facebook user spends 15 hours and 33 minutes online per month [36]. These numbers show that social networks no longer represent the latest craze of an internet bubble but are conversely deeply rooted in our daily habits.

1.1 Problem Statement

1.2 Previous Work

1.3 Goals of this Thesis

1.4 Structure of this Thesis

Chapter

2 Background

This chapter covers briefly the mathematical knowledge required to understand cryptographic algorithms presented later in this text. Although understanding all mathematical details of this chapter can be quite a struggle, the math serves as a fundament of a challenging world containing exciting cryptographic concepts like identity-based encryption.

First, the notion of negligible functions will be introduced followed by an overview of algebraic structures and their properties. Then, a number of theoretic assumptions fundamental for cryptographic security are presented. By exploring these variants of the Diffie-Hellman assumption, the introduction of gap groups and bilinear maps follows naturally. Finally, hash functions are defined as well as their relation to the random oracle assumption.

Note that this chapter only scratches the surface of cryptographic fundamentals required to understand the remainder of the thesis. Definitions and theorems are always provided without proof. For a more in depth discussion about algebraic topics in this chapter, the reader is referred to [31] and [9]. More information on elliptic curves, Diffie-Hellman assumptions and pairing based cryptography can be found in [1].

If the reader feels he has sufficient background of the concepts covered in this chapter, the chapter can be skipped without loss of comprehension.

2.1 Negligible Function

In practice no modern cryptographic algorithm achieves perfect secrecy[1], i.e. with unbounded computational power all practical cryptographic algorithms can be broken. Therefore a more pragmatic definition of security is always considered, namely security against adversaries that are computationally bound to their finite resources. In this pragmatic view of security an algorithm is considered secure only if the probability of success is smaller than the reciprocal of any polynomial function. The negligible function can be used to exactly describe this notion in a formal way.

[1]Note that the one-time pad is not taken into account. Although it is the only proven information secure cryptographic algorithm, it is seldom used in practical cryptographic systems.

Definition 2.1. A **negligible function** in k is a function $\mu(k) : \mathbb{N} \to \mathbb{R}$ if for every polynomial $p(.)$ there exists an N such that for all $k > N$ [23]

$$\mu(k) < \frac{1}{p(k)}$$

The negligible function will be used later on in this chapter to formally describe computationally infeasible problems. In such a context k often represents the security parameter. The larger k will be chosen, the smaller $\mu(k)$ will be.

2.2 Abstract Algebra

Abstract algebra is a field of mathematics that studies algebraic structures such as groups, rings and vector spaces. These algebraic structures define a collection of requirements on mathematical sets such as e.g., the natural numbers $\mathbb{N}$ or matrices of dimension 2 x 2 $\mathbb{R}^{2x2}$. If these requirements hold, abstract properties can be derived. Once a mathematical set is then categorised as the correct algebraic structure, properties derived for the algebraic structure will hold for the set as a whole.

In the light of our further discussion, especially additive and multiplicative groups prove to be essential concepts. However, algebraic groups come with a specific vocabulary such as binary operation, group order and cyclic group that are defined in this section as well.

Definition 2.2 (Binary operation). A *binary operation* * on a set S is a mapping $S \times S \to S$. That is, a binary operation is a rule which assigns to each ordered pair of elements a and b from S a uniquely defined third element $c = a * b$ in the same set S. [31, 9]

Definition 2.3 (Group). A *group* $(G, *)$ consists of a set G with a binary operation $*$ on G satisfying the following three axioms:

1. *Associativity* $\forall a, b, c \in G : a * (b * c) = (a * b) * c$
2. *Identity element* $\forall a \in G, \exists e \in G : a * e = e * a = a$ where e denotes the *identity element* of G
3. *Inverse element* $\forall a \in G, \exists a^{-1} : a * a^{-1} = a^{-1} * a = 1$ where a^{-1} denotes the *inverse element* of a

Definition 2.4 (Commutative group). A group $(G, *)$ is called a *commutative group* or an *abelian group* if in addition to the properties in Definition 2.3, also commutativity holds.

4. **Commutativity** $\forall a, b \in G : a * b = b * a$

Depending on the group operation $*$, $(G, *)$ is called either a *multiplicative group* or an *additive group*. In Definition 2.3 the multiplicative notation is used. For an additive group the inverse of a is often denoted $-a$ [31].

A group $(G, *)$ is often denoted by the more concise symbol G although groups are always defined with respect to a binary group operation $*$. Despite of a more concise notation, any group G still obeys all axioms from Definition 2.3 with respect to an implicitly known group operation $*$.

A perfect example of a commutative group is the set of integers with the addition operation $(\mathbb{Z}, +)$ since the addition is both associative and commutative in $\mathbb{Z}$. Furthermore, the identity element $e = 0$ and the inverse element $\forall a \in \mathbb{Z}$ is $-a \in \mathbb{Z}$. Note that the set of natural numbers with the addition operation $(\mathbb{N}, +)$ is not a commutative group as not every element of $\mathbb{N}$ has an inverse element.

Definition 2.5 (Cyclic group). A group G is *cyclic* if and only if $\forall b \in G, \exists g \in G, \exists n \in \mathbb{Z} : g^n = b$. Such an element g is called a **generator** of $\mathbb{G}$.

Definition 2.5 implies that in a cyclic group every element can be written as a power of one of the group's generators.

Definition 2.6 (Finite group). A group G is *finite* if the number of elements in G denoted $|G|$ is finite. The number of elements $|G|$ in a finite group is called the *group order*.

The set $\mathbb{Z}_n$ denotes the set of integers modulo n. The set $\mathbb{Z}_5$ with the addition operation is a cyclic finite group of order 5. The set $\mathbb{Z}_5\backslash\{0\}$ with the multiplication operation, often denoted $\mathbb{Z}_5^*$, is a cyclic finite group of order 4 where the neutral element $e = 1$. Two is an example of a generator in $\mathbb{Z}_5^*$ since every element in $\mathbb{Z}_5^*$ can be written as $\{2^n | n \in \mathbb{Z}\}$.

Definition 2.7 (Order of an element). Let G be a group. The *order of an element* $a \in G$ is defined as the least positive integer t such that $a^t = e$. If there exists no such t, t is defined as ∞.

Theorem 2.8. *If the order of a group G equals a prime p, the group is cyclic and commutative.*

Definition 2.9 (Subgroup). Given a group $(G, *)$, any H that is a non-empty subset $H \subseteq G$ and satisfies the axioms of a group with respect to the group operation $*$ in H, is a *subgroup of G*.

Definition 2.10 (Ring). A *ring* $(R, +, *)$ consists of a set R with two binary operations $+$ and $*$ on R satisfying the following axioms:

1. $(R, +)$ is an abelian group with identity denoted e
2. *Associativity* $\forall a, b, c \in R : a * (b * c) = (a * b) * c$
3. *Multiplicative identity element* $\forall a \in R, \exists 1 \in R : a * 1 = 1 * a = a$ where 1 denotes the *multiplicative identity element* of R
4. *Left distributivity* $\forall a, b, c \in R : a * (b + c) = (a * b) + (a * c)$

5. *Right distributivity* $\forall a, b, c \in R : (b + c) * a = (b * a) + (c * a)$

Definition 2.11 (Commutative ring). A ring $(R, +, *)$ is called a *commutative ring* or an *abelian ring* if in addition to the properties in Definition 2.10, also commutativity holds.

6. **Commutativity** $\forall a, b \in R : a * b = b * a$

Definition 2.12 (Field). A commutative ring $(R, +, *)$is called a *field* if in addition to the properties in Definition 2.11 and Definition 2.10 all elements of R have a multiplicative inverse.

7. *Multiplicative inverse* $\forall a \in R, \exists a^{-1} : a * a^{-1} = a^{-1} * a = 1$ where a^{-1} denotes the *inverse element* of a

Definition 2.13 (Finite field). A *finite field* or a *Galois Field* is a field F with a finite number of elements. The number of elements $|F|$ of a finite field F is called its *order*.

Definition 2.14 (Ring homomorphism). Given rings R and S, a *ring homomorphism* is a function $f : R \to S$ such that the following axioms hold:

1. $\forall a, b \in R : f(a + b) = f(a) + f(b)$
2. $\forall a, b \in R : f(ab) = f(a) f(b)$
3. $f(e_R) = f(e_S)$ where e_S and e_R denote the identity element of respectively S and R

Definition 2.15 (Bijective function). Any function $f : R \to S$ is bijective if it satisfies the following axioms

1. *Injective* Each element in S is the image of at most one element in R. Hence, $\forall a_1, a_2 \in R$ if $(a_1) = f(a_2)$ then $a_1 = a_2$ follows naturally.
2. *Surjective* Each $s \in S$ is the image of at least one $r \in R$.

Definition 2.16 (Ring isomorphism). A ring isomorphism is a bijective homomorphism.

Informally speaking, a ring isomorphism $f : R \to S$ is a mapping between rings that are structurally the same such that any element of R has exactly one image in S.

Note that $(\mathbb{Z}_n, +, \cdot)$ is a finite field if and only if n is a prime number. Furthermore, if F is a finite field, then F contains p^m elements for some prime p and integer $m \geq 1$. For every prime power order p^m, there is a unique finite field of order p^m. This field is denoted by $\mathbb{F}_{p^m}$ or $GF(p^m)$. The finite field $\mathbb{F}_{p^m}$ is unique up to an isomorphism.

2.3 Number Theoretic Assumptions

This section presents a collection of number theoretic assumptions. The security of our future constructions falls or stands on these assumptions. If one of these assumptions would prove to be invalid, not only this thesis would be superfluous, society would no longer be protected by widely adopted cryptographic protocols like RSA or ElGamal encryption [11, 31].

In the definitions that follow $\langle G, n, g\rangle \leftarrow \mathcal{G}\left(1^k\right)$ is defined as the setup algorithm that generates a group G of order n and a generator $g \in G$ on input of the security parameter k.

Definition 2.17 (DL). The *discrete logarithm problem* is defined as follows. Given a finite cyclic group G of order n, a generator $g \in G$ and an element $a \in G$, find the integer $x, 0 \leq x \leq n-1$ such that $g^x = a$.

The *discrete logarithm assumption* holds if for any algorithm $\mathcal{A}(g, g^x)$ trying to solve the DL problem there exists a negligible function $\mu(k)$ such that

$$\Pr\left[\mathcal{A}(g, g^x) = a \mid \langle G, n, g\rangle \leftarrow \mathcal{G}\left(1^k\right)\right] \leq \mu(k)$$

where the probability is over the random choice of n, g in G according to the distribution induced by $\mathcal{G}\left(1^k\right)$, the random choice of a in G and the random bits of the algorithm $\mathcal{A}$.

Definition 2.18 (CDH). The *Computational Diffie-Hellman problem* is defined as follows. Given a finite cyclic group G of order n, a generator $g \in G$ and g^a, g^b with uniformly chosen random independent elements $a, b \in \{1, \ldots, |G|\}$, find the value g^{ab}.

The *Computational Diffie-Hellman assumption* holds if for any algorithm $\mathcal{A}\left(g, g^a, g^b\right)$ trying to solve the CDH problem there exists a negligible function $\mu(k)$ such that

$$\Pr\left[\mathcal{A}\left(g, g^a, g^b\right) = g^{ab} \mid \langle G, n, g\rangle \leftarrow \mathcal{G}\left(1^k\right)\right] \leq \mu(k)$$

where the probability is over the random choice of n, g in G according to the distribution induced by $\mathcal{G}\left(1^k\right)$, the random choice of a, b in $\{1, \ldots, |G|\}$ and the random bits of the algorithm $\mathcal{A}$.

Definition 2.19 (DDH). The *Decisional Diffie-Hellman problem* is defined as follows. Given a finite cyclic group G of order n, a generator $g \in G$ and g^a, g^b, g^{ab}, g^c with uniformly chosen random independent elements $a, b, c \in \{1, \ldots, |G|\}$, distinguish $\left\langle g, g^a, g^b, g^{ab}\right\rangle$ from $\left\langle g, g^a, g^b, g^c\right\rangle$.

Define $\mathcal{A}(x)$ as an algorithm returning `true` if $x = \left\langle g, g^a, g^b, g^{ab}\right\rangle$ and `false` if $x = \left\langle g, g^a, g^b, g^c\right\rangle$ for $c \neq ab$. The *Decisional Diffie-Hellman assumption* holds if for any such algorithm $\mathcal{A}(x)$ there exists a negligible function $\mu(k)$ such that

$$|\Pr\left[\mathcal{A}\left(\left\langle g, g^a, g^b, g^{ab}\right\rangle\right) = \texttt{true}\right] - \Pr\left[\mathcal{A}\left(\left\langle g, g^a, g^b, g^c\right\rangle\right) = \texttt{true}\right]| \leq \mu(k)$$

where the probability is over the random choice of n, g in G according to the distribution induced by $\mathcal{G}\left(1^k\right)$, the random choice of a, b, c in $\{1, \ldots, |G|\}$ and the random bits of the algorithm $\mathcal{A}$.

Definition 2.19 states that $\left\langle g, g^a, g^b, g^{ab}\right\rangle$ and $\left\langle g, g^a, g^b, g^c\right\rangle$ are *computationally indistinguishable*. It means that no efficient algorithm exists that can distinguish both arguments with non-negligible probability. The concept of computational indistinguishable arguments bears close resemblance to statistically indistinguishable ensembles. The reader is referred to [24] and [25] for a more in depth discussion of the topic. The intuitive interpretation of Definition 2.19 is that g^{ab} looks like any other random element in G.

Someone with the ability to calculate discrete logarithms could trivially solve the CDH problem. That is, if a and b can be derived only from $\left\langle g^a, g^b\right\rangle$, it becomes easy to calculate g^{ab}. Therefore, a group structure where the CDH assumption holds, immediately implies a group where the DL assumption is valid as well. There is no mathematical proof that supports the inverse relation. Thus, a group where the DL problem is hard not necessarily implies the CDH problem. For specific group structures [29] and [30] show that CDH immediately follows from the DL assumption, however, their proof can not be generalised to just any group.

There exists a similar relation between the CDH and the DDH problem. If a powerful algorithm could solve CDH, i.e. derive g^{ab} from $\left\langle g, g^a, g^b\right\rangle$ alone, it would become trivial to distinguish $\left\langle g, g^a, g^b, g^{ab}\right\rangle$ from $\left\langle g, g^a, g^b, g^c\right\rangle$. Again, an inverse relation can not be proven. As a matter of fact, concrete examples of groups exist where CDH is hard although DDH is not.

Therefore, the relation between DL, CDH and DDH is often written as follows

$$DDH \Rightarrow CDH \Rightarrow DL$$

The $\Rightarrow$ notation is then translated into "immediately implies". In a group where DDH is hard both CDH and DL will be hard. On the contrary, there exist group structures where the CDH and the DL assumption hold while DDH can be found easily. Such groups are called *Gap Diffie-Hellman Groups*.

Definition 2.20 (GDH). The *Gap Diffie-Hellman problem* is defined as follows. Solve the CDH problem with the help of a DDH oracle. Given a finite cyclic group G of order n, a generator $g \in G$ and g^a, g^b with uniformly chosen random independent elements $a, b \in \{1, \ldots, |G|\}$, find the value g^{ab} with the help of a DDH oracle $\mathcal{DDH}\left(g, g^a, g^b, z\right)$. Where the DDH oracle $\mathcal{DDH}\left(g, g^a, g^b, z\right)$ is defined to return `true` if $z = g^{ab}$ and `false` if $z \neq g^{ab}$.

The *Gap Diffie-Hellman assumption* holds if for any algorithm $\mathcal{A}\left(g, g^a, g^b\right)$ trying to solve the CDH problem with the help of a DDH oracle $\mathcal{DDH}\left(g, g^a, g^b, z\right)$ there exists a negligible function $\mu(k)$ such that

$$\Pr\left[\mathcal{A}\left(g, g^a, g^b\right) = g^{ab} \mid \langle G, n, g\rangle \leftarrow \mathcal{G}\left(1^k\right)\right] \leq \mu(k)$$

where the probability is over the random choice of n, g in G according to the distribution induced by $\mathcal{G}\left(1^k\right)$, the random choice of a, b in $\{1, \ldots, |G|\}$ and the random bits of the algorithm $\mathcal{A}$.

As discussed in the next Section 2.4 bilinear pairings are an example of a practical usable DDH oracle [28].

2.4 Bilinear Maps

2.4.1 Definition

Definition 2.21 (Admissible bilinear map). Let G_1, G_2 and G_T be three groups of order q for some large prime q. An *admissible bilinear map* $e : G_1 \times G_2 \to G_T$ is defined as a map from the gap groups G_1 and G_2 to the target group G_T that satisfies the following properties:

1. *Bilinearity* $\forall a, b \in \mathbb{Z}, \forall P \in G_1, \forall Q \in G_2 : e\left(aP, bQ\right) = e\left(P, Q\right)^{ab}$
2. *Non-degeneracy* If P is a generator of G_1 and Q is a generator of G_2, $e\left(P, Q\right)$ is a generator of G_T
3. *Computability* There is an efficient algorithm to compute $e\left(P, Q\right)$ for all $P \in G_1$ and $Q \in G_2$

In literature, authors distinguish two types of admissible bilinear maps. A *symmetric bilinear map* is an admissible bilinear map where the gap groups are the same, i.e. $G_1 = G_2$. Definition 2.21 describes the more general *asymmetric bilinear map* where $G_1 \neq G_2$. Schemes relying on symmetric bilinear maps are easier to construct information theoretic security proofs although asymmetric bilinear maps are more efficient and suitable for implementation thanks to their flexible embedding degree [12, 39].

In practice, bilinear maps are constructed using pairings. The most popular pairings implementing admissible bilinear maps are the Weil pairing [12] and the Tate pairing [22]. Both the Tate and the Weil pairing rely on abelian varieties for their implementation. G_1 is mostly an additive elliptic curve group, G_2 a multiplicative elliptic curve group while G_T is a finite field. For instance, the asymmetric Weil pairing is often implemented with a cyclic subgroup of $E\left(\mathbb{F}_p\right)$ of order q for G_2 and a different cyclic subgroup of $E\left(\mathbb{F}_{p^6}\right)$ of the same order q for G_1 where $E\left(\mathbb{F}_{p^6}\right)$ denotes the group of points on an elliptic curve E over the finite field $\mathbb{F}_{p^6}$. The interested reader is referred to [1] for more information concerning elliptic curves and their use in pairing based cryptography. Details on Elliptic Curve Cryptography fall out of the scope of this thesis as it suffices to make abstraction of these concepts for the remainder of the text.

Research [5, 27, 3] has recently shown that the discrete logarithm problem is easier in the symmetric setting because symmetric pairings rely on more structured supersingular (hyper)elliptic curves. Therefore, care should be taken when using symmetric pairings [39].

2.4.2 Bilinear Diffie-Hellman Assumption

It is not a coincidence that G_1 and G_2 are called gap groups. A bilinear map allows to solve the Decisional Diffie-Hellman problem in G_1 and G_2. The DDH problem in G_1 consists of distinguishing $\langle P, aP, bP, abP\rangle$ from $\langle P, aP, bP, cP\rangle$ where $P \in G_1$, P is a generator of G_1 and a, b, c randomly chosen in $\{1, \ldots, |G_1|\}$. Given a symmetric bilinear map $e : G_1 \times G_1 \to G_T$ a solution to this problem can be found by relying on the bilinearity of the pairing as follows:

$$e\left(aP, bP\right) = e\left(P, P\right)^{ab} \overset{?}{=} e\left(P, cP\right) = e\left(P, P\right)^{c}$$

Such that the second equality will hold only if $ab = c$. A similar statement can be made concerning G_2 with the help of the map $e : G_2 \times G_2 \to G_T$. Consequently G_1 and G_2 are both GDH groups. From Section 2.3 it follows that CDH can still be hard in GDH groups because DDH is a stronger assumption [12].

Since DDH in the Gap groups G_1 and G_2 is easy, DDH can not serve as a basis for crypto systems in these groups. Therefore, an alternative to the CDH problem is defined called the Bilinear Diffie-Hellman problem.

In the definition that follows $\mathcal{G}\left(1^k\right)$ is defined to be a BDH parameter generator as in [12], i.e. $\mathcal{G}$ takes as input a security parameter k, $\mathcal{G}$ runs in polynomial time in k and $\mathcal{G}$ outputs a prime number q, the description of two groups G_1, G_2 of order q and the description of an admissible bilinear map $e : G_1 \times G_2 \to G_T$.

Definition 2.22 (BDH). The *Bilinear Diffie-Hellman problem* is defined as follows. Given any admissible bilinear pairing $e : G_1 \times G_2 \to G_T$ with random $P, aP, bP \in G_1$ and random $Q, aQ, bQ \in G_2$ with uniformly chosen random independent elements $a, b, c \in \{1, \ldots, |G|\}$, find $e\left(P, Q\right)^{abc}$

The *Bilinear Diffie-Hellman assumption* holds if for any algorithm $\mathcal{A}\left(P, aP, bP, Q, aQ, bQ\right)$ trying to solve the BDH problem there exists a negligible function $\mu\left(k\right)$ such that

$$\Pr\left[\mathcal{A}\left(P, aP, bP, Q, aQ, bQ\right) = e\left(P, Q\right)^{abc} \mid \langle q, G_1, G_2, e\rangle \leftarrow \mathcal{G}\left(1^k\right)\right] \le \mu\left(k\right)$$

where the probability is over the random choice of q, G_1, G_2, e according to the distribution induced by $\mathcal{G}\left(1^k\right)$, the random choice of a, b in $\{1, \ldots, |G|\}$ and the random bits of the algorithm $\mathcal{A}$.

2.5 Hash Functions

2.5.1 Definition

A *hash function* is a computationally efficient deterministic function mapping binary strings of arbitrary length to binary strings of some fixed length, called *hash-values.*

Cryptographic hash functions have the following desirable properties:

- *Computability:* Given a binary string m, the hash value h can be calculated efficiently $h = \texttt{hash}\left(m\right)$

- *Pre-image resistance:* Given a hash value h, it is infeasible to calculate a corresponding binary string m such that $h = \texttt{hash}(m)$

- *Second pre-image resistance:* Given a binary string m_1, it is hard to find a different binary string m_2 such that $\texttt{hash}(m_1) = \texttt{hash}(m_2)$

- *Strong collision resistance:* Given a `hash` function `hash(.)`, it is hard to find two different binary strings m_1 and m_2 such that $\texttt{hash}(m_1) = \texttt{hash}(m_2)$

Hash functions are useful in a wide plethora of practical applications. Hash functions serve as one way functions in password databases to relax sensitivity of the stored content. Furthermore hash functions are a valuable tool for data authentication and integrity checking. Another use of hash functions is in protocols involving a priori commitments. If the reader is new to the concept of hash functions, he is referred to [31] for an in depth discussion on the topic.

2.5.2 Random Oracles

A *random oracle* is a theoretical black box that returns for each unique query a uniformly random chosen result from its output domain. A random oracle is deterministic, i.e. given a particular input it will always produce the same output.

In a perfect world hash functions can be considered random oracles. That is, if hash functions were perfect, they would behave as random oracles. Therefore, hash functions are often considered random oracles in security proofs. Such security proofs are said to be *proven secure in the random oracle model.* Proofs in the random oracle model first show that an algorithm is secure if a theoretical random oracle would be used. A next step of these security proofs is replacing the random oracle accesses by the computation of an appropriately chosen (hash) function h [7]. Algorithms that do not require such a construction in their security proof are said to be *proven secure in the standard model.*

Although theoretical definitions of random oracles and hash functions are quite similar, some practical implementations of hash functions do not behave like random oracles at all. Canetti at al. show that there exist signature and encryption schemes that are secure in the Random Oracle Model, although any implementation of the random oracle results in insecure schemes [14]. Coron et al. counter these findings with indifferentiability, i.e. if a hash function is indifferentiable from a random oracle the random oracle can be replaced by the hash function while maintaining a valid security proof [18]. Although research results from Coron et al. are debated in [21] and [33], it is a common belief that proofs in the random oracle model provide some evidence that a system is secure. As a matter of fact, indifferentiability from random oracles certainly contributed to the victory of Keccak in the NIST hash function competition for a new SHA-3 hashing standard as all final round hashing algorithms supported this property [6].

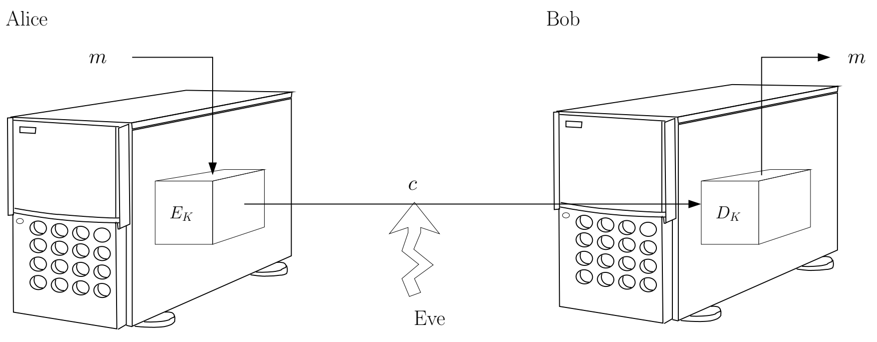

Figure 2.1: A cryptosystem [2]

2.6 Cryptology - this section is temporary moved from chapter 3

Cryptology is the science describing how to hide confidential information. Cryptology consists of two complementary fields that continuously try to outwit each other: cryptography and cryptanalysis. On the one hand, cryptography is the practice and study of techniques trying to hide information from undesired third parties. On the other hand, cryptanalysis is the domain of cryptology trying to derive information from hidden data.

2.6.1 Symmetric Cryptography

Figure 2.1 shows a typical cryptographic system often shortened to "cryptosystem". In a typical cryptosystem one party (often called Alice) tries to send a message m over an insecure channel to another party (often called Bob). The channel is insecure as third parties like Eve can eavesdrop on the channel to read out data that is being sent over.

Confidentiality

Figure 2.1 achieves confidentiality, i.e. the information in m is protected from disclosure to unauthorised parties. To prevent eavesdroppers from reading out the message m, Alice and Bob have agreed on a key k that is unknown to the outside world. Before sending a message m over the insecure channel, Alice encrypts the message m to a ciphertext c under the secret key k using an encryption algorithm E_k such that $c = E_k(m)$. Ideally c looks like random gibberish to eavesdroppers like Eve. Bob can then read out the original plaintext message m by applying the decryption algorithm D_k under the same key k. Cryptosystems as the one described in Figure 2.1 are called *symmetric* as both Alice and Bob have to use the same key k for encryption and decryption.

Most cryptosystems that are practically used today satisfy Kerchoff's principle. Kerchoff's principle states that although the encryption and decryption mechanism are known to the outside world, the cryptosystem assures confidentiality as long as the symmetric key k remains secret.

One-time Pad

A simple symmetric encryption algorithm E_k could be to XOR the binary message m with a binary key k such that the ciphertext is equal to $c = m \oplus k$. Decryption D_k would then consist of an XOR operation with the same binary symmetric key k such that $m = c \oplus k = (m \oplus k) \oplus k$. In such a scheme, the key k should be a random binary string with the same length as the plaintext message m. This scheme was originally proposed by Vernam and is therefore often called the *Vernam scheme*.

Further research on the Vernam scheme by Mauborgne showed that the Vernam scheme can be proven information theoretic secure if k is chosen completely random and used only once. Because of these requirements on the key k, the Vernam scheme is more widely known as the *one-time pad*.

Practical Encryption Algorithms

Because the one-time pad is proven information theoretic secure, it can not be broken even if the adversary has access to unlimited computing power. Although this is a desirable property, the one-time pad is not frequently used in modern cryptosystems due to its impractical key management.

Suppose Alice and Bob have a lot of secret information to share. This would require that the a priori agreed key k is long enough to hide all this information. Once the size of the message m becomes larger than the key k, Alice and Bob should agree on new random bits in k to secure the remainder of their conversation. In fact, they have to agree upfront on as much random bits in k as there will be bits in the message m.

Because such large random symmetric keys k are not practical in real-life applications, block cipher modes and stream ciphers are widely used. These are algorithms that accept a fixed size symmetric key but allow to encrypt larger messages m by introducing deterministic pseudo randomness. As already mentioned in Section 2.3, these algorithms require a more pragmatic view on cryptography because they only ensure that disclosure of information is computationally difficult but not impossible. Common examples of block ciphers are AES and DES. They can be used in CFB, CTR or CFB mode to name a few. Stream ciphers include Trivium and RC4. For more information on block ciphers, stream ciphers and modes of operation the reader is referred to [31].

2.6.2 Asymmetric Cryptography

In the information society of today, it would not be practical if Alice should meet Bob in real-life each time she wants to privately agree on a new symmetric key k. Now suppose that it would be possible to encrypt messages with a key *pk* and decrypt them with a corresponding different key *sk*. In such a setting, Bob could publish his personal encryption key pk_{Bob} while keeping his decryption key sk_{Bob} private thereby allowing Alice to immediately start sending private messages to Bob.

The concept of using a different key for encryption than decryption is often referred to as *asymmetric cryptography.* The term *public-key cryptography* describes the same idea and is interchangeably used in literature.

The concept of asymmetric cryptography bears close resemblance to the old-school mailbox system. Everyone can put letters in a mailbox, i.e. encrypt, but only a person with a privately owned key can retrieve letters, i.e. decrypt [32].

Trapdoor One-way Function

Public-key cryptography revolutionised thanks to a paper from Diffie and Hellman in 1976 proposing a private key exchange algorithm, now famously known as Diffie-Hellman key exchange [19][2]. Although Diffie-Hellman key exchange simplified the most important key management aspect of symmetric cryptography at the time, the real trigger for asymmetric cryptography appeared to be the introduction of a trapdoor one-way function.

Definition 2.23 (One-way Function). A function $f : (0,1)^* \to (0,1)^*$ that is computable in polynomial time, is said to be a one-way function if for any algorithm $\mathcal{A}(f(x))$ trying to invert $f(x)$, there exist a negligible function $\mu(k)$ such that

$$\Pr\left[f\left(\mathcal{A}\left(f\left(x\right)\right)\right) = f\left(x\right)\right] \leq \mu\left(k\right)$$

where the probability is over the random choice of x from the uniform random distribution on $(0,1)^k$ and the random bits of the algorithm $\mathcal{A}$.

A hash function (Section 2.5) is a practical implementation of a one-way function.

Definition 2.24 (Trapdoor one-way function). A one-way function $f : (0,1)^* \to (0,1)^*$ is said to be a trapdoor one-way function if there exist a specific algorithm $\mathcal{A}'\left(f\left(x\right), \mathcal{H}\right)$ that can invert $f(x)$ based on an additional hint $\mathcal{H}$, such that for any negligible function $\mu(k)$

$$\Pr\left[f\left(\mathcal{A}'\left(f\left(x\right)\right)\right) = f\left(x\right)\right] > \mu\left(k\right)$$

where the probability is over the random choice of x from the uniform random distribution on $(0,1)^k$ and the random bits of the algorithm $\mathcal{A}'$.

The Computational Diffie-Hellman problem (Section 2.3) is a trapdoor one-way function because it is hard to find g^{ab} given $\left\langle g, g^a, g^b \right\rangle$ but easy given $\left\langle g, g^a, g^b, \mathcal{H} \right\rangle$ if the hint $\mathcal{H}$ equals a or b.

[2]Diffie-Hellman key exchange allows two parties that can only communicate over an insecure channel to agree on a secret while no external passive eavesdropper with limited computing power can derive the secret.

Practical Encryption Algorithms

Trapdoor one-way functions can be used as a generic construction for asymmetric encryption by publishing all the parameters for the one-way function publicly while keeping the corresponding hint $\mathcal{H}$ private.

Digital Signatures

A digital signature resembles a handwritten signature in that it proofs a particular person has approved a particular message. However, a digital signature is harder to forge than its handwritten counterpart due to the computational hardness assumptions digital signatures rely on.

Definition 2.25 (Digital signature). A digital signature $S_A(m)$ associates a message m with a known sender A in such a way that a recipient B is assured about the following properties:

1. *Authentication:* B can be certain that A is the sender of the message.
2. *Non-repudiation:* A can not deny having sent the message m.
3. *Integrity:* B can be certain that the message m is delivered consistently, i.e. unaltered from how A originally drafted the message m.

Algorithm 1 explains how a generic signature scheme is often constructed. The key pair $\langle sk_A, vk_A \rangle$ is often Note that a signature scheme only shifts the authentication problem. Verification of a signature $S_A(m)$ only ensures the message m originates from the owner of the key pair $\langle sk_A, vk_A \rangle$.

Algorithm 1 Generic Signature Scheme

In a digital signature scheme each entity A has a publicly known verifying key vk_A and a corresponding private signing key sk_A. A generic signature scheme consists of two algorithms:

1. `Sign`(sk_A, m): Entity A signs the message m using its private signing key sk_A resulting in a signature $S_A(m)$
2. `Verify`$(pk_A, S_A(m), m)$: Entity B verifies the signature $S_A(m)$ with the public verifying key vk_A of A. The `Verify` step returns `true` or `false` depending on the validity of the signature.

Certification Authorities

OpenPGP

2.7 Conclusion

The first part of this chapter introduced the concepts of a negligible function as well as algebraic structures such as groups and finite fields. These basic notions were used further on to define number theoretic hard problems that serve as a basis for security. Starting from the discrete logarithm assumption, several variants of the Diffie-Hellman problem were introduced eventually leading to the Gap Diffie-Hellman assumption. Notion of the Gap Diffie-Hellman assumption allowed to uncover gap groups and their use in admissible bilinear maps. The Bilinear Diffie-Hellman assumption was defined as a computationally infeasible problem for the construction of cryptographic protocols relying on bilinear maps. Finally, this chapter concluded with differences between security under random oracle assumptions and security in the standard model.

Now the reader has knowledge of the mathematic fundaments, more advanced cryptographic constructions like identity-based encryption, broadcast encryption and distributed key generation are revealed in Chapter 3.

Chapter

3 Literature Review

3.1 Identity-Based Encryption

Shamir [35] already proposed a concept of identity-based cryptography in 1984. In identity-based cryptography any string can be a valid public key for encryption or signature schemes thereby eliminating the need for digital certificates. Identity-based cryptography proves to be particularly elegant if the public key is related to an attribute that uniquely identifies the identity of the user like an e-mail address, an IP address or a telephone number. Consequently, identity-based cryptography reduces system complexity and the cost for establishing and managing the Public Key Infrastructure (PKI) [4].

3.1.1 Definition

A generic Identity-Based Encryption (IBE) scheme is composed of four probabilistic polynomial time algorithms:

`IBE.Setup(`1^k`)` On input of a security parameter k, outputs a master secret s_k and public parameters $params$.

`IBE.Extract(`$params, s_k,$ `id):` Takes public parameters $params$, the master secret s, and an `id` as input and returns the private key $s_{\texttt{id}}$ corresponding to the identity `id`.

`IBE.Encrypt(`$params,$ `id`$, m$`):` Returns the encryption c of the message m on the input of the public parameters $params$, the `id`, and the arbitrary length message m.

`IBE.Decrypt(`$s_{\texttt{id}}, c$`):` Decrypts the ciphertext $c =$ `IBE.Encrypt`$(params, \texttt{id}, m)$ back to the message m on input of the secret key $s_{\texttt{id}}$ corresponding to the receiving identity `id`.

Although Shamir easily constructed an identity-based signature scheme based on RSA in 1984, the use case of IBE remained an open problem until the introduction of bilinear maps. In [12] Boneh and Franklin propose the first practically usable IBE

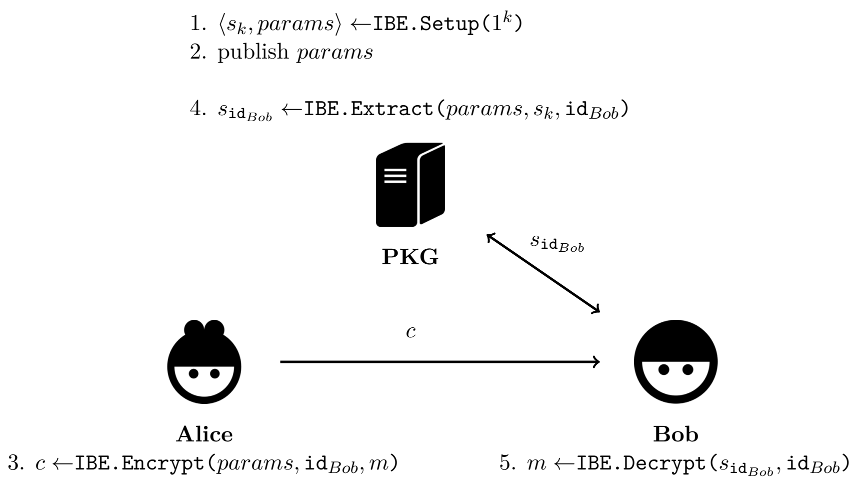
1. $\langle s_k, params\rangle \leftarrow$ IBE.Setup(1^k)
2. publish $params$
4. $s_{\mathtt{id}_{Bob}} \leftarrow$ IBE.Extract($params, s_k, \mathtt{id}_{Bob}$)
PKG
$s_{\mathtt{id}_{Bob}}$
c
Alice
3. $c \leftarrow$ IBE.Encrypt($params, \mathtt{id}_{Bob}, m$)
Bob
5. $m \leftarrow$ IBE.Decrypt($s_{\mathtt{id}_{Bob}}, \mathtt{id}_{Bob}$)

Figure 3.1: Multiple (n, t)-PKG IBE for OSNs overview, for a message m published for the set $\mathcal{S}$ for $t = 3$.

scheme based on the Weil pairing. However, the security proof in [12] still relies on the random oracle assumption. Canetti et al. [15] succeed in proposing a secure IBE scheme without having to rely on the random oracle model. However, the attacker model in [15] requires the adversary to declare which identity it will target, therefore the scheme in [12] is considered more secure as attackers can adaptively choose the targeted identity. Gentry [**?**] proposes a more efficient alternative to this scheme without random oracles while achieving shorter public parameters.

3.1.2 Anonymous Identity-Based Encryption

3.2 Broadcast Encryption

3.2.1 Definition

3.2.2 Anonymous Broadcast Encryption

3.2.3 Outsider-Anonymous Broadcast Encryption

3.3 Secret Sharing

3.3.1 Definition

Definition 3.1 (Secret Sharing Scheme). A *Secret Sharing Scheme* is a cryptographic scheme that divides a secret S into n pieces of data $S_1, \ldots, S_n$ called *shares*. Shares are distributed over n different parties called *shareholders* such that specific subsets of the distributed shares allow reconstruction of the original secret S.

Definition 3.2 (Threshold scheme). A (t, n) *threshold scheme* $(t \leq n)$ is a secret sharing scheme by which a trusted party securely distributes n different shares S_i to n different parties P_i for $1 \leq i \leq n$ such that any subset of t or more different shares S_i easily allows to reconstruct the original secret S. Knowledge of $t-1$ or less shares is insufficient to reconstruct the original secret S.

Definition 3.3 (Perfect threshold scheme). A (t, n) threshold scheme is said to be *perfect* if no subset of fewer than t shareholders can derive any partial information in the information theoretic sense about the original secret S even with infinite computational resources.

3.3.2 Shamir Secret Sharing

In 1979, both Shamir [34] and Blakley [10] independently found an algorithm achieving perfect threshold secret sharing. Shamir's solution was based on polynomial interpolation while Blakley's algorithm relied on finite geometries. Blakley secret sharing uses more bits than necessary as it describes multidimensional planes. In contrast, Shamir secret sharing requires as many bits for each share as the length of the original secret. Therefore Shamir secret sharing has gained more popularity in both research communities and in practical implementations.

Algorithm 2 Shamir's (t, n) threshold scheme [31]

Goal: A trusted party T distributes shares of a secret S to n parties.
Result: If a subset of at least t out of n shareholders collaborates, they can reconstruct the original secret S.

1. *Setup* The trusted party T begins with a secret integer $S \geq 0$ it wishes to distribute among n parties
   a) T chooses a prime $p > \max(S, n)$ and defines $a_0 = S$
   b) T selects $t-1$ random, independent coefficients $a_1, \dots, a_{t-1}, 0 \leq a_j \leq p-1$ defining the random polynomial over $\mathcal{Z}_p$, $f(x) = \sum_{j=0}^{t-1} a_j x^j$
   c) T computes $S_i = f(i) \bmod p, 1 \leq i \leq n$ and securely transfers the share S_i to shareholder P_i, along with a public index i.
2. *Reconstruction* Any group of t or more shareholders pool their shares. Their shares provide t distinct points $(x, y) = (i, S_i)$ allowing computation of the coefficients $a_j, 1 \leq j \leq t-1$ of $f(x)$ by Lagrange interpolation. The secret is recovered by calculating

$$f(0) = \sum_{i=1}^{t} y_i \prod_{1 \leq j \leq t, j \neq i} \frac{x_j}{x_j - x_i} = S$$

The idea behind Shamir secret sharing is elegant in its simplicity. Any polynomial $f(x)$ of degree $t-1$ is uniquely defined by t points lying on the polynomial. For example, it is possible to draw only one straight line between 2 different coordinates, a quadratic is fully defined by 3 different coordinates and so on. If the trusted party randomly generates a polynomial of degree $t-1$ it suffices to securely distribute one of n different coordinates on the curve to each party $P_i, 0 \leq i \leq n$. A subset of at least t different shareholders has to collaborate in order to reconstruct the original polynomial by interpolation. For security reasons the polynomial $f(x)$ is calculated in a finite field modulo a large prime number p. The complete mechanism of Shamir's threshold scheme can be found in Algorithm 2. The mechanism behind reconstruction in Algorithm 2 is explained because the coefficients of an unknown polynomial $f(x)$ of degree less than t, defined by points $(x_i, y_i), 1 \leq i \leq t$ are given by the Lagrange interpolation formula

$$f(x) = \sum_{i=1}^{t} y_i \prod_{1 \leq j \leq t, j \neq i} \frac{x - x_j}{x_i - x_j}$$

A proof of this formula is omitted but can be found in [37].

3.3.3 Verifiable Secret Sharing

Verifiable secret sharing [17] tries to ensure the participating parties that their received shares are consistent by providing a verification mechanism. This verification mechanism can either detect an unfair dealer during setup or participants submitting incorrect shares during the reconstruction phase. The first verifiable secret sharing schemes were *interactive*, i.e. interaction between shareholders and the trusted party was required to verify their shares. In *non-interactive verifiable secret sharing* only the trusted party is allowed to send messages to the future shareholders. Shareholders can not communicate with each other neither can they send messages back to the trusted party. Non-interactive verifiable secret sharing is preferred over interactive alternatives as their is no chance of shareholders accidentally leaking too much information.

Popular verifiable secret sharing schemes are Feldman's scheme [20] and Benaloh's scheme [8]. No further details are given as a basic notion of verifiable secret sharing suffices for the remainder of this text.

3.4 Distributed Key Generation

3.5 Conclusion

The final section of the chapter gives an overview of the important results of this chapter. This implies that the introductory chapter and the concluding chapter don't need a conclusion.

Nunc sed pede. Praesent vitae lectus. Praesent neque justo, vehicula eget, interdum id, facilisis et, nibh. Phasellus at purus et libero lacinia dictum. Fusce aliquet. Nulla eu ante placerat leo semper dictum. Mauris metus. Curabitur lobortis. Curabitur sollicitudin hendrerit nunc. Donec ultrices lacus id ipsum.

Chapter

4

Outsider Anonymous Identity-Based Broadcast Encryption

4.1 Security Model

4.1.1 Threat Model

4.1.2 Goals

4.2 Proposed Scheme

4.2.1 Scheme

4.2.2 Security Proof

4.2.3 Evaluation

4.3 Conclusion

Chapter

5 Implementation

5.1 Existing Solutions

5.2 Anonymous Identity-Based Broadcasting Implementation

5.2.1 Implemented Scheme

5.2.2 Data Structures

5.3 Distributed Key Generation Implementation

5.3.1 Implemented Scheme

5.3.2 Data Structures

5.4 Evaluation

5.5 Performance Analysis

5.6 Conclusion

Chapter

6 Conclusion

The final chapter contains the overall conclusion. It also contains suggestions for future work and industrial applications.

Appendices

Appendix

A Installing and Executing the Code

Appendices hold useful data which is not essential to understand the work done in the master thesis. An example is a (program) source. An appendix can also have sections as well as figures and references[**?**].

A.1 Setting up the DKG

A.2 Setting up Scramble

Appendix

B The Last Appendix

Appendices are numbered with letters, but the sections and subsections use arabic numerals, as can be seen below.

B.1 Lorem 20-24

B.2 Lorem 25-27

Bibliography

[1] Pairing based cryptography. Master's thesis, Technische Universiteit Eindhoven, 2004.

[2] White-box cryptography. Master's thesis, KU Leuven, 2009.

[3] G. Adj, A. Menezes, T. Oliveira, and F. Rodríguez-Henríquez. Weakness of x25;$_{3}^{6\cdot 509}$ for discrete logarithm cryptography. In Cao and Zhang [16], pages 20–44.

[4] J. Baek, J. Newmarch, R. Safavi-naini, and W. Susilo. A survey of identity-based cryptography. In *Proc. of Australian Unix Users Group Annual Conference*, pages 95–102, 2004.

[5] R. Barbulescu, P. Gaudry, A. Joux, and E. Thomé. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT*, volume 8441 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2014.

[6] G. Barthe, B. Grégoire, S. Heraud, F. Olmedo, and S. Z. Béguelin. Verified indifferentiable hashing into elliptic curves. *Journal of Computer Security*, 21(6):881–917, 2013.

[7] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In D. E. Denning, R. Pyle, R. Ganesan, R. S. Sandhu, and V. Ashby, editors, *ACM Conference on Computer and Communications Security*, pages 62–73. ACM, 1993.

[8] J. C. Benaloh. Secret sharing homomorphisms: Keeping shares of a secret sharing. In A. M. Odlyzko, editor, *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 251–260. Springer, 1986.

[9] G. Birkhoff and S. MacLane. *A Survey of Modern Algebra.* The Macmillan Comp., 1965.

[10] G. Blakley. Safeguarding cryptographic keys. In *Proceedings of the 1979 AFIPS National Computer Conference*, pages 313–317, Monval, NJ, USA, 1979. AFIPS Press.

[11] D. Boneh. The decision diffie-hellman problem. In J. Buhler, editor, *ANTS*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer, 1998.

[12] D. Boneh and M. K. Franklin. Identity based encryption from the Weil pairing. *IACR Cryptology ePrint Archive*, 2001:90, 2001.

[13] J. Bullas. 22 social media facts and statistics you should know in 2014. URL: `http://www.jeffbullas.com/2014/01/17/20-social-media-facts-and-statistics-you-should-know-in-2014/`, last checked on 2014-05-08.

[14] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004.

[15] R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. *IACR Cryptology ePrint Archive*, 2003:83, 2003.

[16] Z. Cao and F. Zhang, editors. *Pairing-Based Cryptography - Pairing 2013 - 6th International Conference, Beijing, China, November 22-24, 2013, Revised Selected Papers*, volume 8365 of *Lecture Notes in Computer Science*. Springer, 2014.

[17] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In *FOCS*, pages 383–395, 1985.

[18] J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-damgård revisited: How to construct a hash function. In V. Shoup, editor, *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448. Springer, 2005.

[19] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[20] P. Feldman. A practical scheme for non-interactive verifiable secret sharing. In *FOCS*, pages 427–437. IEEE Computer Society, 1987.

[21] E. Fleischmann, M. Gorski, and S. Lucks. Some observations on indifferentiability. *IACR Cryptology ePrint Archive*, 2010:222, 2010.

[22] G. Frey, M. Müller, and H.-G. Rück. The tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. *IEEE Transactions on Information Theory*, 45(5):1717–1719, 1999.

[23] O. Goldreich. On the foundations of modern cryptography. In B. S. K. Jr., editor, *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 46–74. Springer, 1997.

[24] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270 – 299, 1984.

[25] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[26] K. Jones. The growth of social media v2.0. URL: `http://www.searchenginejournal.com/growth-social-media-2-0-infographic/77055/`, last checked on 2014-05-08.

[27] A. Joux. A new index calculus algorithm with complexity l(1/4+o(1)) in very small characteristic. *IACR Cryptology ePrint Archive*, 2013:95, 2013.

[28] A. Joux and K. Nguyen. Separating decision diffie-hellman from computational diffie-hellman in cryptographic groups. *J. Cryptology*, 16(4):239–247, 2003.

[29] U. M. Maurer and S. Wolf. Lower bounds on generic algorithms in groups. In K. Nyberg, editor, *EUROCRYPT*, volume 1403 of *Lecture Notes in Computer Science*, pages 72–84. Springer, 1998.

[30] U. M. Maurer and S. Wolf. The relationship between breaking the diffie-hellman protocol and computing discrete logarithms. *SIAM J. Comput.*, 28(5):1689–1721, 1999.

[31] A. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996.

[32] C. Paar and J. Pelzl. *Understanding Cryptography - A Textbook for Students and Practitioners.* Springer, 2010.

[33] T. Ristenpart, H. Shacham, and T. Shrimpton. Careful with composition: Limitations of indifferentiability and universal composability. *IACR Cryptology ePrint Archive*, 2011:339, 2011.

[34] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[35] A. Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and D. Chaum, editors, *CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1984.

[36] StatisticBrain. Social networking statistics. URL: `http://www.statisticbrain.com/social-networking-statistics/`, last checked on 2014-05-08.

[37] P. Wiki. Lagrange interpolation formula. URL: `http://www.proofwiki.org/wiki/Lagrange_Interpolation_Formula`.

[38] Worldometers. Worldometers real time world statistics. URL: `http://www.worldometers.info/`, last checked on 2014-05-08.

[39] X. Zhang and K. Wang. Fast symmetric pairing revisited. In Cao and Zhang [16], pages 131–148.

Master thesis filing card

Student: Stijn Meul

Title: Practical Identity-Based Encryption for Online Social Networks

UDC: 621.3

Abstract:

Here comes a very short abstract, containing no more than 500 words. LaTeX commands can be used here. Blank lines (or the command `\par`) are not allowed!

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Thesis submitted for the degree of Master of Science in Electrical Engineering, option Embedded Systems and Multimedia

Thesis supervisors: Prof. dr. ir. Bart Preneel
Prof. dr. ir. Vincent Rijmen

Assessors: Prof. dr. ir. Claudia Diaz
Prof. dr. ir. Frank Piessens

Mentor: Filipe Beato