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 **vi**

List of Symbols **viii**

1 Introduction **1**
1.1 Problem Statement 1
1.2 Existing Solutions 2
1.3 Goals 3
1.4 Main Idea 3
1.5 Structure of this Thesis 4

2 Required Background **5**
2.1 Complexity Theory 5
2.2 Abstract Algebra 6
2.3 Number Theoretic Assumptions 9
2.4 Bilinear Maps 11
2.5 Cryptographic Definitions 13
2.6 Summary 16

3 Cryptographic Building Blocks **17**
3.1 Public Key Infrastructures 17
3.2 Identity-Based Encryption 19
3.3 Broadcast Encryption 26
3.4 Secret Sharing 31
3.5 Distributed Key Generation 33

4 Design of a Practical Encryption Scheme for Online Social Networks **37**
4.1 Model of the Current Situation 37
4.2 Threat Model 40
4.3 Our Proposal 43

4.4 Summary 51

5 Implementation **55**
5.1 Software Architecture 55
5.2 Implementation Details 58
5.3 Implemented Scheme 60
5.4 Performance Analysis 61
5.5 Limitations of Current Implementation 62
5.6 Summary 63

6 Conclusion **65**

A Gentry's IBE Scheme **69**

B Installing and Executing the Code **71**
B.1 Setting up the DKG 71
B.2 Setting up Scramble 71

C Dutch Summary **73**

D English Paper **75**

Bibliography **77**

Abstract

Nowadays Online Social Networks (OSNs) constitute an important and useful communication channel. At the same time, coarse-grained privacy preferences protect the shared information insufficiently. Cryptographic techniques can provide interesting mechanism to protect privacy of users in OSNs. However, this approach faces several issues, such as, OSN provider acceptance, user adoption, key management and usability. We suggest a practical solution that uses Identity Based Encryption (IBE) to simplify key management and enforce confidentiality of data in OSNs. Moreover, we devise an outsider anonymous broadcast IBE scheme to disseminate information among multiple users, even if they are not using the system. Finally, we demonstrate the viability and tolerable overhead of our solution via an open-source prototype.

List of Figures and Tables

List of Figures

1.1 Multiple (n, t)-PKG IBE for OSNs overview, for a message m published for the set $\mathcal{S}$ for $t = 3$. The PKG infrastructure can be maintained by virtually any organisation with an incentive to make OSNs more private. 4

3.1 Generic identity-based encryption scheme. The blue arrow denotes an insecure channel that can be eavesdropped. 20

4.1 Model of the current OSN situation. Entities with access to the message m are coloured blue. 39
4.2 Model of the desired OSN situation. Entities with the ability to decrypt the ciphertext are coloured blue. 49

5.1 Original Scramble Architecture 56
5.2 New Scramble Architecture 57
5.3 Class Diagram of Client Side C++ MIRACL Based Back-end 60

List of Tables

4.1 Security comparison of considered IBE schemes 44
4.2 Performance comparison of considered IBE schemes in MIRACL 45

5.1 Performance of the proposed scheme in function of the number of intended recipients. 61

List of Abbreviations

ANO-IBE	Anonymous IBE
ANO-IND-CCA	Anonymity preserving IBE scheme that is indistinguishable under chosen ciphertext attacks
ANO-IND-CPA	Anonymity preserving IBE scheme that is indistinguishable under chosen plaintext attacks
DKG	Distributed Key Generation
IBE	Identity-Based Encryption
IND-CCA	Indistinguishability under Chosen Ciphertext Attack
IND-CPA	Indistinguishability under Chosen Plaintext Attack
OSN	Online Social Network
PKG	Public Key Generator
UI	User Interface

List of Symbols

λ	Security parameter
l	The number of bits required to realise security level λ
s	Secret
sk_i	Private key corresponding to the public key pk_i or the public verifying key vk_i depending on the application
pk_i	Public key with corresponding private key sk_i
vk_i	Verifying key with corresponding signing key sk_i
msk	Master secret key
$\{0,1\}^l$	Binary bit sequence of length l
$\{0,1\}^*$	Binary bit sequence of variable length
m	Message
c	Ciphertext
v, w	Binary bit sequences
$\{v \parallel w\}$	Concatenated bit sequences
$\mathtt{id}_{Alice}$	Identity of Alice
$s_{\mathtt{id}_{Alice}}$	IBE private key corresponding to the identifier $\mathtt{id}_{Alice}$
k	Generic symmetric session key
$E_k(m)$	Symmetric encryption of the message m under session key k
$D_k(c)$	Symmetric decryption of the ciphertext c under session key k
G	Group $(G, *)$
$S_A(m)$	Signature of entity A on message m
$S_{sk_A}(m)$	Signature generated by the signing key sk_A of entity A on message m
$e : G_1 \times G_2 \rightarrow G_T$	Bilinear map
U, P, Q	Points on an elliptic curve
$e(P, Q)$	Bilinear map for the points $P \in G_1, Q \in G_2$ such that $e(P, Q) \in G_T$
$\mathcal{A}(a, b)$	Algorithm $\mathcal{A}$ with parameters a and b
$\langle a, b, c\rangle \leftarrow \mathcal{A}(d, e)$	Algorithm $\mathcal{A}$ with parameters d and e, returns the collection of values a, b, c

Chapter

1 Introduction

The online social network (OSN) is the most impactful internet trend at the dawn of the 21st century. Words like tweeting, sharing, liking, trending and tagging have found common acceptance in the vocabulary of current internet users, while services like Facebook, Google+, LinkedIn and Twitter have become part of everyday life. OSNs offer millions of users an efficient and reliable channel to distribute and share information. At the same time, OSNs store large amounts of data which prompts several privacy concerns. In particular, it is possible to infer a considerable amount of sensitive information from the shared and stored content. Currently, users are allowed to configure "privacy preferences" in order to limit and select which users or groups can access the shared content. These preferences are generally too coarse-grained and difficult to configure [28]. Another problem is that these preferences do not exclude the provider along with the dangers of data leaks [53] nor external governments [94].

1.1 Problem Statement

All these worrisome issues motivate the need for effective techniques to properly protect user's privacy in OSNs. Several solutions have been proposed and advocated to use cryptographic mechanisms in order to address the privacy issues, either by an add-on atop of existing OSNs [8, 16, 68, 83], or by complete new privacy-friendly architectures [43], mainly decentralised [44, 49]. In general, those solutions suffer from user adoption and key management issues as users are required to register and then share, certify and store public keys [10]. Completely new architectures represent a difficult step for users as the trade off of moving away from the commonly used social ecosystem compared with the risk of losing interactions is high. Arguably, current centralised OSNs are here to stay and will be continue to be actively used by millions of people. In light of recent events, such as Edward Snowden's whistle-blowing on US surveillance programs [94], OSN providers have all interest to maintain their users and a privacy-friendly image.

1.2 Existing Solutions

Several existing solutions have been proposed in literature, all trying to solve most of the aforementioned issues in OSNs.

flyByNight [82] is a Facebook application that protects user data by storing it in encrypted form on Facebook. It relies on Facebook servers for its key management and is thus not secure against active attacks by Facebook itself.

NOYB (None Of Your Business) [68] replaces details of a user with details from other random users thereby making this process only reversible by friends. However, the proposed solution does not apply to user messages or status updates that are the most frequently used features in the OSNs considered in this thesis.

FaceCloak [83] stores published Facebook data on external servers in encrypted form and replaces the data on Facebook with random text from Wikipedia. This could be a useful mechanism to prevent OSNs from blocking security aware users because they are scared to see their advertising revenues shrink. However, this approach has the disadvantage that other users could take this data as genuine user content which may lead to social issues. Furthermore, FaceCloaks architecture leads to an inefficient key distribution system.

Persona [8] is a scheme that can be used as a Firefox extension to let users of an OSN determine their own privacy by supporting the ability to encrypt messages to a group of earlier defined friends based on *attribute-based encryption* (ABE) [98]. The scheme supports a wide range of meaningful use cases. For instance, sending messages to all friends that are related to a certain attribute or even encrypting messages to friends of friends. However, the major drawback of this system is that, for every new friend a public key is exchanged before he is able to interact in the privacy preserving architecture consequently requiring an infrastructure for broadcasting and storing public keys. Furthermore, to support the encryption of messages to friends of friends, user defined groups should be made available publicly thereby making the public key distribution system even more complicated. Finally the proposed ABE encryption scheme is 100 to 1000 times slower than a standard RSA operation [8].

Scramble [16] is a Firefox extension that allows users to define groups of friends that are given access to stored content on OSNs. The tool uses public key encryption based on OpenPGP [33] to broadcast encrypted messages on any platform. Furthermore Scramble provides the implementation of a tiny link server such that OSN policies not allowing to post encrypted data are bypassed. However, as indicated by usability studies [108] OpenPGP has a higher usage threshold because an average user does not manage to understand OpenPGP properly. Additionally, Scramble has to rely on the security decisions of the web of thrust. It therefore inherits the unpleasant property of OpenPGP that the user can not be sure that the used PGP key actually belongs to the intended Facebook profile.

The most unattractive property of all the above applications is that they have to rely on a rather complex infrastructure. Persona has to support an extended public key distribution system and Scramble relies on the leap-of-faith OpenPGP web of

trust. All proposed solutions require users with no cryptographic background on asymmetric cryptography to make responsible decisions concerning the management of their keys. Furthermore, maintaining such complex key infrastructures becomes more and more complex as more users subscribe.

1.3 Goals

The goal of this thesis is to develop an architecture that solves the aforementioned issues thereby taking the challenges and pitfalls from earlier solutions into account. Specifically, the architecture should present the following properties:

- **User friendly:** The average OSN user should be able to use the resulting architecture, i.e. a user with no knowledge on cryptographic primitives.
- **Applicable:** The original OSN environment should not be altered since some OSN providers are probably not willing to support a more confidential architecture because it could possibly hurt their business model.
- **Immediately ready to use:** No additional registration or subscription to third party key architectures should be required to enable usage of the system. As soon as a user subscribes to the OSN provider he should be able to start receiving confidential messages.

1.4 Main Idea

Identity Based Encryption (IBE) [103] solutions overcome the key management problem as the public key of the user can be represented by any valid string, such as the email, unique `id` and username. Therefore, by using an OSN username any savvy and concerned user can share encrypted content with other users who are not using the solution, thereby motivating curious ones to use the system as well. Nevertheless, IBE-based systems require a trusted central Private Key Generator (PKG) server to generate the private parameters for each user based on a master secret. Consequently, such an architecture only shifts the trusted party from the OSN to the PKG. However, this problem can be mitigated if the master secret is divided among multiple PKGs following a Distributed Key Generation (DKG) [93] protocol based on Verifiable Secret Sharing (VSS) [40]. A DKG protocol allows n entities to jointly generate a secret requiring that a threshold t of the n entities does not get compromised. In fact, each entity holds only a share of the master secret, that can be reconstructed by at least t shares.

Many OSN users are not only represented on a single OSN but on several, thus, can also hold multiple public keys. Moreover, the multi-PKG setting could be supported and maintained by different organisations, each with their own motivations to support more private OSNs. In particular, if OSN providers see their advertisement revenues drop due to privacy concerned users deleting their profiles, they have an incentive to support and maintain such a multi-PKG setting. Since collaboration between

OSN providers that compete along is assumed to be a difficult task and orthogonal to their economical business model, the different PKGs do not compromise the security of the DKG protocol. Figure 1.1 depicts an overview example of a possible model, where a user authenticates to t-PKGs of his choice using, e.g. a similar token as in open id protocols, to retrieve his private key. This action can be performed after the reception of encrypted content as a consequence of user curiosity. The PKG servers can also be represented by governmental entities or subsidised research institutions from different continents, with no incentives to collaborate nor overcome more powerful adversaries using legal measures [86] among at least t-PKGs.

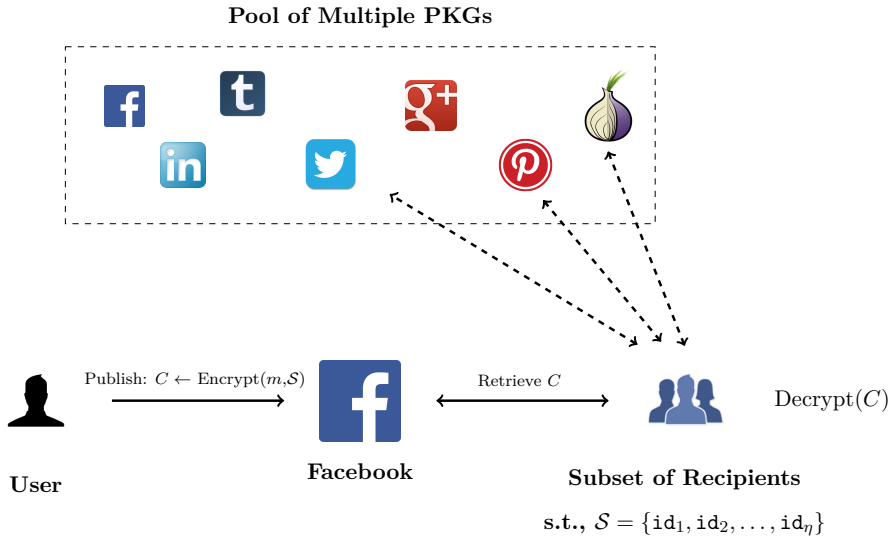

Figure 1.1: Multiple (n, t)-PKG IBE for OSNs overview, for a message m published for the set $\mathcal{S}$ for $t = 3$. The PKG infrastructure can be maintained by virtually any organisation with an incentive to make OSNs more private.

1.5 Structure of this Thesis

Chapter

2 Required Background

This chapter briefly covers the required background to understand cryptographic algorithms presented later in this text. The mathematical details of this chapter represent a fundament of a challenging world containing exciting cryptographic concepts like identity-based encryption. If the reader feels he has sufficient background of the concepts covered in this chapter, the chapter can be skipped without loss of comprehension.

Note that this chapter only overviews the 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 [90] and [20]. More information on elliptic curves, Diffie-Hellman assumptions and pairing based cryptography can be found in [2].

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

2.1 Complexity Theory

Complexity theory classifies mathematical problems according to their inherent difficulty. The inherent difficulty of a mathematical problem is expressed in terms of the required resources to solve the problem independent of the algorithm used [90].

Definition 2.1 (Asymptotic upper bound). A function $f(n)$ which is non-negative for all integers $n \geq 0$, has an asymptotic upper bound $g(n)$ denoted $f(n) = O(g(n))$, if there exists an integer n_0 and a constant $c > 0$ such that for all integers $n \geq n_0, f(n) \leq cg(n)$

Definition 2.2 (Polynomial-time algorithm [90]). A *polynomial-time algorithm* is an algorithm whose worst-case running time function is of the form $O(n^c)$ where n is an input parameter and c is a constant.

Definition 2.3 (Exponential-time algorithm [90]). Any algorithm whose running time is not computationally bounded like a polynomial-time algorithm, is called an *exponential-time algorithm.*

Exponential-time algorithms are inefficient since they can take a long time to complete. Security of cryptographic algorithms is guaranteed if no polynomial-time algorithm exists to reverse the cryptographic operation without additional information.

An algorithm is considered secure against computationally bounded adversaries 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.

Definition 2.4. A **negligible function** in λ is a function $\mu(\lambda) : \mathbb{N} \to \mathbb{R}$ if for every polynomial $p(\cdot)$ there exists an N such that for all $\lambda > N$ [62]

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

The negligible function is used along this chapter to formally describe computationally infeasible problems. In such a context λ often represents the security parameter. The larger λ will be chosen, the smaller $\mu(\lambda)$ 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 2.5 (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. [20, 90]

Definition 2.6 (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 2.7 (Commutative group). A group $(G, *)$ is called a *commutative group* or an *abelian group* if in addition to the properties in Definition 2.6, also commutativity holds.

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

Depending on the group operation $*$, $(G, *)$ is either called a *multiplicative group* or an *additive group*. In Definition 2.6 the multiplicative notation is used. For an additive group the inverse of a is often denoted $-a$ [90].

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 2.6 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 2.8 (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 2.8 implies that in a cyclic group every element can be written as a power of one of the group's generators.

Definition 2.9 (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 $e = 1$. For example, 2 is a generator in $\mathbb{Z}_5^*$ since every element in $\mathbb{Z}_5^*$ can be written as $\{2^n | n \in \mathbb{Z}\}$.

Definition 2.10 (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 2.11. *If the order of a group G equals a prime p, the group is cyclic and commutative.*

Definition 2.12 (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 2.13 (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 2.14 (Commutative ring). A ring $(R, +, *)$ is called a *commutative ring* or an *abelian ring* if in addition to the properties in Definition 2.13, also commutativity holds.

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

Definition 2.15 (Field). A commutative ring $(R, +, *)$is called a *field* if in addition to the properties in Definition 2.14 and Definition 2.13 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 2.16 (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 2.17 (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 2.18 (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 2.19 (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 cryptographic security of our future constructions falls or stands on these assumptions [23, 90].

In the definitions that follow $\langle G, n, g\rangle \leftarrow \mathcal{G}\left(1^{\lambda}\right)$ 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 2.20 (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

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

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

Definition 2.21 (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

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

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 2.22 (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

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

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

Definition 2.22 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*. This implies that no efficient algorithm exists that can distinguish both arguments with non-negligible probability. The concept of computational indistinguishability bears close resemblance to statistical indistinguishability. The reader is referred to [63, 64] for a more in depth discussion of the topic. The intuitive interpretation of Definition 2.22 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 the CDH assumption immediately follows from the DL assumption as shown in [87, 88]. 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. Contrarily, 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 2.23 (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

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

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

2.4 Bilinear Maps

It can be shown that bilinear pairings are an example of a practical usable DDH oracle [74].

2.4.1 Definition

Definition 2.24 (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* $\forall a, b \in \mathbb{Z}, \forall P \in G_1, \forall Q \in G_2 : e(aP, bQ) = e(P, Q)^{ab}$
2. *Non-degeneracy* If P is a generator of G_1 and Q is a generator of G_2, $e(P, Q)$ is a generator of G_T
3. *Computability* There is an efficient algorithm to compute $e(P, Q)$ for all $P \in G_1$ and $Q \in G_2$

In literature, authors distinguish two types of admissible bilinear maps: symmetric and asymmetric 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 2.24 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 [27, 111].

In practice, bilinear maps are constructed using pairings. The most popular pairings implementing admissible bilinear maps are the Weil pairing [27] and the Tate pairing [57]. 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 $E(\mathbb{F}_p)$ 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 [2] 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.

Recent research [7, 12, 73] has has shown that the discrete logarithm problem is easier in the symmetric setting because symmetric pairings rely on more structured supersingular (hyper)elliptic curves. Therefore, it is discouraged to rely on symmetric pairings in practical implementations [111].

2.4.2 Bilinear Diffie-Hellman Assumption

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 is found by relying on the bilinearity of the pairing as follows:

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

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. Since DDH (Definition 2.22) is a stronger assumption than CDH (Definition 2.21), CDH can still be hard in GDH groups [27].

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 $\mathcal{G}\left(1^{\lambda}\right)$ is defined to be a BDH parameter generator as in [27], i.e. $\mathcal{G}$ takes as input a security parameter λ, $\mathcal{G}$ runs in polynomial time in λ 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 2.25 (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

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

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

2.5 Cryptographic Definitions

This section defines basic cryptographic notions and the corresponding notation that is applied consistently throughout the remainder of the text.

2.5.1 Terminology

Definition 2.26 (Perfect randomness). A bit sequence s contains *perfect randomness* or is said to be chosen *uniformly random* if every bit b in s could have been the result of the toss of a fair coin. That is, the probability of $b = 1$ equals the probability of $b = 0$ or more formally, $\Pr[b = 1] = \Pr[b = 0] = \frac{1}{2}$.

Definition 2.27 (Confidentiality). *Confidentiality* is the assurance to an entity information is protected from disclosure to unauthorised entities.

Definition 2.28 (Integrity). *Integrity* is the assurance to an entity information was not modified by unauthorised entities.

Definition 2.29 (Authentication). *Authentication* is the assurance to an entity that another entity effectively has a claimed identity.

Definition 2.30 (Authenticity). *Authenticity* is the assurance to an entity information comes from the claimed entity.

Definition 2.31 (Non-repudiation). *Non-repudiation* is the assurance to an entity of authenticity and integrity of information which undeniably links the originating entity as the source of information.

2.5.2 Symmetric Cryptography

Symmetric cryptographic algorithms require a shared secret between different entities to achieve confidentiality. Practically, this translates to the same symmetric key k being used for encryption and decryption.

Definition 2.32 (Encryption scheme). An encryption scheme consists of two polynomial time algorithms which achieve confidentiality:

1. An encryption algorithm $c \leftarrow \mathtt{E}_k(m)$ that encodes the plaintext m to a ciphertext c under symmetric key k such that only parties in possession of k can derive m from c.

2. A decryption algorithm $m \leftarrow \mathtt{D}_k(c)$ that decodes the ciphertext c back to the plaintext message m on input of the same symmetric key k.

Definition 2.33 (Authenticated encryption). An authenticated encryption scheme consists of two polynomial time algorithms which achieve confidentiality and authenticity:

1. An encryption algorithm $\langle c, t \rangle \leftarrow \mathtt{E}_k(m, a)$ that encrypts the plaintext m to a ciphertext c under symmetric key k, thereby generating an authentication tag t that provides integrity and authenticity on both m and authenticated data a.

2. A decryption algorithm $\langle m, t' \rangle \leftarrow \mathtt{D}_k(c, a)$ that decrypts the ciphertext c back to the plaintext message m under the same symmetric key k as used for encryption, thereby generating an authentication tag t' that provides integrity and authenticity on both m and authenticated data a.

The difference in notation between an encryption scheme and an authenticated encryption scheme can be derived from the different number of arguments required by their polynomial time algorithms.

2.5.3 Asymmetric Cryptography

Asymmetric cryptography assigns a key pair $\langle sk_A, pk_A \rangle$ to every entity A in the system. The *private key* is denoted sk_A and only known by A, while pk_A represents the *public key* which is made available to every entity in the system. A private key sk_A is mathematically related to the corresponding public key pk_A since pk_A is derived from sk_A by applying a one-way function. The one-way property of the function implies that there exists no polynomial time algorithm to derive sk_A from pk_A, e.g. in the ElGamal encryption scheme [58] the public key is calculated as $pk = g^{sk}$ in a group $\mathbb{Z}_p$ for some large prime p. As long as the DL assumption from Definition 2.20 holds it is infeasible to derive sk from pk.

The concept of each entity possessing an asymmetric key pair enables secure communication between entities who have never met by encrypting under the correct public keys.

Definition 2.34 (PKI). A *public key infrastructure* (PKI) is an infrastructure authenticating key pairs $\langle sk_A, pk_A \rangle$ effectively belong to the claimed user A.

Definition 2.35 (Digital signature). A digital signature $S_A(m)$ achieves non-repudiation on a message m from a known sender A. m is signed with A's private signing key sk_A and verified with A's public verifying key vk_A.

Definition 2.36 (Commitment scheme). A *commitment scheme* allows an entity to commit to a chosen value while keeping it hidden to others, with the ability to reveal the committed value later [62]. After revealing the committed value, any other entity can verify the value has not changed between commitment and revelation. This is achieved by two polynomial time algorithms:

1. `CS.Commit`(m, r): Returns a commitment $c_{m,r}$ to a message m and a random binary sequence r.

2. `CS.Verify`$(c_{m,r}, m', r')$: On input of a commitment $c_{m,r}$, a message m' and a random binary sequence r' it returns `true` if $c_{m,r} \leftarrow$`CS.Commit`$(params, m, r)$ with $m = m'$ and $r = r'$ and `false` otherwise.

For a more elaborate discussion on commitment schemes the reader is referred to the original paper from Brassard et al. [32].

2.5.4 Hash Functions

The concept of hash functions is required to further explain random oracles. Random oracles are a useful assumption when proving the security of certain cryptographic algorithms.

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 $h = \texttt{hash}(m)$
- *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 for a wide variety of practical applications. For instance, hash functions serve as one way functions in password databases to relax sensitivity of the stored content. In addition, hash functions represent 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 [90] for an in depth discussion on the topic.

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, their output would look like perfect random bit sequences. 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 [18]. 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. [34] 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 [34]. 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 [41]. Therefore, it is a common belief that proofs in the random oracle model provide some evidence that a system is secure. Although research results from Coron et al. are debated in [54] and [97]. In 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 [15].

2.6 Summary

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.

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. From the discrete logarithm assumption, several variants of the Diffie-Hellman problem were introduced, eventually leading to the Gap Diffie-Hellman assumption. The 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.

Chapter

3 Cryptographic Building Blocks

This chapter overviews the cryptographic building blocks used to design the encryption mechanism for online social networks proposed in this thesis.

This chapter is organised as follows. An introduction is given to public key infrastructures and their drawbacks. Then, identity-based encryption (IBE) is overviewed as an alternative to the existing public key infrastructures along with its drawbacks and advantages, the different security definitions and the evolution of IBE in literature. This is followed by an elaborate discussion on broadcast encryption (BE) and secret sharing. Finally, distributed key generation is described as a possible solution to the inherent key escrow problem of IBE.

3.1 Public Key Infrastructures

Asymmetric cryptography assigns a key pair $\langle sk_A, pk_A \rangle$ to every user A to allow secure communication between parties who never met. The infrastructure authenticating all public key values is called the Public Key Infrastructure (PKI) (Definition 2.34). However, PKI systems only shift the problem from trusting the users to trusting their keys. For example, if Eve could make the PKI system believe that her own public key pk_{Eve} actually represents the public key of Alice pk_{Alice}, Eve would be able to read all Alice's confidential communication as she obviously has the private key sk_{Eve} corresponding to pk_{Eve}. Therefore, it is important that public key systems rely on an architecture that authenticates whether key pairs belong to the claimed owner. In practice this is mostly achieved with the help of certification authorities or a web of trust.

3.1.1 Certification Authorities

In a traditional PKI system, all entities in the system trust a central party called the *Certification Authority* (CA). It is the CA that guarantees public keys belong to the claimed owner.

Suppose Alice wants to start using a key pair $\langle pk_A, sk_A \rangle$. She has to authenticate herself with the CA by correctly following a protocol that confirms Alice's identity, usually over offline channels. Once Alice is authenticated with the CA, Alice sends

the public key pk_A to the CA along with a proof showing that Alice also owns the corresponding private key sk_A. This "proof of correct possession" often takes the form of a signature $S_{sk_A}(pk_A)$ generated by the private key sk_A on the public key pk_A.

Once the CA is convinced of the authenticity of Alice's public key, it distributes a certificate approving that pk_A effectively belongs to Alice. To avoid forged certificates, the CA signs Alice's certificate with its private key sk_{CA}. Anyone doubting the authenticity of the public key pk_A can get convinced pk_A effectively belongs to Alice by checking the signature of the CA with the CA's public key pk_{CA}.

In practice, CAs often approve the trustworthiness of other CAs by issuing certificates on their signing keys. In this way, often highly complex hierarchical architectures are achieved that boil down to the trust in one signing key of the highest authority. This puts heavy requirements on the CA's infrastructure as a compromised CA signing key can break the system completely. Indeed, a compromised signing key would allow to sign certificates of unauthenticated public keys or even certificates of public keys that belong to malicious entities.

If an entity's private key is lost or leaked to a third party, it can be revoked by the CA. CAs achieve this by periodic publication of *revocation lists*. These revocation lists contain all compromised public keys. Consequently, users relying on a PKI should always verify these continuously growing lists before trusting a keypair. Thereby, revocation lists not only make the system less transparent, they also impose high demands on the infrastructure of entities relying on the PKI.

To partially get around the issue of revocation lists, certificates contain an expiration date. After expiration, a certificate should no longer be trusted. However, this requires keypair owners to contact CAs more frequently to sign new certificates each time the previous one has expired. Clearly, this puts a high computational demand on the authentication procedure of the CAs as well.

3.1.2 OpenPGP and Web of Trust

An alternative to the traditional PKI setting relying on CAs is a *web of trust*. In a web of trust any entity can rate the trustworthiness of a public key. For example, if Bob receives Alice's public key personally during a date, the public key can be considered more trustworthy than when Bob receives Alice's key via e-mail. Web of trust systems allow users to vet for the authenticity other users' keys in the system. A standardised web of trust system is OpenPGP [33].

The major advantage of a web of trust is that there no longer needs to be a CA with highly secure infrastructure as the publication of certificates now becomes a shared responsibility.

The system also has its drawbacks. Usability studies already have shown that non tech-savvy users have problems using PGP systems [108]. Furthermore, users are now required to judge for themselves whether they can trust a public key or not. This gives more responsibility to users than most of them can handle without proper knowledge of the consequences to their actions.

3.2 Identity-Based Encryption

The concept of identity-based cryptography was proposed by Shamir [103] 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) [9].

3.2.1 Definition

A generic Identity-Based Encryption (IBE) scheme is composed of four probabilistic polynomial time algorithms [27]:

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

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

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

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

Figure 3.1 illustrates these generic algorithms. A trusted Public Key Generator (PKG) generates a master private key msk and public parameters $params$ on input of the security parameter λ. Next, the PKG publishes the public parameters $params$ while storing msk preferably in encrypted format on a local disk. If Alice wants to send a message m to Bob, it suffices for her to know the public parameters $params$ and the id $\mathtt{id}_{Bob}$, uniquely identifying Bob. Then, Alice encrypts the message to a ciphertext c that is sent over an insecure channel to Bob. On receipt of the ciphertext, Bob authenticates to the PKG over a secure channel to request his private key $sk_{\mathtt{id}_{Bob}}$. Subsequently, the PKG generates the private key $sk_{\mathtt{id}_{Bob}}$ corresponding to Bob's identity $\mathtt{id}_{Bob}$ on input of the master secret key msk, Bob's id $\mathtt{id}_{Bob}$ and public parameters $params$. Subsequently, the PKG sends $sk_{\mathtt{id}_{Bob}}$ back again over a secure channel. Bob has now all the required information to decrypt the ciphertext c to its original plaintext message m.

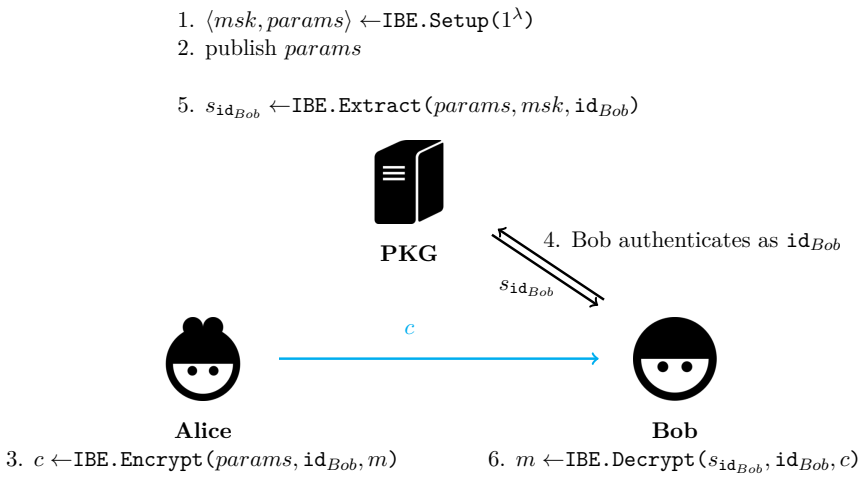

Figure 3.1: Generic identity-based encryption scheme. The blue arrow denotes an insecure channel that can be eavesdropped.

3.2.2 Comparison of IBE and PKI Schemes

Now we turn to overview the main advantages and disadvantages of generic IBE schemes when compared to more tradional PKI systems.

Disadvantages

Single point of failure The PKG generates every private key $sk_{\mathtt{id}}$ in the system thereby creating a single point of failure. If a PKG disconnects due to an excessive amount of extraction requests, new users can no longer receive their private keys. However, users already owning a secret key can continue decrypting ciphertexts since this requires no additional communication with the PKG. A PKI system often consists of a hierarchy of CAs. Consequently, an offline CA only takes down a specific part of the PKI. However, an offline CA can no longer issue certificates either.

Key escrow The PKG is required to be trusted since it learns $sk_{\mathtt{id}}$ for every entity in the system. A malicious PKG server could use this information to start eavesdropping on the insecure channel between Alice and Bob (the blue arrow in Figure 3.1) while decrypting all ciphertexts that are being sent over. The undesired property that private keys have to be shared with a trusted third party is often called *key escrow* in literature [6]. Since traditional PKIs only authenticate key pairs, key escrow is not an issue.

Revocation of public keys Generic IBE schemes do not support revocation of public keys. However, Bob's private key $sk_{\mathtt{id}_{Bob}}$ can still get compromised if he is careless with its storage. In fact, the research community has been focused on the revocation of IBE keys extensively [22, 26, 70, 81]. Key revocation often requires additional infrastructure that complicates the elegancy of the currently proposed

IBE scheme. The major drawback of revoking Bobs key is that Bob can no longer receive encrypted messages because his public key is part of his identity. Therefore, a pragmatic solution to this issue could be to append expiration dates to the public keys. Consequently, public keys will only be valid for a limited amount of time thereby restricting the damage that could be done with a compromised private key [27]. On the other hand, traditional PKIs publish revocation lists. Although these revocation list are a burden to the complexity of the PKI's infrastructure, they support a necessary feature for practical key management systems.

Advantages

Complexity of the system Only one PKG suffices to realise the system, which relaxes expensive infrastructure requirements on the system. Due to the support of revocation lists and the often hierarchical organisation, PKI systems are complex structures with a high amount of redundancy.

User friendlyness Users who have no background on cryptographic primitives no longer have to make conscious decisions on key lengths or the randomness of their keys. In an IBE system a public key is a string available to anyone in the system. This is generally conceived as more transparent to users lacking a background on cryptography. An average user knows what a username or an e-mail address represents and to whom it belongs while an authenticated public key is generally not a familiar concept.

Opt-in by default Another useful property of an IBE scheme is that a recipient is not required to actively subscribe to a hierarchy of CAs neither a web of trust before a sender can start sending him messages. In this way, the possibility to send encrypted messages becomes inherently part of any system in which the users are assigned unique identifiers. This is particularly useful in systems where the majority of the users has no knowledge about cryptographic primitives. Users do no longer need to generate a key pair neither subscribe to a third party infrastructure. It suffices to recall how their connections can be uniquely identified in the system to learn their public keys.

3.2.3 Security of IBE

IBE schemes follow similar security notions as generic public key systems. Therefore, definitions of security are often subtle as different levels of security can be distinguished. In literature *indistinguishability under chosen plaintext attack* (IND-CPA) and *indistinguishability under chosen ciphertext attack* (IND-CCA) are considered. Anonymity of the encryption scheme is an additional property of the scheme that is often desired [17].

Note that both the notion of IND-CPA, IND-CCA and anonymity are only introduced in an informal way in this section to give a basic understanding of these concepts to the reader. For a more formal description of IND-CPA and IND-CCA, the reader is referred to [27], whereas for a more formal description of ciphertext anonymity the reader is referred to [5].

Indistinguishability Under Chosen Plaintext Attack

Indistinguishability under chosen plaintext attack (IND-CPA) is described by the negligible advantage an adversary has in trying to distinguish which of both given plaintext messages m_0 and m_1 generated a ciphertext c. It captures the notion of *semantic security*, i.e. that any ciphertext c should not give more information about the original plaintext m than any other random binary string of the same length.

IND-CPA is best defined with the help of a game that challenges the adversary. If the adversary has negligible advantage trying to win the IND-CPA game in Game 1, the IBE system is said to be IND-CPA secure.

Game 1 Generic IBE-IND-CPA Game [4]

Goal: An adversary is challenged by a game to check the IND-CPA security of an IBE scheme.

Result: This IBE-IND-CPA Game helps to define the concept of IND-CPA security for IBE schemes.

1. The challenger runs $\langle msk, params\rangle \leftarrow \texttt{IBE.Setup}(1^\lambda)$ and returns $params$ to the adversary.
2. The adversary can start querying an oracle $O_{Extract}(\texttt{id}_i)$ that returns a private key $sk_{\texttt{id}_i} \leftarrow \texttt{IBE.Extract}(params, msk, \texttt{id})$ corresponding to an adversary defined identity $\texttt{id}_i$.
3. The adversary picks two equal length plaintext messages m_0 and m_1 and an identity $\texttt{id}_{encrypt}$. The adversary honestly passes $\langle m_0, m_1, \texttt{id}_{encrypt}\rangle$ to the challenger.
4. The challenger picks a random bit b and executes $c \leftarrow \texttt{IBE.Encrypt}(params, \texttt{id}_{encrypt}, m_b)$. The challenger gives c to the adversary.
5. The adversary continues querying the oracle $O_{Extract}(\texttt{id}_i)$ adaptively.
6. The adversary outputs a bit b' based on the ciphertext c. If $b = b'$ the adversary wins the game. If $b \neq b'$ or if the adversary queried the oracle $O_{Extract}(\texttt{id}_i)$ with $\texttt{id}_i = \texttt{id}_{encrypt}$ during step 2 or step 5, the adversary loses the game.

Indistinguishability Under Chosen Ciphertext Attack

Indistinguishability under chosen ciphertext (IND-CCA) is a more demanding level of security. Therefore, an algorithm that is IND-CCA secure is considered more secure than an IND-CPA secure algorithm. IND-CCA security means that an adversary has no advantage in trying to distinguish which of both given plaintext messages m_0 and

m_1 generated a ciphertext c even if the adversary has access to a list of (plaintext, ciphertext)-tuples.

IND-CCA is defined with the help of a game that challenges an adversary similar to the IND-CPA game. Compared to the IND-CPA game, the IND-CCA game contains two additional steps in which the adversary gets access to another oracle. If the adversary has negligible advantage trying to win the IND-CCA game from Game 2, the IBE system is said to be IND-CCA secure.

In literature a distinction is often made between a *non-adaptive* case (IND-CCA1) and an *adaptive* case (IND-CCA2) of IND-CCA. In the non-adaptive case, step 6 from Game 2 is not allowed. More precisely, an IBE scheme that satisfies Game 2 is said to be IND-CCA2 secure.

Anonymous Identity-Based Encryption

An IBE scheme is called anonymous (ANO-IBE) when the ciphertext does not leak the identity of the recipient. In the overview illustrated in Figure 3.1, this implies that no eavesdropper on the insecure channel between Alice and Bob could derive that Bob is the recipient based on the information in the ciphertext c alone [31].

ANO-IBE is defined with the help of a game that challenges an adversary similar to the IND-CPA game. If the adversary has negligible advantage trying to win the ANO-IBE game in Game 3, the IBE system is said to be anonymous.

Gentry [60] presents the first scheme which combines the notions of IND-CPA and IND-CCA with ANO-IBE. Therefore, system is then said to be IND-ANO-CPA secure or IND-ANO-CCA secure if it satisfies a modified version of the game in Game 3. For a more detailed discussion on the topic the reader is referred to the original paper [60].

3.2.4 Overview

Although Shamir [103] 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. Boneh and Franklin [27] proposed the first practically usable IBE scheme based on the Weil pairing, however, the security proof still relies on the random oracle assumption. At the same time, Sakai and Kasahara [95] proposed a different IBE scheme independently from Boneh and Franklin. The scheme from Sakai and Kasahara initially received less attention though, because the original presentation is in Japanese and lacking a security proof. Subsequently, Sakai and Kasahara [100] proposed an extended version of their original scheme which is proven to be IND-CCA secure in the random oracle model by Chen et al. [38]

Canetti et al. [35] introduced the first secure IBE scheme without relying on the random oracle model. Nevertheless, the attacker model in [35] requires the adversary to declare upfront which identity `id` is targeted during step 5 of the CCA Game (Algorithm 2) and step 4 of the CPA Game. Therefore, the scheme by Boneh and Franklin [27] is considered more secure as attackers can adaptively choose the

Game 2 Generic IBE-IND-CCA Game [4]

Goal: An adversary is challenged by a game to check the IND-CCA security of an IBE scheme.

Result: This IBE-IND-CCA Game helps to define the concept of IND-CPA security for IBE schemes.

1. The challenger runs $\langle msk, params \rangle \leftarrow \texttt{IBE.Setup}(1^\lambda)$ and returns $params$ to the adversary.
2. The adversary can start querying an oracle $O_{Extract}(\texttt{id}_i)$ that returns a private key $sk_{\texttt{id}_i} \leftarrow \texttt{IBE.Extract}(params, msk, \texttt{id})$ corresponding to an adversary defined identity $\texttt{id}_i$.
3. The adversary can start querying another oracle $O_{Decrypt}(sk_{\texttt{id}_i}, c_j)$ that returns a plaintext $m_j \leftarrow \texttt{IBE.Decrypt}(sk_{\texttt{id}_i}, c_j)$ corresponding to an adversary defined ciphertext c_j and identity $\texttt{id}_i$.
4. The adversary picks two equal length plaintext messages m_0 and m_1 and an identity $\texttt{id}_{encrypt}$. The adversary honestly passes $\langle m_0, m_1, \texttt{id}_{encrypt} \rangle$ to the challenger.
5. The challenger picks a random bit b and executes $c \leftarrow \texttt{IBE.Encrypt}(params, \texttt{id}, m_b)$. The challenger gives c to the adversary.
6. The adversary continues querying the oracle $O_{Extract}(\texttt{id}_i)$ adaptively.
7. The adversary continues querying the oracle $O_{Decrypt}(sk_{\texttt{id}_i}, c_j)$ adaptively.
8. The adversary outputs a bit b' based on the ciphertext c. If $b = b'$ the adversary wins the game. Otherwise, the adversary loses the game. If the adversary queried the oracle $O_{Extract}(\texttt{id}_i)$ with $\texttt{id}_i = \texttt{id}_{encrypt}$ during step 2 or step 6 or if the adversary queried the oracle $O_{Decrypt}(sk_{\texttt{id}_i}, c_j)$ with $c_j = c$ during step 3 or step 7, the adversary loses the game as well.

Game 3 Generic ANO-IBE Game [4]

Goal: An adversary is challenged by a game to check the ANO-IBE security of an IBE scheme.

Result: This ANO-IBE Game helps to define the concept of ANO-IBE security for IBE schemes.

1. The challenger runs $\langle msk, params \rangle \leftarrow \texttt{IBE.Setup}(1^\lambda)$ and returns $params$ to the adversary.
2. The adversary can start querying an oracle $O_{Extract}(\texttt{id}_i)$ that returns a private key $sk_{\texttt{id}_i} \leftarrow \texttt{IBE.Extract}(params, msk, \texttt{id}_i)$ corresponding to an adversary defined identity $\texttt{id}_i$.
3. The adversary picks a plaintext message m and an identity $\texttt{id}_{encrypt}$. The adversary honestly passes $\langle m, \texttt{id}_{encrypt} \rangle$ to the challenger.
4. The challenger picks a random bit b and computes $c \leftarrow \texttt{IBE.Encrypt}(params, \texttt{id}_{encrypt}, m)$ if $b = 0$. If $b = 1$, the challenger computes $c \leftarrow \texttt{IBE.Encrypt}(params, \texttt{id}_{encrypt}, r)$ where r is a random bit sequence with the same length as the message m. The challenger gives c to the adversary.
5. The adversary continues querying the oracle $O_{Extract}(\texttt{id}_i)$ adaptively.
6. The adversary outputs a bit b' based on the ciphertext c. If $b = b'$ the adversary wins the game. If $b \neq b'$ or if the adversary queried the oracle $O_{Extract}(\texttt{id}_i)$ with $\texttt{id}_i = \texttt{id}_{encrypt}$ during step 2 or step 5, the adversary loses the game.

targeted identity. Later, Boneh and Boyen [24] presented a variant to [35] which also realises only selective ID security.

Waters [106] is the first to present a scheme that is IND-CCA secure in the standard model. Drawback of the scheme from Waters [106] is that it requires large public parameters. Gentry [60] proposes a more efficient alternative to this scheme in the standard model while achieving shorter public parameters. However, the scheme from Gentry relies on a complicated hardness assumption called q-BDHE. It is only after the introduction of the Dual System paradigm by Waters [107] in 2009 that IND-CCA security can be achieved in the standard model based on reasonable assumptions. De Caro et al. [37] are the first to define an IND-ANO-CCA secure IBE scheme on the Dual System construction of Waters [107].

Although all these contributions were a step forward in the evolution of IBE, not all of these schemes are ANO-IBE. Most IBE systems in the random oracle model can be proven anonymous. Therefore, the IBE scheme from Boneh and Franklin [27] is IND-ANO-CCA secure. In the standard model, it appeared to be harder to construct

ANO-IBE schemes at first sight, e.g. it can be proven that the scheme from Boneh and Boyen [24] is not anonymous in its original form. The scheme from Gentry [60] was the first anonymous IBE scheme in the standard model. Boyen and Waters [31] published almost synchronously another IBE scheme in the standard model that is also IND-ANO-CCA secure. In 2010, Ducas [48] showed that even schemes that were first considered not anonymous like the one from Boneh and Boyen [24] but also [25, 106] can be proven anonymous when relying on asymmetric pairings thereby making anonymity a more common property in IBE schemes.

3.2.5 Most Attractive IBE Schemes

In the standard model mainly the anonymous IBE constructions from Gentry [60] and De Caro et al. [37] have the most satisfying properties. However, IBE constructions in the standard model often come at the cost of higher computational requirements [30]. Certainly the scheme from De Caro demands a higher amount of computational resources since it relies on composite order groups. Although methods [55, 78] have been developed to convert IBE schemes from composite order groups to single order prime groups, these methods do not apply to the scheme from De Caro et al. [77]

From all schemes discussed in Section 3.2.4 the ones initially developed by Boneh and Franklin [27] and Sakai and Kasahara [100] are the most attractive ones in the random oracle model because of their anonymity and non-selective security. Consequently, it is not a coincidence that both schemes have found description in an informational RFC document. Sakai and Kasahara IBE is described in RFC 6508 [67] and RFC 6509 [66]. Boneh and Franklin IBE can be found in RFC 5409 [84].

Because the ANO-IND-CPA secure scheme and the ANO-IND-CCA secure scheme from Boneh an Franklin [27] are important for the remainder of this text, they are both included in Algorithm 4 and Algorithm 5 respectively.

3.3 Broadcast Encryption

Another relevant aspect of encryption in OSNs is how one encrypted message can be securely broadcasted to multiple users. Broadcast encryption (BE) was introduced by Fiat and Naor [52], as a public-key generalisation to a multi user setting. A BE scheme allows a user to encrypt a message m to a subset $\mathcal{S}$ of users in a public key system, such that, only users in the set $\mathcal{S}$ are able to decrypt the message. The computational overhead of BE is generally bound to the ciphertext and the number of recipients.

3.3.1 Definition

A generic Broadcast Encryption (BE) scheme is composed of four probabilistic polynomial time algorithms:

`BE.Setup(`1^λ`)` : On input of a security parameter λ, generates the public parameters *params* of the system.

Algorithm 4 IND-ANO-CPA Boneh and Franklin IBE [27]

Goal: Alice wants to send an IBE encrypted message to Bob.

Result: Alice sends an IBE encrypted ciphertext c that is successfully decrypted by Bob.

1. `Setup`(1^λ): Let λ be the security parameter for a security level of l bits.
   a) Execute setup algorithm $\langle q, G_1, G_2, e : G_1 \times G_2 \to G_T, P \in G_1 \rangle \leftarrow \mathcal{G}\left(1^\lambda\right)$ to generate the parameters
      i. A large prime q
      ii. Gap groups G_1 and G_2 of order q
      iii. An admissible bilinear map $e : G_1 \times G_2 \to G_T$
      iv. A random generator $P \in G_1$
   b) Choose a uniformly random $msk \in \mathbb{Z}_q^*$ and calculate
   $$P_{pub} = mskP$$
   c) Choose cryptographic hash functions
      i. $H_1 : \{0,1\}^* \to G_1$
      ii. $H_2 : G_2 \to \{0,1\}^l$
2. `Extract`($params, msk,$ `id`):
   a) Compute $Q_{\texttt{id}} = H_1(\texttt{id}) \in G_1$
   b) Set the private key of `id` to $sk_{\texttt{id}} = mskQ_{\texttt{id}}$
3. `Encrypt`($params,$ `id`$, m$):
   a) Compute $Q_{\texttt{id}} = H_1(\texttt{id})$
   b) Choose a random $r \in Z_q$
   c) Encrypt the plaintext message m to the ciphertext c as
   $$c = \langle rP, m \oplus H_2\left(g_{\texttt{id}^r}\right)\rangle = \langle U, v\rangle \quad \text{with} \quad g_{\texttt{id}} = e\left(Q_{\texttt{id}}, P_{pub}\right) \in G_T$$
4. `Decrypt`($sk_{\texttt{id}}, c$): Decrypt the ciphertext c back to the plaintext message m as
   $$m = v \oplus H_2\left(e\left(sk_{\texttt{id}}, U\right)\right)$$

Algorithm 5 IND-ANO-CCA Boneh and Franklin IBE [27]

Goal: Alice wants to send an IBE encrypted message to Bob.

Result: Alice sends an IBE encrypted ciphertext c that is successfully decrypted by Bob.

1. `Setup`(1^λ):
   a) As in Algorithm 4
   b) As in Algorithm 4
   c) Choose cryptographic hash functions
      i. $H_1 : \{0,1\}^* \rightarrow G_1$
      ii. $H_2 : G_2 \rightarrow \{0,1\}^l$
      iii. $H_3 : \{0,1\}^l \rightarrow (0,1)^l$
2. `Extract`$(params, msk, \texttt{id})$: As in Algorithm 4
3. `Encrypt`$(params, \texttt{id}, m)$:
   a) Compute $Q_{\texttt{id}} = H_1\{\texttt{id}\}$
   b) Choose a random $sigma \in (0,1)^l$
   c) Compute $r = H_3\{sigma, m\}$
   d) Encrypt the plaintext message m to the ciphertext c as

$$c = \langle rP, sigma \oplus H_2\left(g_{\texttt{id}}^r\right), m \oplus H_3\left(sigma\right)\rangle = \langle U, v, w\rangle$$
$$\text{with } \; g_{\texttt{id}} = e\left(Q_{\texttt{id}}, P_pub\right) \in G_T$$

4. `Decrypt`$(sk_{\texttt{id}}, c)$: Decrypt the ciphertext c back to the plaintext message m as follows
   a) Compute $sigma = v \oplus H_2\left(e\left(sk_{\texttt{id}}, U\right)\right)$
   b) Compute $m = w \oplus H_3\left(sigma\right)$
   c) Set $r = H_3\left(sigma, m\right)$. Test that $U = rP$. If not, reject the ciphertext.
   d) Output m as the decryption of c

BE.KeyGen(*params*) : Returns the public and private key (pk_i, sk_i) for each user i while taking the public parameters *params* into account.

BE.Encrypt($m, \mathcal{S}$) : Takes a set of public key values $\mathcal{S} = \{pk_i \dots pk_{|\mathcal{S}|}\}$ corresponding to users i in the system along with a plaintext message m to generate a corresponding ciphertext c.

BE.Decrypt(c, sk_i): Reconstructs m from c using the private key sk_i if the corresponding public key $pk_i \in \mathcal{S}$. Otherwise, return $\perp$.

Note that this definition is stated generically enough to allow all kinds of public keys to be used. Therefore, not only traditional PKIs can benefit from BE schemes, but also IBE schemes in which a public identifier $\texttt{id}_i$ serves as a public key pk_i.

3.3.2 Overview

The problem of BE has been widely studied in literature since its first introduction by Fiat and Naor [52]. This section highlights the most important evolutions of BE in literature. The summary that follows is far from complete as it only considers publications that are relevant to our final goal: achieving user-friendly broadcast encryption for OSNs.

Broadcast Encryption

The implementation from Fiat and Naor [52] requires a ciphertext of size $O\left(t \log^2 t \log n\right)$ to be secure against t colluding users. The first fully collusion resistant scheme was proposed in [91] by Naor et al. thereby making the ciphertext size independent of the number of colluding users. A collusion resistant BE scheme refers to a broadcast encryption scheme that is secure even if all users that are not in the recipient set $\mathcal{S}$ would collaborate. Halevy and Shamir further reduce the required ciphertext length for collusion resistant schemes in [69]. It is the first paper in a series of many [47, 65, 79] that achieves ciphertext sizes only dependent on the number of revoked users $O(r)$. Boneh, Gentry and Waters [25] are the first to consider utilisation of bilinear maps to realise constant size ciphertexts and $O(n)$ public keys.

Identity-Based Broadcast Encryption

Sakai and Furukawa are the first to define a collusion resistant identity based broadcast encryption (IBBE) scheme in [99]. Independently from [99] Delerablée realises a similar IBBE scheme and claims to be the first as well in [45]. The size of the public key in both [99] and [45] is proportional to the maximum size of the intended set of recipients while realising short ciphertexts and private keys.

Baek et al. [9] define an IBBE scheme that requires only one pairing computation. The scheme in [9] is proven secure under the random oracle assumption where the attacker ties himself to a selective-ID attack. Gentry and Waters achieve identity based broadcast encryption with sublinear ciphertexts in [61]. Their scheme is proven

secure against a stronger notion of adaptive security where the attacker can adaptively alter its queries depending on earlier received information. Barbosa and Farshim [11] proposed an identity-based key encapsulation scheme for multiple parties which is an extension of *mKEM* as considered by Smart [105] to the identity-based setting. An mKEM is a Key Encapsulation Mechanism which takes multiple public keys as input. An encrypted message under mKEM consists of an encapsulated session key k and a symmetric encryption $E_k(m)$ of the plaintext message m under k. However, the scheme from Smart [105] is only proven secure under the random oracle assumption.

Anonymous Broadcast Encryption

All earlier mentioned references describing BE require the intended set of recipients to be published to realise higher efficiency. Barth, Boneh and Waters [14] are the first to design a BE scheme that takes the anonymity of the recipient into account.

Definition 3.1 (Anonymity). A BE scheme is said to be *anonymous* if it hides who is included in the recipient set.

The proposed anonymous broadcast encryption (ANOBE) scheme from Barth, Boneh and Waters [14] implies a linear dependency of the ciphertext on the number of recipients and can only be proven secure in the random oracle model. In [80] Libert et al., propose an alternative ANOBE scheme that is proven secure in the standard model. Both [14] and [80] propose a tag based system that allows efficient decryption at the cost of making the public master key linear dependent on the total number of users. Krzywiekci et al. [76] propose a scheme that is proportional to the number of revoked users, although the security proof is rather informal. In [110], Yu et al. design an architecture that even hides the number of users in the recipient set using Attribute Based Encryption (ABE) [98]. [50]

However, ABE requires that all users are assigned attributes such that all users who have sufficient attributes in common can decrypt the message. In networks where the total number of users is large it can be a work intensive task to label each user with the correct attributes.

Outsider-Anonymous Broadcast Encryption

Fazio and Perera introduce the notion of outsider anonymous broadcast encryption in [50]. The scheme relies on IBE to encode where a recipient is positioned in a publicly published tree to achieve sublinear ciphertexts. It is remarkable that sublinear ciphertexts are achieved while attaining recipient anonymity to all users that are outside the intended set of receivers. However, the scheme has the drawback of immediately fixing the total number of users that are allowed in the system. Furthermore, an additional architecture is required to maintain the tree of subscribed users. Finally, although IBE is used, the scheme does not allow to represent public keys of users by their public identifiers because the public key needs to be the position of a user in the tree structure of the external architecture. In this way, most of the desirable properties of IBE cancel out.

Although the scheme from Fazio and Perera does not fit the requirements for user-friendly broadcast-encryption in OSNs, it is useful to remember their definition of outsider-anonymity.

Definition 3.2 (Outsider Anonymity). A BE scheme is called *outsider anonymous* if the identities of the recipients are known to the other identities in the recipient set $\mathcal{S}$ while remaining secret to other parties of the BE scheme.

3.3.3 Most Attractive BE Schemes

From all schemes discussed in Section 3.2.4 mainly the scheme from Libert et al. [81] has the most attractive properties as it is proven secure in the standard model at almost no reduced computational efficiency. The scheme supports anonymity in both identity-based BE as well as traditional asymmetric cryptosystems.

If anonymity is not an issue, different BE schemes have to be considered depending on the goals of the target application. The scheme from Libert et al. [80] will certainly not have the most desirable properties in non-anonymous BE environments since it can not benefit from higher efficiency due to the recipient being publicly known.

3.4 Secret Sharing

IBE mainly profits from a less complex architecture and increased ease of usability. Nevertheless, the major drawback of IBE seems to be the inherent key escrow property. In order to get around this issue, distributed key generation seems a promising solution. However, before diving into distributed key generation protocols, some knowledge on secret sharing is appropriate.

3.4.1 Definition

Definition 3.3 (Secret Sharing Scheme). A *Secret Sharing Scheme* is a cryptographic scheme that divides a secret s into n pieces of data $\sigma_1, \ldots, \sigma_n$ called *shares*. Shares are distributed over n different parties called *shareholders* such that only specific subsets of the distributed shares allow reconstruction of the original secret s.

Definition 3.4 (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 σ_i to n different parties P_i for $1 \leq i \leq n$ such that any subset of t or more different shares σ_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 3.5 (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.

3.4.2 Shamir Secret Sharing

In 1979, both Shamir [102] and Blakley [21] independently proposed 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 6 Shamir's (t, n) threshold scheme [90]

Goal: A dealer D 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* A dealer D begins with a secret integer $s \geq 0$ it wishes to distribute among n parties
   a) D chooses a prime $p > \max(s, n)$ and defines $a_0 = s$
   b) D selects $t-1$ random, independent coefficients $a_1, \ldots, a_{t-1}, 0 \leq a_j \leq p-1$ defining the random polynomial over $\mathbb{Z}_p$, $f(x) = \sum_{j=0}^{t-1} a_j x^j$
   c) D computes $\sigma_i = f(i) \bmod p, 1 \leq i \leq n$ and securely transfers the share σ_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 $(x, y) = (i, \sigma_i)$ 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

$$f(0) = \sum_{i=1}^{t} y_i b_i = s \quad \text{with} \quad b_i = \prod_{1 \leq j \leq t, j \neq i} \frac{j}{j-i}$$

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 6. The mechanism behind

reconstruction in Algorithm 6 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

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

A proof of this formula is omitted but can be found in [109].

3.4.3 Verifiable Secret Sharing

Verifiable secret sharing [40] 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 [51] and Benaloh's scheme [19]. No further details are given as a basic notion of verifiable secret sharing suffices for the remainder of this text.

3.5 Distributed Key Generation

Distributed key generation is inspired on secret sharing. The idea behind distributed key generation is that a secret s can be shared among n shareholders without the requirement for a centralised dealer D as in Algorithm 6. In this way, a secret can be negotiated between all shareholders without any of the shareholders explicitly computing the secret. The major advantage of such a scheme is that no party in the scheme requires a higher level of trust since no party explicitly knows the secret. Similarly to the Shamir secret sharing scheme a group of t or more shareholders will need to pool their shares in order to reconstruct the secret s.

3.5.1 Definition

Definition 3.6 (Distributed key generation scheme). A *distributed key generation scheme* is a (t, n) perfect threshold scheme $(t \leq n)$ that requires no trusted party. That is, a distributed key generation scheme is a cryptographic scheme that negotiates a secret s with n different parties $P_1, \ldots, P_n$ by letting each party P_i distribute shares σ_{ij} of its own private secret σ_i with all other parties P_i where $1 \leq i \leq n, 1 \leq j \leq n$. At least t out of n parties will need to collude in order to compute the original secret s explicitly.

3.5.2 Pedersen Distributed Key Generation

The first usable distributed key generation protocol was defined by Pedersen [93]. A later publication from Gennaro et al. [59] proves the Pedersen scheme to be insecure in its original form in the presence of malicious key generation centers.

Although the Pedersen scheme [93] is proven insecure, it is most instructive to describe the protocol in its original form as later schemes like the one from Gennaro [59] extensively rely on the same concepts. Therefore, the original Pedersen protocol is shown in Algorithm 7. TODO for Algorithm 8: Complete it after fully understanding it

Algorithm 7 Pedersen's distributed key generation [93]

Goal: A secret s is negotiated with n uniquely numbered parties $\{P_1, \ldots, P_n\}$ without any of the parties explicitly computing the secret s.

Result: If a subset of at least t out of n parties colludes, they can reconstruct the original secret s.

1. *Setup* At initialisation, a setup algorithm $\langle p, g\rangle \leftarrow \mathcal{G}\left(1^{\lambda}\right)$ is executed that returns a large prime number p and a generator g of $\mathbb{Z}_p$ on input of a security parameter λ. After execution of $\mathcal{G}\left(1^{\lambda}\right)$ each party $P_i, 1 \leq i \leq n$ should do the following:

   a) P_i generates a random private key $sk_i \in \mathbb{Z}_p$ and publishes the corresponding public key $pk_i = g^{sk_i}$

   b) P_i chooses $t-1$ random independent coefficients $a_{i,1}, \ldots, a_{i,t-1}, 0 \leq a_{i,j} \leq p-1$ defining a random polynomial $f_i(x)$ over $\mathbb{Z}_p$, $f_i(x) = \sum_{b=0}^{t-1} a_{i,j} x^b$.

   c) P_i commits to the coefficients $a_{i,1}, \ldots, a_{i,t-1}, 0 \leq a_{i,j} \leq p-1$ by broadcasting $A_{ib} = g^{a_{i,b}} \bmod p$ for $b = 1, \ldots, t$ to all other parties.

   d) P_i computes the share $\sigma_{ij} = f_i(j) \bmod p$ and securely transfers the share σ_{ij} to party P_j along with a signature $S_{P_i}(\sigma_{ij})$ authenticating the share. P_i keeps σ_{ii} to itself.

   e) P_i verifies for each share σ_{ji} received from P_j whether it is consistent by verifying that

   $$g^{\sigma_{ji}} = \prod_{b=0}^{n-1} \left(A_{jb}\right)^{i^k} \bmod p$$

   If the check fails for an index j, P_i broadcasts a complaint against P_j along with the received share σ_{ij} and its signature $S_{P_j}(\sigma_{ij})$. If a party receives t complaints, he is excluded from the set of participating parties $\mathcal{Q}$.

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

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

Chapter

4 Design of a Practical Encryption Scheme for Online Social Networks

In the first section of this chapter (Section 4.1), a model is developed that allows to describe the current OSN situation. By defining every considered entity in the OSN, the resulting model serves as the framework in which we design our future constructions. In a next step (Section 4.2), the different threats within the model are defined. The section on the threat model first highlights the current privacy threats, followed by a definition of the current adversaries and the assumptions on these adversaries. In Section 4.3 our proposal is derived by stating cryptographic goals based on the earlier threat model. This is followed by design decisions on how to achieve these goals and how this impacts our model. Section 4.3 is concluded with a concrete proposal in the form of an algorithm along with an evaluation section motivating why our cryptographic design goals are successfully met.

4.1 Model of the Current Situation

The most commonly accepted definition of an *Online Social Network* (OSN) in literature is from Boyd and Ellison [29]. However, since this definition is still too generic for the remainder of this text a slightly modified version is presented here.

Definition 4.1 (Online Social Network [29]). An *online social network* (OSN) is a web-based service that allows individuals to:

1. Construct a public or semi-public profile within a bounded system
2. Articulate a list of other users with whom they share a connection.
3. View, traverse and share content with their list of connections and those made by others within the system

For the definition of our OSN model we focus on messages sent by users. Therefore, we define several different entities that are present in our model.

Definition 4.2 (OSN user). An *OSN user* U is any entity that has a profile on the OSN and thus identifiable by a unique identifier $\mathtt{id}_U$. The set containing all users of an OSN is denoted $\mathcal{U}$.

An OSN user can perform different activities within the infrastructure of the OSN. Depending on the performed activity, the user is labeled as one of three different roles: a sender, a friend or an intended recipient.

Definition 4.3 (Sender). A *sender* A is an OSN user who broadcasts a message m over the OSN infrastructure to varying subsets of OSN users, called the *intended recipient set* $\mathcal{S}$, such that $\mathcal{S} \subseteq \mathcal{U}$.

Definition 4.4 (Intended recipient). An *intended recipient* of a plaintext message m is an OSN user who is explicitly designated by a sender A to have access to the content of m.

Definition 4.5 (Friend). An OSN user who shares a connection with another OSN user U in the OSN infrastructure, is called a *friend of the user* U.

Different entities within the OSN have access rights to the profile $\texttt{id}_U$ of a user U. If abstraction is made of entities with access to only specific content, it suffices to define four different sets of entities, each with their own access rights: the set of the user's friends $\mathcal{F}_U$, the intended recipient set $\mathcal{S}$, the set of entities with access to the OSN $\mathcal{V}$ and the set of entities with access to the user's profile $\mathcal{V}_U$.

Definition 4.6 (Friends set). The set of all friends associated to a user U is the *friends set* $\mathcal{F}_U$, such that $\mathcal{F}_U \subseteq \mathcal{U}$.

Definition 4.7 (Intended recipient set). The *intended recipient set* of a message m is the set of all intended recipients of m. The intended recipient set $\mathcal{S}$ takes the form of a list of `id`'s uniquely identifying other users' profiles in the OSN infrastructure.

Definition 4.8 (Viewers set). Any entity that is given access to the OSN belongs to the *viewers set* $\mathcal{V}$.

Definition 4.9 (Profile viewers set). All viewers with access to non-public content of a profile $\texttt{id}_U$ of a user U are in the *profile viewers set* $\mathcal{V}_U \subseteq \mathcal{V}$.

Many different entities can be part of the set $\mathcal{V}$, e.g. OSN users, advertising companies, system administrators of the OSN or software applications specifically developed for the OSN. Usually, the OSN determines who is part of $\mathcal{V}$. Therefore, a user U often has no control in who is a member of $\mathcal{V}_U$.

As illustrated by Figure 4.1, Sender U wants to broadcast a message m over the OSN infrastructure to the intended recipient set $\mathcal{S}$. As U only wants to share the message with a specific group of friends, U defines the intended recipient set, such that $\mathcal{S} \subset \mathcal{F}_U$. Next, U sends m to the OSN's distribution server along with the intended recipient set $\mathcal{S}$. The OSN Server further distributes the message to all users in $\mathcal{S}$. Also a subset of third party applications and advertisers get access to the distributed message if they are inside the viewers group $\mathcal{V}_U$. Every entity who has access to the message is coloured blue in Figure 4.1.

Figure 4.1 illustrates previous definitions applied to an OSN as it is often encountered on the internet. The different sets in Figure 4.1 are defined as follows:

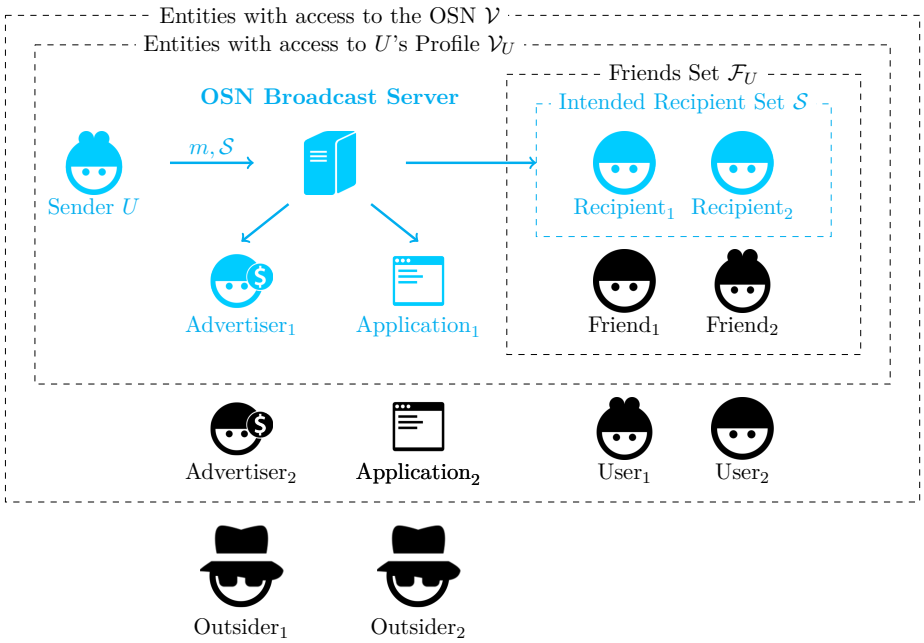

Figure 4.1: Model of the current OSN situation. Entities with access to the message m are coloured blue.

- The intended recipient set,

$$\mathcal{S} = \{\text{Recipient}_1, \text{Recipient}_2\}$$

- The set of friends of user U,

$$\mathcal{F}_U = \{\mathcal{S}, \text{Friend}_1, \text{Friend}_2\}$$

- The set of viewers who have access to the profile of user U,

$$\mathcal{V}_U = \{\mathcal{F}_U, \text{Sender } U, \text{Advertiser}_1, \text{Application}_1\}$$

- The set of entities with access to the OSN,

$$\mathcal{V} = \{\mathcal{V}_U, \text{User}_1, \text{User}_2, \text{Advertiser}_2, \text{Application}_2\}$$

- The set of all users in the OSN,

$$\mathcal{U} = \{\mathcal{F}_U, \text{Sender } U, \text{User}_1, \text{User}_2\}$$

The OSN's infrastructure stores almost everything within the viewer set $\mathcal{V}$. The profiles of all users within the friends set $\mathcal{F}_B$, the list of `id`'s within the intended

recipient set $\mathcal{S}$, access rights of applications and advertisers that are part of $\mathcal{V}_B$ and access rights of entities within the set $\mathcal{V}$ are all explicitly stored somewhere on the servers of the OSN.

Note that not all OSNs support the functionality to define intended recipient sets $\mathcal{S}$ on a per message basis. In OSNs like Twitter the standard privacy settings are such that message are always published publicly. Therefore, the model from Figure 4.1 only holds for a specific subset of OSNs like Facebook or Google+. More public OSNs like Twitter would require less sets of entities to model their behaviour.

It requires almost no additional effort to transform the model from Figure 4.1 such that it also takes the sharing of other media than messages into account. The model could then adopted for use on SNSs like Youtube or Instagram as well. However, this falls out of the scope of this thesis.

4.2 Threat Model

4.2.1 Privacy Threats

Currently an OSN as illustrated in Figure 4.1 presents several issues that can be classified as follows.

Misplaced Trust in the OSN Provider

Users have to trust the OSN provider, however this trust is often misplaced on several fronts.

Trust in the intended recipient set There is a mismatch between the expectations of Sender U and the functionality of the OSN. A user has higher demands on the specification of the intended recipient set than effectively implemented by the OSN. When a privacy-aware user like sender U takes the effort to define an intended recipient set $\mathcal{S}$, U expects to have full control on who has access to the messages m. In reality, sender U only has partial control since the OSN determines all other entities in $\mathcal{V}_U$ that are not part of sender U's friend list $\mathcal{F}_U$. In some OSNs a user first has to give permission to third party applications before access is granted to the user's content. Note that this gives more control to OSN users on determining who is inside the viewers set $\mathcal{V}_U$. Nevertheless, in practice it is still hard to get a concise overview from the OSN on everyone inside $\mathcal{V}_U$.

Trust in the broadcasting mechanism Any user broadcasting messages over the OSN infrastructure has to trust the OSN that it effectively operates as claimed. If the OSN broadcast server in Figure 4.1 would accidentally broadcast messages publicly despite of the sender's privacy settings, the privacy of the sender is breached. Even more worrisome is the fact that the sender will almost certainly never discover these privacy violations.

Trust in the data storage policy All messages are stored on the infrastructure provided by the OSN. However, has to trust the OSN will treat this data responsible. Nevertheless, most OSN data storage policies store all content for an unlimited

amount of time on their servers. The user thereby loses the control over his own data and to whom and how long it is still available.

Mismatch of Needs Besides the earlier mentioned issues, the OSN often operates with a corporate mentality. Although the user desires a relatively private OSN environment, the needs of the OSN are different. The OSN has no initiative to stop adding advertisers and applications to the set of entities with access to a user's profile $\mathcal{V}_B$. The more information advertising companies receive from the OSN provider, the better they can tailor advertisements to the user. The more third party applications rely on the OSNs infrastructure, the more appealing the OSN business model looks like. Therefore, OSNs have often not enough incentives to offer stricter access control policies to their users.

Dependency on the OSN's Privacy Infrastructure

Often, users are also dependent on the privacy infrastructure as it is currently provided by the OSN.

Dependency on accessibility of infrastructure Users of the OSN have to rely on the security of the OSN's infrastructure. If one of the outsiders in Figure 4.1 would succeed in hacking the OSN's digital infrastructure, he would have immediate access to all sensible information stored on the OSN's servers. Similarly, local governments can subpoena the OSN to disclose sensible information on certain users with the argument of national security.

Dependency on privacy preferences Another significant point is that the OSN fully determines which access policies are supported. Not all OSNs offer the definition of an intended recipient set on a per message basis. Even OSNs currently supporting this functionality can suddenly stop offering the service. Moreover, nothing prevents OSN providers from changing their privacy policy on a regular basis, thereby complicating users to define the access policy of their choice.

4.2.2 Adversaries

We consider adversaries that are honest but curious, i.e. passive adversaries that do not actively try to prevent the broadcasting process but are curious for the broadcasted content. An adversary in the earlier defined model, is any computationally bounded entity trying to violate one or several of the following properties:

1. **Confidentiality:** The entity tries to violate the confidentiality of encrypted messages, i.e. uncovering information within a broadcasted ciphertext. This can either be the intended recipient set $\mathcal{S}$ or the actual content of the plaintext message m.

2. **Integrity:** The entity tries to violate the integrity of encrypted messages, i.e. changing the ciphertext c or the plaintext m such that it differs from the way it was originally drafted by the sender.

3. **Availability:** Any entity apart from the original sender, tries to prevent messages from being broadcasted by bringing down parts of the architecture or the OSN.

4.2.3 Assumptions

Assumptions on the OSN

OSNs are assumed not to violate integrity neither availability. Nothing can be done to prevent the OSN from actively altering its own resources to bring down the proposed IBE architecture. It can not be prevented that a user of IBE on the OSN infrastructure gets blocked by the OSN provider. Neither can it be prevented that OSNs delete messages because they are encrypted. OSNs can also easily impersonate the owner of a profile in their infrastructure. Therefore, from this moment onwards, OSNs are assumed to not act as an active adversary. It is assumed that as soon as privacy aware users notice this kind of OSN behaviour, they will adopt to more reliable OSN alternatives. However, our assumptions do not prevent the OSN from trying to passively break confidentiality as they have every motivation for it in the context of their current business model.

Another assumption on the OSN's infrastructure is that the authentication mechanism of the OSN is secure. This is primarily important for the authenticity of the broadcasted messages on a user's profile as further discussed in Section 4.3.2.

Previous assumptions on the OSN are ideal assumptions that will probably hold as long as only the minority of the OSN users applies encryption mechanisms to their network. It is still unknown how OSNs will react if the proposed encryption mechanism finds common acceptance in their wide user base.

Furthermore, it is assumed that the OSN does not rely on traffic analysis to derive more information on the encrypted content. For example, suppose a user just visited a URL to an external news website broadcasted over the OSN infrastructure. With high probability the message broadcasted immediately after the user's visit, is a reaction to the content in the article. Furthermore, if Bob visits Alice's profile on a daily basis, almost certainly Bob regularly includes Alice as a recipient in his broadcasted messages as well.

Note that the latter assumption is not valid in practice. Generally, the attractiveness of traffic analysis is tempered by generating dummy traffic. However, protecting against traffic analysis falls out of the scope of this thesis.

Assumptions on the User

In order to achieve a strong encryption mechanism, users are unlimited in their abilities to behave as an adversary. However, one subtle assumption is made on users that are part of $\mathcal{S}$. Users in $\mathcal{S}$ are assumed not to break their social contract. That is, if a sender broadcasts a confidential message to a selected set of recipients, the recipients are assumed not to decrypt the encrypted message and rebroadcast the confidential content to any entity not in the original recipient set. In fact, no existing encryption mechanism provides protection against such misbehaviour, although

traitor tracing schemes [39] discourage users from treating by indicating who broke his social contract. However, for the remainder of this text intended recipients are assumed trustworthy in that they not break the social contract.

4.3 Our Proposal

4.3.1 Cryptographic Goals

The general design goals (Section 1.3) stated that a new privacy enhancing architecture for OSNs should be user friendly, applicable and immediately ready to use. Besides from these general design goals, it is now possible to define specific cryptographic requirements as well. A well-designed encryption scheme should be able to achieve the following cryptographic goals when publishing a message m to a set of intended recipients $\mathcal{S}$ on an OSN with the help of an encryption scheme:

- **Confidentiality:** The message is protected from disclosure to unauthorised parties, i.e. all entities that are not explicitly in the recipient set $\mathcal{S}$.
- **Outsider recipient anonymity:** The intended recipients of a broadcasted message should be anonymous to any entity not included in $\mathcal{S}$. This implies that neither the OSN has to know who the recipients are. (Definition 3.2 gives a more formal definition of outsider-anonymity).
- **No redundancy:** The message should be published only once to reach every recipient in $\mathcal{S}$.
- **Authenticity:** The recipients of the message have reasonable assurances of the message's origin.
- **Integrity:** The recipients are assured the message is distributed in its original form as posted by the sender.
- **No key escrow:** Private keys are only disclosed to the owners of the public key. No other entity should be able to have more information on one's secret key in the information theoretic sense.
- **Key validation:** All users of the system should be able to verify the correctness of their private keys.
- **Limited key validity:** Private keys of users should only be valid for a limited period of time to limit the damage of potentially lost private keys.

4.3.2 Design Decisions

In this section, our architecture is further developed by keeping the aforementioned cryptographic design goals in mind.

	Security Proof		
IBE Scheme	IND-ANO-CCA	Standard model	Assumption
Boneh and Franklin	✓	✗	BDH
Sakai and Kasahara	✓	✗	BDHI
Gentry	✓	✓	q-BDHE

Table 4.1: Security comparison of considered IBE schemes

Confidentiality

Confidentiality can be achieved by applying an encryption scheme before broadcasting a message. Current solutions like Scramble [16] and Persona [8] rely on rather classic public key infrastructures thereby requiring the OSN user to subscribe to a third party key infrastructure. These key infrastructures are required to authenticate and store the public keys of all security aware users. However, this does not correspond to the general design goals from Section 1.3 stating that the proposed solution should be both user friendly and immediately ready to use.

Identity-based encryption (IBE) can be used to achieve both confidentiality and the general design goals of usability and applicability. During the design of our scheme, three IBE schemes were considered as a potential candidate: Boneh and Franklin IBE [27], Sakai and Kasahara IBE [100] and Gentry IBE [60]. For a more elaborate discussion on why only these schemes were considered, the reader is referred to Section 3.3.2.

Table 4.1 lists the different security properties of all schemes. The Gentry IBE scheme has the highest security level since it is the only scheme proven secure in the standard model. In the random oracle model, Boneh and Franklin IBE is preferred over Sakai and Kasahara IBE since it relies on the BDH assumption which is more widely accepted than the stronger BDHI assumption.

The execution times of all considered IBE schemes are illustrated in Table 4.2. Experiments were conducted on an Intel Core 2.4 GHz i5 processor with 8 Gb of 1600 MHz DDR3L onboard memory. Pairing computations were implemented using the multi-precision MIRACL library [101]. The Gentry IBE scheme was first transformed to the asymmetric setting to give a fair basis of comparison. The exact transformed Gentry IBE scheme is depicted in Appendix A.

Table 4.2 clearly illustrates the price there is to pay for security in the standard model. Therefore, Boneh and Franklin IBE was chosen as the preferred IBE scheme despite the dependency on the random oracle assumption.

IBE requires that OSN profiles can be uniquely identified by a unique public identifier `id`. However, the decision on which string to use as identifier is highly dependent on the underlying OSN and therefore implementation dependent.

	Execution time (ms)			
IBE Scheme	IBE.Setup	IBE.Extract	IBE.Encrypt	IBE.Decrypt
Boneh and Franklin	368.10	13.84	271.90	252.82
Sakai and Kasahara	1257.72	20.49	319.83	259.17
Gentry	24.49	37.46	1136.65	911.32

Table 4.2: Performance comparison of considered IBE schemes in MIRACL

Outsider Recipient Anonymity

The outsider anonymity requirement from Section 4.3.1 is imposed on the recipient set since our solution is developed in the context of OSNs where user interaction plays an important role. Therefore, it is useful that members of the intended recipient set $\mathcal{S}$ know each other. For example, suppose that Alice broadcasts an encrypted message intended to Bob and Dylan using a scheme that fully hides the identity of the recipients. This implies that $\mathtt{id}_{Bob}, \mathtt{id}_{Dylan} \in \mathcal{S}$. As a reaction to Alice's message, Bob wants to write a reply to start a discussion. However, as Bob does not know which other users are allowed to see Alice's message, he can now only encrypt his reply to Alice thereby preventing Dylan from joining the discussion. Nevertheless, this discussion could have been useful to Dylan as well because otherwise Alice would not have included Dylan as a recipient in $\mathcal{S}$ in the first place.

From the outsider-anonymity requirement, it immediately follows that users not necessarily need to be friends to receive each other's messages. In the specific example of Alice, Bob and Dylan, it could be that Bob and Dylan both have Alice as a common friend while no immediate friend connection exists between Bob and Dylan. This should be taken into consideration when determining the identifiers of Bob's and Dylan's profiles, $\mathtt{id}_{Bob}$ and $\mathtt{id}_{Dylan}$ respectively.

As discussed in Section 3.3.2, broadcast encryption schemes can be made more efficient if the recipient set $\mathcal{S}$ is public. So if user interaction is really that important, why not make the intended recipient set public? Consider the example in which Bob's girlfriend celebrates her birthday in a few weeks. When Bob's girlfriend notices that Bob broadcasted an encrypted message to all her friends without including her as a recipient, she will probably know Bob is up to something. This is just one example of possible many that illustrates the negative impact on security, broadcasting of the recipient set $\mathcal{S}$ can have on real life situations. Depending on the context, information can be deduced about the message without decrypting it to plain text.

No redundancy

From the no redundancy requirement it immediately follows that a broadcast encryption scheme should be used, preferably one that hides the anonymity of recipients in the intended recipient set $\mathcal{S}$ to the outside world. However, apart from the outsider-anonymous broadcast encryption scheme from Fazio and Perera [50], no efficient schemes of this kind are described in literature. Since the BE scheme from Fazio

and Perera does not fully benefit from the advantages of IBE, the ANOBE scheme from Libert et al. [80] is preferred for further implementation. Since recipients still have to know who else is included in $\mathcal{S}$, the list of `id`s within $\mathcal{S}$ is concatenated to the plaintext message before encryption.

The scheme from Libert et al. also offers non-repudiation by using signature schemes. Note however, that a trusted authority authorising and publishing the public keys is required for the implementation of signature schemes. Because the general design goals were applicability and user friendliness, no third party PKI can be supported. Therefore, the implemented scheme does not rely on signatures as in the original proposal from Libert et al. [80].

If the security parameter is chosen to be λ, the IBE scheme in Algorithm 5 can only encrypt messages with a maximum length of l bits. This can be seen since in the last step of `IBE.Encrypt` the message m is encrypted by an XOR operation with the result of a hash function $H_3 : \{0,1\}^l \rightarrow \{0,1\}^l$. Because asymmetric IBE schemes can only encrypt these fixed length messages, the scheme from Libert et al. [80] is altered such that the ciphertext in the original proposal contains a with IBE encrypted symmetric session key k that is the same for each user in the recipient set $\mathcal{S}$ on a per message basis. The actual plaintext is then encrypted with a symmetric encryption scheme based on a mode of operation to support longer message lengths.

Authenticity and Integrity

Authenticity and integrity can be achieved at the same time by relying on an authenticated encryption scheme. The integrity of a message is then as strong as the security guarantees of the authenticated encryption scheme.

Note however, that the authentication mechanism still relies on the security guarantees of the OSN. Since no third party PKI mechanism is used, there is no trusted party verifying the identity corresponding to a public key. In OSNs this is not an issue if IBE is used with unique profile identifiers as a public key. Consequently, such an IBE scheme ensures that messages encrypted under a public identifier can only be seen by the owner of the corresponding OSN profile. Verifying whoever owns the OSN profile remains the responsibility of the OSN and the judgement of the OSN profile's connections. However, if the authentication mechanism of the OSN is inadequate, anyone could login to a user's profile to impersonate the actual owner of the profile. Therefore, our proposed solution can not be more secure than the authentication mechanism of the OSN.

In more traditional communication schemes, authenticated encryption uses the symmetric key as agreed during an authenticated key agreement protocol like the Station-to-Station protocol [46]. Authenticity of ciphertexts generated by the authenticated encryption scheme than immediately follows from the usage of the same symmetric session key k as earlier agreed during the protocol. However, since in the proposed solution every OSN user should be able to immediately broadcast confidential messages to other users of the OSN, no key agreement protocols will be used. With the publication of only one broadcast ciphertext, every user in the intended recipient set $\mathcal{S}$ should be immediately able to decrypt it to the original

plaintext message m. Therefore, there is no real authenticity in the value of the tag t generated by the authenticated encryption scheme because anyone with access to the user's profile could have chosen a random symmetric session key k and have used it as an input to the authenticated encryption scheme. Unless, the only one with access to the user's profile is the actual owner of the profile. Therefore, the authenticity guaranteed by the authenticated encryption scheme boils down to the security of the authentication mechanism as powered by the OSN.

No Key Escrow and Key Validation

One of the major drawbacks of IBE schemes is that they inherently imply key escrow (Section 3.2.2). To circumvent the key escrow property of IBE schemes, multiple PKGs can be used implementing a distributed key generation (DKG) mechanism for IBE. Users can then verify their private keys by relying on the basics of commitment schemes (Section **??**).

For the exact details on how a commitment scheme can achieve this verification mechanism, the reader is immediately referred to the exact proposed scheme in Section 4.3.4.

Limited Key Validity

IBE schemes do not allow revocation of public keys (Section 3.2.2). A solution to circumvent this drawback is by concatenating an expiration date to all public identifiers `id`. However, these expiration dates should be publicly available to all OSN users since they are part of the public IBE key. To avoid the management of a third party infrastructure keeping track of expiration dates of all users, a special type of function could be used mapping identifiers `id` to dates. An example of such a function is shown in Algorithm 8.

Algorithm 8 is constructed such that step 1 to 4 only need to be executed once. The sender then stores values d_1, h_1, m_1 locally and only repeats step 5 for each recipient of the message. The exact implementation details could be hidden from the user in software. Different variants of Algorithm 8 could be applied as well. The most important aspect is that everyone in the system uses the same function to map strings to expiration dates.

4.3.3 Updated Model

For the sake of completeness, PKGs are added as an additional entity to our model.

Definition 4.10 (PKG in our security model). A *Public Key Generator* (PKG) is an entity in the security model that never colludes with any other entity in the model since its prime motivation is to improve the current security situation in OSNs.

In the threat model, PKGs are considered to always behave as described in the DKG protocol. Note that this is a simplification of a PKG as it is often encountered in real-world applications. However, considering PKGs as malicious requires far more

Algorithm 8 A function mapping strings to dates

Goal: Avoid a third party infrastructure that keeps track of expiration dates of key pairs in an IBE system

Result: On input of a public identifier `id` the algorithm returns an expiration date in the form `d/M/y h:m`.

1. Choose a hash function $H : \{0,1\}^* \rightarrow \{0,1\}^l$ mapping binary strings of arbitrary length to binary strings of a fixed length l.
2. Calculate $r = H(\texttt{id})$ and interpret the result r as an integer.
3. Calculate $tot_m = r \bmod 40320 = r \bmod (60 \cdot 24 \cdot 28)$, where tot_m denotes the expiration time in minutes within a certain month.
4. Calculate three integers d_1, h_1, m_1 denoting an expiration day, hour and minute respectively with $1 \leq d_1 \leq 28$, $0 \leq h_1 \leq 23, 0 \leq m_1 \leq 59$ as follows
   a) The expiration minute is calculated as $m_1 = tot_m \bmod 60$
   b) The expiration hour is calculated as $h_1 = \frac{tot_m}{60} \bmod 24$
   c) The expiration day is calculated as $d_1 = \frac{tot_m}{60 \cdot 24}$

   It can be shown that d_1, h_1, m_1 are chosen uniformly random within their boundaries if the random oracle assumption holds for the hash function $H(\cdot)$.
5. Let `nowIsEarlierThan(`d_1`,`h_1`,`m_1`)` be a function that returns `true` if the current time `d/M/y h:m` is before d_1`/M/y` h_1`:`m_1 and `false` otherwise. Output the expiration date as
   a) If `nowIsEarlierThan(`d_1`,`h_1`,`m_1`)` = `true`, return d_1`/M/y` h_1`:`m_1.
   b) If `nowIsEarlierThan(`d_1`,`h_1`,`m_1`)` = `false` and `M`+1 $\leq$ 12, return d_1`/M/y` h_1`:`m_1.
   c) Else return d_1`/1/(y+1)` h_1`:`m_1

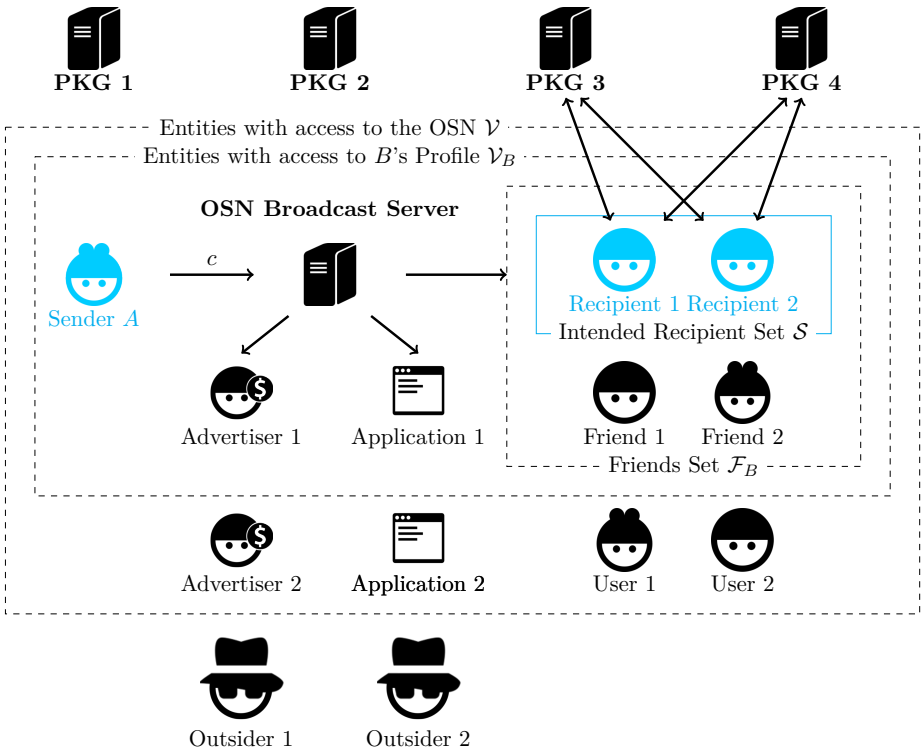

Figure 4.2: Model of the desired OSN situation. Entities with the ability to decrypt the ciphertext are coloured blue.

complex distributed key generation algorithms which are out of the scope of this thesis. For DKG protocols that can be used in more hostile PKG environments as encountered in practice, the reader is referred to Kate et al. [75].

Figure 4.2 illustrates the changes on the original model of the OSN situation in Figure 4.1. At the top of Figure 4.2 four PKGs are introduced implementing a (t, n) DKG protocol with $t = 2$ and $n = 4$ (Section 3.5). Double arrows represent the secure authentication process in which the recipients communicate with the PKG to receive a share of their secret key. In Figure 3.1 this communication was illustrated by two separate single arrows between the PKG and Bob. However, abstraction is made of the exact communication protocol between the recipients and the PKG.

Apart from the newly introduced PKGs, Figure 4.2 differs from Figure 4.1 in several ways. Sender A no longer specifies the set of intended recipients $\mathcal{S}$ to the OSN broadcast server. Therefore, the OSN broadcast server delivers the message to all entities with access to A's profile $\mathcal{V}_A$. Note that even Friend 1 and 2 are able to see the broadcasted message which was not the case in Figure 4.1. However, since the broadcasted message is actually a ciphertext c of the original message m, only the entities in blue will be able to read the confidential content of the original plaintext message m.

4.3.4 Scheme

Taking all aforementioned cryptographic design decisions into account results in the scheme presented in Algorithm 9.

4.3.5 Evaluation

Algorithm 9 achieves the earlier stated cryptographic design goals from Section 4.3.1. However, it is too early to conclude on the more general design goals from the introduction (Section 1.3) since the achievement of this goals is implementation dependent. The cryptographic goals are realised as follows:

Confidentiality The proposed scheme achieves confidentiality as in [27, 80], because a session key k can only be obtained if the recipient holds the corresponding secret key $s_{\mathtt{id}_i}$ to an identifier $\mathtt{id}_i$ that is included in $\mathcal{S}$. Confidentiality of k is thus guaranteed by ANO-IND-CCA secure Boneh and Franklin IBE [27]. Confidentiality of the plaintext message m than immediately follows from the confidentiality of the authenticated encryption in step 5 of `Publish`.

Outsider recipient anonymity Our solution is made anonymous by relying on the ANO-IBE scheme from Boneh and Franklin [24]. Furthermore, the broadcasting mechanism is inspired by the BE scheme from Libert et al. [80] which is recipient anonymous as well. The BE scheme is applied to broadcast k. In terms of efficiency, users are required to decrypt w_i on average $O(\eta/2)$ before obtaining k due to the anonymity of the BE scheme. Both Barth et al. [14] and Libert et al. [80] propose using a tag based system to hint users where they can find their symmetric key. However, it was deliberately decided to not implement such property in the scheme as it introduces a dependency of the public parameters linear in the total number of users in the system.

Algorithm 9 is outsider recipient anonymous due to the concatenation of the recipient set to the plaintext message m in step 4 of `Publish`.

No redundancy The broadcast message $\mathcal{B}$ should only be published once since $\mathcal{B}$ contains a concatenation w of the with IBE encrypted session key k for every recipient in $\mathcal{S}$.

Authenticity and integrity Authenticity and integrity are guaranteed by the authenticated symmetric encryption scheme and the authentication mechanism of the OSN. If the assumptions on the OSN's authentication mechanism hold, only the owner of an OSN profile should be able to actively broadcast messages in name of the corresponding profile identifier $\mathtt{id}_i$.

No key escrow and key validation Key escrow is avoided by the DKG protocol included in both the `Setup` and `KeyGen` stages of the algorithm. The last check of the `KeyGen` step of Algorithm 9 validates the correctness of a users' shares. However, this check does not ensure security against malicious PKGs since the Pedersen protocol [93] is insecure in the presence of malicious PKGs. That is, malicious PKGs can still affect the outcome of certain bits of the shared master secret key msk with

non-negligible advantage. To circumvent these issues, the DKG scheme from Gennaro et al. [59] should be implemented. However, since the scheme is developed in a threat model where PKGs are assumed trustworthy, this concern falls out of the scope of this thesis. Implementation of the DKG protocol from Gennaro et al. [59] occurs similar to the scheme from Pedersen [93] since it relies on the same mathematical concepts. Therefore, adapting Algorithm 9 to a more hostile DKG environment should be straightforward.

Limited key validity For the sake of clarity, concatenation of expiration dates with public keys is not explicitly included in Algorithm 9. However, with the help of Algorithm 8 this should be trivial since only the interpretation of the identifier symbol $\texttt{id}_i$ changes from a permanent identifier of a user to only a temporary identifier when concatenated with an expiration date.

The proposed solution can be used in any OSN that assigns unique public identifiers, such as usernames. Since the public keys are represented as strings, users are not required to upload keys to an additional third party server. Distributed key generation solves the key escrow issues that come with IBE solutions.

4.4 Summary

Algorithm 9 An outsider recipient anonymous identity-based broadcast encryption scheme

`Setup`(λ, t, n): Outputs the public *params* of the system with respect to the security parameter λ, the number of PKGs n and the threshold t.

1. On input of security parameter λ generate a prime q, two groups G_1, G_2 of order q, and an admissible bilinear map $e : G_1 \times G_2 \to G_T$. Choose random generators $P \in G_1$ and $Q \in G_2$.
2. Choose cryptographic hash functions $H_1 : \{0,1\}^* \to G_1$, $H_2 : G_T \to \{0,1\}^l$ and $H_3 : \{0,1\}^l \to \{0,1\}^l$, such that H_1, H_2 can be modelled as random oracles.
3. Each PKG j generates $n-1$ shares σ_{jv} of a Pedersen VSS scheme by executing `DKG.Setup`, and redistributing the $n-1$ shares σ_{jv} with the other v PKGs.
4. Each PKG j publishes $P_{pub}^{(j)} = sk_j P$, s.t., $sk_j = \sum_{v=1}^{n} \sigma_{jv}$.

The master secret key $msk = \sum_{j \in \Lambda} b_j sk_j$ for $b_j = \prod_{z \in \Lambda} \frac{z}{z-j}$ cannot be retrieved unless Λ is a subset of size t different PKG servers. The following parameters are published publicly:

$$params = \{q, G_1, G_2, e, P, Q, H_1, H_2, H_3, t, n, P_{pub}^{(0)}, \ldots, P_{pub}^{(n)}\}$$

`KeyGen`$(\{\texttt{PKG}_0, \ldots, \texttt{PKG}_t\}, \texttt{id}_i)$: On input of a user $\texttt{id}_i$ the subset Λ of size t of PKG servers, generates a valid private key for $\texttt{id}_i$.

1. User with identifier $\texttt{id}_i$, authenticates to Λ or all PKGs and sends $\texttt{id}_i$.
2. Each PKG computes $Q_{\texttt{id}_i} = H_1(\texttt{id}_i)$, and $Q_{priv,\texttt{id}_i}^{(j)} = sk_j Q_{\texttt{id}_i}$, where sk_j is the secret key from PKG j.
3. The user $\texttt{id}_i$ computes the shared public parameter P using the Lagrange coefficients b_j as follows:

$$P = \sum_{j \in \Lambda} b_j P_{pub}^{(j)} \quad \text{for} \quad b_j = \prod_{z \in \Lambda} \frac{z}{z-j}$$

4. All PKGs in Λ return $Q_{priv,\texttt{id}_i}^{(j)}$ to the corresponding user $\texttt{id}_i$ over a secure channel.
5. Each user verifies for each $Q_{priv,\texttt{id}_i}^{(j)}$ value whether,

$$e\left(Q_{priv,\texttt{id}_i}^{(j)}, P\right) \stackrel{?}{=} e\left(Q_{\texttt{id}_i}, P_{pub}^{(j)}\right)$$

Next, $\texttt{id}_i$ calculates the private key $sk_{\texttt{id}_i}$ using the Lagrange coefficients b_j as follows:

$$sk_{\texttt{id}_i} = \sum_{j \in \Lambda} b_j Q_{priv,\texttt{id}_i}^{(j)} \quad \text{for} \quad b_j = \prod_{z \in \Lambda} \frac{z}{z-j}$$

In this way, no user or PKG learns the master key msk of the system. This algorithm combines `DKG.Reconstruct`, `IBE.Extract` and `BE.KeyGen` algorithms.

`Publish`$(params, \mathcal{S}, m)$**:** Takes the message m, the subset $\mathcal{S}$ of size η and the public parameters $params$, output a broadcast message $\mathcal{B}$.

1. Generate a random symmetric session key $k \leftarrow \{0,1\}^l$.
2. Choose a random value $\rho \in \{0,1\}^l$ and compute r as a hash of concatenated values $r = H_3(\{\rho \parallel k\})$
3. For each recipient $\mathtt{id}_i \in \mathcal{S}$, compute the ciphertext, running the `IBE.Encrypt` algorithm, as follows.

$$w_i = \rho \oplus H_2\left(g^r_{\mathtt{id}_i}\right) \quad \text{where} \quad g_{\mathtt{id}_i} = e\left(Q_{\mathtt{id}_i}, P_{pub}\right) \in G_T$$

4. Let w be a randomised concatenation, then the authenticated data $\mathcal{A}$ is computed as

$$\begin{aligned} \mathcal{A} &= \{\eta \parallel rP \parallel k \oplus H_3(\rho) \parallel w_1 \parallel w_2 \parallel \ldots \parallel w_\eta\} \\ &= \{\eta \parallel U \parallel v \parallel w\} \quad \text{for} \quad w = \{w_1 \parallel w_2 \parallel \ldots \parallel w_\eta\} \end{aligned}$$

   And $\mathcal{M}$ a concatenation of the intended recipient set $\mathcal{S}$ and the plaintext message m, such that $\mathcal{M} = \{m \parallel \mathcal{S}\}$. (`BE.Encrypt`)

5. Apply authenticated symmetric encryption

$$\langle c, t \rangle \leftarrow \mathtt{E}_k(\mathcal{M}, \mathcal{A})$$

6. The following message is then published in the OSN

$$\mathcal{B} = \{\mathcal{A} \parallel t \parallel c\}$$

`Retrieve`$(params, sk_{\mathtt{id}_i}, \mathcal{B})$**:** on input of the broadcast message $\mathcal{B}$ and the private key $sk_{\mathtt{id}_i}$ of user $\mathtt{id}_i$, reconstruct the plaintext message m. This algorithm comprises the `{IBE,BE}.Decrypt` algorithms. For each $i \in \{\}$

1. Compute $w_i \oplus H_2(e(sk_{\mathtt{id}_i}, U)) = \rho$ for $sk_{\mathtt{id}_i}$, and $v \oplus H_3\{\rho\} = k$
2. Set $r = H_3(\rho, k)$. Verify $U \stackrel{?}{=} rP$. If the check fails, try next W_i and return to 1.
3. Retrieve $\langle \mathcal{M}, t' \rangle \leftarrow \mathtt{D}_k(c, \mathcal{A})$
4. Verify whether $t' \stackrel{?}{=} t \in \mathcal{B}$, and return m. Otherwise return $\perp$.

Chapter

5 Implementation

5.1 Software Architecture

There is still a long way to go from Algorithm 9 towards a practical implementation for OSNs. As a proof of concept, Algorithm 9 is applied to Facebook, the largest online social network at the time of writing. Recall that one of the more general goals of this thesis is to develop a solution that is applicable, i.e. a solution that does not require the OSN environment to be altered. Until so far this goal is successfully met since Algorithm 9 could be designed completely based on a more general model for OSNs. The resulting algorithm is a solution that can be applied to virtually any OSN that allows to uniquely distinguish users based on unique public identifiers, a requirement that discriminates almost no existing OSNs.

5.1.1 Software Environment

Despite the avoidance of complex third party infrastructures, some software is needed that effectively implements Algorithm 9 in a user-friendly way. Ideally, this is an easy-to-install piece of software that runs as an additional layer on top of the current infrastructure of the OSN user. Therefore, it was chosen to implement Algorithm 9 in the form of a browser extension.

Since Scramble [16] already has a user friendly interface that supports all required use cases to implement encryption on OSNs, it is natural to integrate our IBE scheme into Scramble. Besides from Scramble being open source and lightweight, the most important trigger to modify the existing Scramble code is that it is developed at KU Leuven.

5.1.2 Existing Environment

Recall from Section 1.2 that Scramble [16] is a Firefox extension that currently relies on OpenPGP [33] for key management. Due to the dependency on OpenPGP, Scramble is independent of any OSN. In fact, Scramble only functions as an encryption and decryption tool that can be used on any website offering users to submit content. However, users who want to be part of the recipient set of the uploaded messages

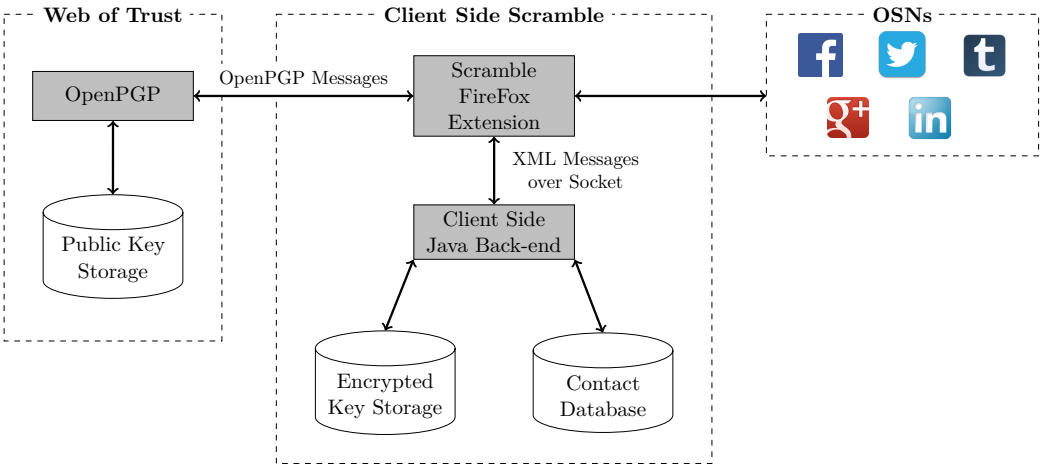

Figure 5.1: Original Scramble Architecture

need to have uploaded a public key to the OpenPGP network beforehand. The existing Scramble environment is shown in Figure 5.1.

Since Scramble is a FireFox extension, the user interface (UI) is implemented in Javascript. Although Javascript is ideal for synchronous UIs, it is not the desired programming language for computational demanding tasks such as encryption and decryption. Therefore, Scramble communicates with a back-end in Java that implements all cryptographic operations. Every time a user selects a computation intensive task in the Javascript UI, Javascript sends an XML message requesting the result from the client side Java back-end. The Java back-end processes the request and immediately sends the result in another XML message back to the UI. Sending and receiving of XML messages between FireFox extension and Java back-end, takes place synchronously over a socket listening on an internal port.

Scramble relies on OpenPGP for key management. Therefore, the FireFox extension communicates with a web of trust (Section 3.1.2) storing all public keys of users who subscribed to the OpenPGP network. Because the OpenPGP network stores more keys than a Scramble user needs, Scramble offers the functionality to store public keys from the OpenPGP network locally in a contact database via the client side Java back-end. Furthermore, the client-side Java back-end has access to an encrypted list of the user's secret keys corresponding to public keys that are already in the OpenPGP network. With the help of a passphrase the Java back-end has access to these private keys to allow encryption of received messages.

5.1.3 Changes to the Existing Environment

The altered Scramble architecture is schematically illustrated in Figure 5.2. Scramble still offers the original functionality as an alternative to our IBE implementation. However, for reasons of conciseness Figure 5.2 omits the original OpenPGP implementation although it is not removed.

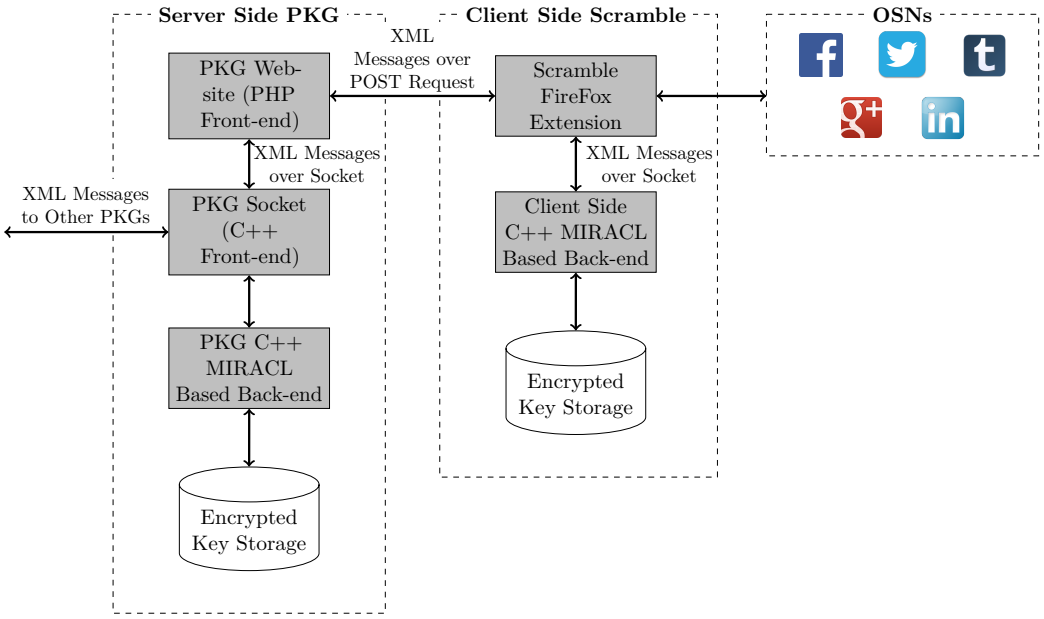

Figure 5.2: New Scramble Architecture

The new client side Scramble architecture implements a C++ based back-end instead of the earlier Java back-end because the most efficient pairing-based multi precision libraries are written in C. In fact only two pairing-based libraries are widely accepted in practical implementations: MIRACL [101] and PBC [3]. MIRACL was preferred over PBC since it is generally faster in its pairing computations. All core algorithms of MIRACL are implemented in C while a C++ wrapper allows object-oriented programming.

The contact database is removed from the original Scramble implementation as illustrated in Figure 5.1 since public keys no longer have to be explicitly stored in the architecture. More specifically, since Scramble can rely on IBE, the public keys are inherently part of the supported OSN. Therefore, the FireFox extension falls back on a number of calls to supported OSN APIs in order to get all public keys of one's connections.

Figure 5.2 exchanges the web of trust from Figure 5.1 for a DKG infrastructure in order to support IBE without key escrow. For clarity, only one PKG is shown since all PKGs will have the same structure. The PKG supports two front-ends: a C++ based front-end and a PHP based front-end.

The C++ based front-end of the PKG only serves as a front-end during execution of the DKG protocol. Negotiation of the shares is implemented over synchronous sockets. At the startup of the PKG socket, the administrator is asked for a secret passphrase. Then, the sockets start listening on a predetermined port until all shares are correctly negotiated. Once all PKGs have exchanged their shares, the PKG calculates its public parameters and finishes the `Setup` step from Algorithm 9. The coefficients of the secret polynomial are encrypted with the earlier specified

passphrase along with the negotiated secret share. After storage of these secret parameters, the C++ socket starts listening on another port to handle requests from the PHP front-end. Handling of private key extraction requests is multithreaded to handle requests of multiple clients at the same time.

The PHP based front-end receives private key extraction queries from the Scramble FireFox extension in the form of POST requests. The PKG communicates the requested `id` to the listening C++ socket in the form of an XML message. After the required MIRACL based pairing computations, the PKG socket sends the response back to the PHP webpage such that it is published in the form of an XML message over HTTPS [96].

Note that the new architecture from Figure 5.2 can be applied to virtually any OSN that identifies its users with publicly available identifiers. However, Scramble is made more user friendly by implementing OSN specific API calls. Currently, an implementation for Facebook serves as a proof-of-concept. Further extension to support other OSNs could be subject of future work.

5.2 Implementation Details

The implementation of Algorithm 9 still requires some practical design decisions that are further discussed in this section.

5.2.1 Type of Elliptic Curve

The underlying elliptic curve determines the groups of the bilinear pairing $e : G_1 \times G_2 \rightarrow G_T$ and thus the security level of the application. The MIRACL library supports 5 different curves and 6 different security levels. However, the Barreto-Lynn-Scott (BLS) curve [13] is preferred since it provides the highest level of security in MIRACL. BLS curves rely on Ate pairings [71] with an embedding degree of 24. Consequently, BLS curves are considered suitable for a security level of 256 bits.

5.2.2 Authenticated Encryption Scheme

An AES-GCM [89] implementation is used for the authenticated encryption scheme in Algorithm 9 since it is one of the more efficient authenticated encryption schemes unencumbered by patents. The AES-GCM implementation as provided by MIRACL is used with authentication tags of 128 bits. Apart from a symmetric key of 128, 192 or 256 bits, AES-GCM also requires an Initialisation Vector (IV). The recommended length of the IV is 96 bits because it can be handled more efficiently [72].

5.2.3 Key Lengths

Since the maximum level of security in MIRACL is determined by the BLS curve, all implemented key lengths in Algorithm 9 follow from the 256 bits security level ($l = 256$). Note that in Step 1 of `Retrieve` the decryption of a recipient's session key is calculated as:

$$w_i \oplus H_2\left(e\left(sk_{\mathtt{id}_i}, U\right)\right) = \rho$$

followed by

$$v \oplus H_3\{\rho\} = k$$

with

$$H_2 : G_T \rightarrow \{0,1\}^l$$

Since $l = 256$, w_i and ρ are binary sequences of 256 bits. Hence, ρ only contains sufficient randomness to securely encrypt a session key k of the same length. The full 256 bit k is not completely used for symmetric encryption as the AES-GCM scheme still requires an IV as well. Since the randomness and key freshness of the IV is at least as important as the secrecy of k [1], it was deliberately chosen not to apply key derivation functions to derive a separate symmetric encryption key and IV from the same value k. Therefore, the first 128 bits of k are used as the symmetric key of AES-GCM, while the second 96 bits function as an IV. According to NIST, this should ensure confidentiality at least until 2030. However, if a higher level of security should be required, key derivation on k could be considered.

5.2.4 Hash Functions

The hash function $H_1 : \{0,1\}^* \rightarrow G_1$ is implemented using the MIRACL function call `hash_to_group`. Hash function $H_2 : G_T \rightarrow \{0,1\}^{256}$ relies on the MIRACL`hash_to_aes_key` function. Both H_1 and H_2 internally fall back on SHA-256 [92]. The MIRACL provided SHA-256 algorithm is also used for implementation of the hash $H_3 : \{0,1\}^{256} \rightarrow \{0,1\}^{256}$.

5.2.5 Generating Random Numbers

Random numbers are generated using the MIRACL build-in strong random number generator. The strong random generator is initialised with a time-of-day value and a binary array of 1024 bits read from `/dev/urandom`. These values are then used as a seed for the generation of random numbers. The practical implementation of the MIRACL strong random generator is based on an advice published by RSA Laboratories in [85].

5.2.6 Public Key

The decision on which string to use as a public IBE key `id` is dependent on the underlying OSN. The desired properties for `id` are the following:

1. The public key should uniquely identify the user
2. The public key should be mandatory for every user of the social network
3. The public key should not change frequently over time

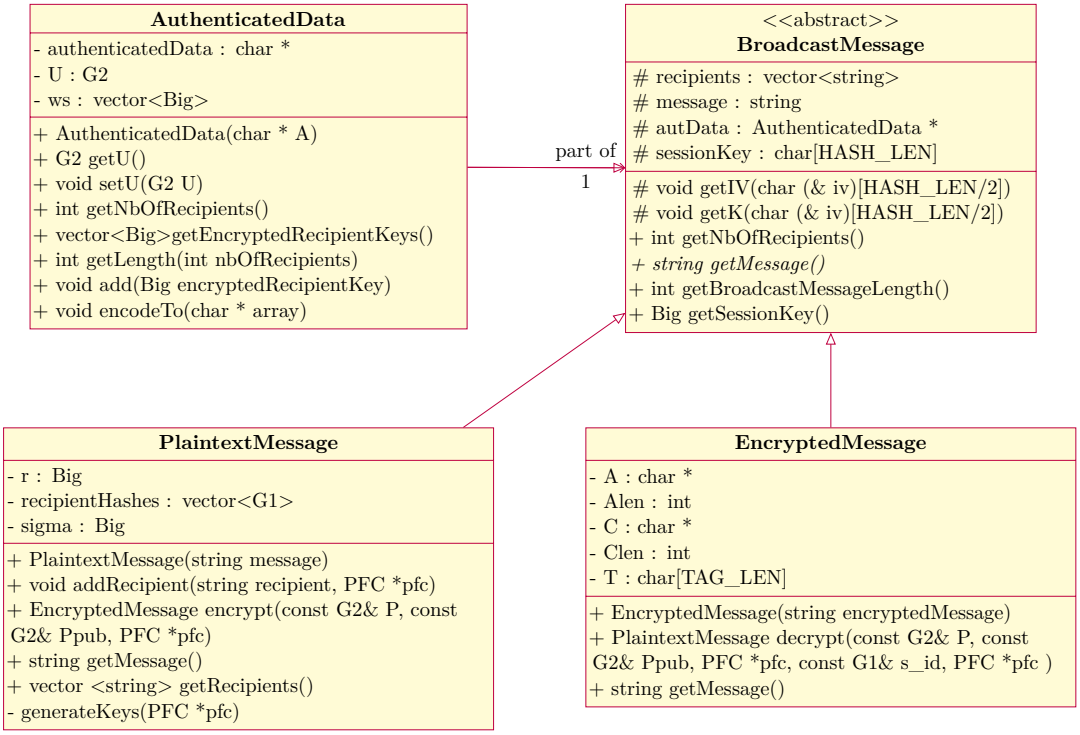

Figure 5.3: Class Diagram of Client Side C++ MIRACL Based Back-end

4. The public key should be an inherent part of the infrastructure of the social network thereby meaning that the previous three properties are already ensured by the provider of the social network.

5. The public key should be publicly available, even to users that are not part of the set of entities with access to the user's profile $\mathcal{V}_{\texttt{id}}$

In Facebook the best key satisfying these properties is a Facebook username. A Facebook username is ensured to be unique since it is part of a profile's URL, e.g. `http://www.facebook.com/profile.name` where `profile.name` functions as `id`. Moreover, the Facebook policy ensures that this username is mandatory for every user and can only be changed once. Lastly, the Facebook username is also public to outsiders using the Facebook API thereby fulfilling all the required conditions for an IBE key in our architecture.

5.3 Implemented Scheme

5.3.1 Client-Side Implementation

Code Structure

TODO: give a short description on what each function implements

	Execution Times (ms)	
Number of Recipients	Encryption	Decryption
1	284.5	275.4
10	2564.5	460.9
15	3799.6	560.6
50	12300.5	1237.8
100	25867.7	2260.2

Table 5.1: Performance of the proposed scheme in function of the number of intended recipients.

Encountered Issues

TODO: Say something about pre-computation in MIRACL and why the const G2& is necessary.

5.3.2 Server-Side Implementation

Code Structure

TODO: Add a class diagram for the server-side implementation
TODO: give a short description on what each function implements

Encountered Issues

TODO: Say something about the difficulties of multithreading MIRACL

5.4 Performance Analysis

Performance experiments were conducted on an Intel Core 2.4 GHz i5 processor with 8 Gb of 1600 MHz DDR3L onboard memory. Table 5.1 illustrates the execution times for Algorithm 9 with the implementation details from Section 5.2.

Encryption times are linearly dependent on the number of intended recipients since more intensive pairing computations are required for each additional user within the recipient set.

Each recipient has to decrypt w_i an average of $\eta/2$ times to retrieve the secret session key k. However, the experiments measured the worst-case execution time for decryption since the recipient had to decrypt all w_i values before retrieving k in the last attempt. After recovering k, the recipient executes AES-GCM for the decryption of the ciphertext c to the plaintext message m. Note that recipient anonymity of $\mathcal{S}$ comes at the cost of reduced efficiency, as to hide $\mathcal{S}$ it is required to produce more `IBE.Encrypt` calls.

TODO: Add another table listing how much time each step of Algorithm 9 requires.

5.5 Limitations of Current Implementation

The algorithm as it is currently implemented only serves as a proof-of-concept since it lacks a number of requirements needed for secure use in practice. These aspects were not implemented due to the limited time available. Although all core requirements for the protocol in Algorithm 9 are present, only the aspects for practical usage in more hostile environments are missing. However, including these aspects should be straightforward and is only a matter of implementation instead of deliberate design decisions.

5.5.1 Client Side

Since the IBE scheme is integrated in the existing Scramble UI, not many adaptions should be made to the client side implementation. The major drawback of our IBE proposal is that it makes the Scramble plugin more dependent on the underlying OSN. Consequently, every OSN should be separately integrated into the Scramble plugin since each OSN offers its own API calls for reading out friend connections. More OSNs than Facebook should be actively supported by the IBE Scramble tool to motivate the use of the plugin.

5.5.2 Server Side

In its present form, the PKG is only simulated in a local environment. Therefore, it suffices to rely on synchronous communication over C++ sockets each listening on a different port. However, before adaption to a world-wide DKG network is possible, the protocol should take more asynchronous aspects into account such as connection-loss, DOS-attacks or undelivered packets. Kate et al. [75] propose a more advanced DKG protocol in the asynchronous setting that could be adapted for our IBE setting.

Since the DKG protocol is currently always simulated in a local environment, all PKG sockets communicate their XML messages in plain text. More secure socket protocols should be considered such as SSL [56] for adoption to a more hostile distributed setting.

Furthermore, the current assumptions on the PKGs (Section **??**) are too severe for practical environments. In practice the current Pedersen DKG scheme [93] is insecure and is better replaced by the one from Gennaro et al. [59]. If all mentioned updates on the current server side scheme are effectively achieved, more relaxed assumptions on the PKG are in place. With less stringent PKG assumptions, the DKG network could even be partially supported by the OSN providers to show their good intensions of making their networks more private.

5.5.3 Limitations Effecting Both Client and Server

Currently, private key extraction takes the form of a POST request over HTTPS [96]. Although the PKGs publish the response in plain text to their PHP website, the communication is encrypted by a self-signed SSL certificate. Ideally, this certificate

is signed by a world-wide trusted CA such that the client-side implementation can effectively verify whether it is communicating with a valid PKG.

Although web communication between the client side Scramble tool and the server side PKG is already encrypted, a client can request the private key parameters for every `id` since there is no authentication mechanism checking the user's identity. However, Facebook provides third party authentication in its API. If a Facebook login dialogue is integrated in the Scramble UI, the returned Facebook authentication token serves as a proof to the PKG that the requester of the private key resembles the owner of the corresponding Facebook profile.

5.6 Summary

Chapter

6 Conclusion

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

Appendices

Appendix

A Gentry's IBE Scheme

TODO for this Appendix: change the notation such that it corresponds to all other pairing based computations in this thesis Gentry [60] proposed the first IND-ANO-CCA secure scheme in the random oracle model. The original proposed scheme from Gentry relies on symmetric pairings. A transformed version of the algorithm to the asymmetric setting can be found in Algorithm 10.

The correctness of the transformed scheme in Algorithm 10 can be proven as follows.

$$
\begin{aligned}
& e\left(u, h_{\mathrm{ID},2} h_{\mathrm{ID},3}^{\beta}\right) v^{r_{\mathrm{ID},2}+r_{\mathrm{ID},3}\beta} \\
& \quad = e\left(p_1^{s(\alpha-\mathrm{ID})}, \left(h_2 h_3^{\beta}\right)^{\frac{1}{\alpha-\mathrm{ID}}} q_2^{\frac{-\left(r_{\mathrm{ID},2}+r_{\mathrm{ID},3}\beta\right)}{\alpha-\mathrm{ID}}}\right) e\left(p_1, q_2\right)^{s\left(r_{\mathrm{ID},2}+r_{\mathrm{ID},3}\beta\right)} \\
& \quad = e\left(p_1^{s(\alpha-\mathrm{ID})}, \left(h_2 h_3^{\beta}\right)^{\frac{1}{\alpha-\mathrm{ID}}}\right) = e\left(p_1, h_2\right)^{s} e\left(p_1, h_3\right)^{s\beta}
\end{aligned}
$$

Thus, the check passes. Moreover, as in the ANO-IND-CPA scheme,

$$
e\left(u, h_{\mathrm{ID}}\right) v^{r_{\mathrm{ID},1}} = e\left(p_1^{s(\alpha-\mathrm{ID})}, h^{\frac{1}{\alpha-\mathrm{ID}}} q_2^{\frac{-r_{\mathrm{ID},1}}{\alpha-\mathrm{ID}}}\right) e\left(p_1, q_2\right)^{s r_{\mathrm{ID},1}} = e\left(p_1, h\right)^{s},
$$

as required.

Algorithm 10 Gentry's asymmetric IBE Scheme [60]

Let G_1, G_2 and G_T be groups of order p and let $e : G_1 \times G_2 \to G_T$ be the bilinear map. The IBE system works as follows.

Setup: The PKG picks random generators $p_1, g_1 \in G_1$, generators $q_2, h_1, h_2, h_3 \in G_2$ and a random $\alpha \in \mathbb{Z}_p$. It sets $g_1 = p_1^{\alpha} \in G_1$. It chooses a hash function H_1 and $H_2 : \{0,1\}^* \to \{0,1\}^n$ from a family of universal one-way hash functions. The public *params* and private *masterkey* are given by

$$params = (p_1, q_2, h_1, h_2, h_3, H_1, H_2) \qquad masterkey = \alpha$$

KeyGen: To generate a private key for identity $\mathrm{ID} \in \mathbb{Z}_p$, the PKG generates random $r_{\mathrm{ID},i} \in \mathbb{Z}_p$ for $i \in \{1,2,3\}$, and outputs the private key

$$d_{\mathrm{ID}} = \{(r_{\mathrm{ID},i}, h_{\mathrm{ID},i}) : i \in \{1,2,3\}\}, \text{ where } h_{\mathrm{ID},i} = \left(h_i q_2^{-r_{\mathrm{ID},i}}\right)^{\frac{1}{\alpha - \mathrm{ID}}} \in G_2$$

If $\mathrm{ID} = \alpha$, the PKG aborts. As before, we require that the PKG always use the same random values $\{r_{\mathrm{ID},i}\}$ for ID.

Encrypt: To encrypt $m \in \{1,0\}^n$ using identity $\mathrm{ID} \in \mathbb{Z}_p$, the sender generates random $s \in \mathbb{Z}_p$, and sends the ciphertext

$$\begin{aligned} C &= \left(g_1^s p_1^{-s \cdot \mathrm{ID}},\ e(p_1, q_2)^s,\ m \oplus H_2\{e(p_1, h_1)^s\},\ e(p_1, h_2)^s\, e(p_1, h_3)^{s\beta}\right) \\ &= (u, v, w, y) \end{aligned}$$

Note that $u \in G_1, v \in G_T, w \in \{1,0\}^n$ and $y \in G_T$. We set $\beta = H_1\{u, v, w\}$. Encryption does not require any pairing computations once $e(p_1, q_2)$, and $\{e(p_1, h_i)\}$ have been pre-computed or alternatively included in *params*.

Decrypt: To decrypt ciphertext $C = (u, v, w, y)$ with ID, the recipient sets $\beta = H_1\{u, v, w\}$ and tests whether

$$y = e\left(u, h_{\mathrm{ID},2} h_{\mathrm{ID},3}^{\beta}\right) v^{r_{\mathrm{ID},2} + r_{\mathrm{ID},3}\beta}$$

If the check fails, the recipient outputs $\perp$. Otherwise, it outputs

$$m = w \oplus H_2\{e(u, h_{\mathrm{ID},1})\, v^{r_{\mathrm{ID},1}}\}$$

Appendix

B 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[**?**].

B.1 Setting up the DKG

B.2 Setting up Scramble

Appendix

C Dutch Summary

As this thesis is written in English a Dutch summary ranging from 1 to 5 pages is included here.

Appendix

D English Paper

The English PETS paper is included here and should have the IEEE layout as proposed at `http://www.ieee.org/publications_standards/publications/authors/authors_journals.html`

Bibliography

[1]

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

[3] On the implementation of pairing-based cryptography. Master's thesis, Stanford, 2007.

[4] Public-key encryption with oblivious keyword search. priced oblivious transfer. Master's thesis, KU Leuven, 2008.

[5] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. In Shoup [104], pages 205–222.

[6] H. Abelson, R. Anderson, S. M. Bellovin, J. Benalob, M. Blaze, W. Diffie, J. Gilmore, P. G. Neumann, R. L. Rivest, J. I. Schiller, and B. Schneier. The risks of key recovery, key escrow, and trusted third-party encryption. *World Wide Web J.*, 2(3):241–257, June 1997.

[7] 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 [36], pages 20–44.

[8] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin. Persona: an online social network with user-defined privacy. In P. Rodriguez, E. W. Biersack, K. Papagiannaki, and L. Rizzo, editors, *SIGCOMM*, pages 135–146. ACM, 2009.

[9] 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.

[10] E. Balsa, L. Brandimarte, A. Acquisti, C. Diaz, and S. F. G´urses. Spiny CACTOS: OSN users attitudes and perceptions towards cryptographic access control tools. In *Workshop on Usable Security*, Lecture Notes in Computer Science, page 10, San Diego,CA,USA, 2014. Springer-Verlag.

[11] M. Barbosa and P. Farshim. Efficient identity-based key encapsulation to multiple parties. In N. P. Smart, editor, *IMA Int. Conf.*, volume 3796 of *Lecture Notes in Computer Science*, pages 428–441. Springer, 2005.

[12] 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.

[13] P. S. L. M. Barreto, B. Lynn, and M. Scott. Constructing elliptic curves with prescribed embedding degrees. In S. Cimato, C. Galdi, and G. Persiano, editors, *SCN*, volume 2576 of *Lecture Notes in Computer Science*, pages 257–267. Springer, 2002.

[14] A. Barth, D. Boneh, and B. Waters. Privacy in encrypted content distribution using private broadcast encryption. In G. D. Crescenzo and A. D. Rubin, editors, *Financial Cryptography*, volume 4107 of *Lecture Notes in Computer Science*, pages 52–64. Springer, 2006.

[15] 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.

[16] F. Beato, M. Kohlweiss, and K. Wouters. Scramble! your social network data. In S. Fischer-Hübner and N. Hopper, editors, *PETS*, volume 6794 of *Lecture Notes in Computer Science*, pages 211–225. Springer, 2011.

[17] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In C. Boyd, editor, *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer, 2001.

[18] 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.

[19] 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.

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

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

[22] A. Boldyreva, V. Goyal, and V. Kumar. Identity-based encryption with efficient revocation. *IACR Cryptology ePrint Archive*, 2012:52, 2012.

[23] 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.

[24] D. Boneh and X. Boyen. Efficient selective-id secure identity-based encryption without random oracles. In C. Cachin and J. Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 2004.

[25] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In Cramer [42], pages 440–456.

[26] D. Boneh, X. Ding, G. Tsudik, and C.-M. Wong. A method for fast revocation of public key certificates and security capabilities. In D. S. Wallach, editor, *USENIX Security Symposium*. USENIX, 2001.

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

[28] J. Bonneau and S. Preibusch. The privacy jungle: On the market for data protection in social networks. *Economics of Information Security and Privacy*, pages 121–167, 2010.

[29] D. M. Boyd and N. B. Ellison. Social network sites: Definition, history, and scholarship. *J. Computer-Mediated Communication*, 13(1):210–230, 2007.

[30] X. Boyen. A tapestry of identity-based encryption: practical frameworks compared. *IJACT*, 1(1):3–21, 2008.

[31] X. Boyen and B. Waters. Anonymous hierarchical identity-based encryption (without random oracles). In C. Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 290–307. Springer, 2006.

[32] G. Brassard, D. Chaum, and C. Crépeau. Minimum disclosure proofs of knowledge. *J. Comput. Syst. Sci.*, 37(2):156–189, 1988.

[33] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer. OpenPGP Message Format. RFC 4880 (Proposed Standard), Nov. 2007. Updated by RFC 5581.

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

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

[36] 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.

[37] A. D. Caro, V. Iovino, and G. Persiano. Fully secure anonymous hibe and secret-key anonymous ibe with short ciphertexts. *IACR Cryptology ePrint Archive*, 2010:197, 2010.

[38] L. Chen and Z. Cheng. Security proof of sakai-kasahara's identity-based encryption scheme. *IACR Cryptology ePrint Archive*, 2005:226, 2005.

[39] B. Chor, A. Fiat, M. Naor, and B. Pinkas. Tracing traitors. *IEEE Transactions on Information Theory*, 46(3):893–910, 2000.

[40] 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.

[41] J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-damgård revisited: How to construct a hash function. In Shoup [104], pages 430–448.

[42] R. Cramer, editor. *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*. Springer, 2005.

[43] E. D. Cristofaro, C. Soriente, G. Tsudik, and A. Williams. Hummingbird: Privacy at the time of twitter. *IACR Cryptology ePrint Archive*, 2011:640, 2011.

[44] L. A. Cutillo, R. Molva, and M. Önen. Safebook: A distributed privacy preserving online social network. In *WOWMOM*, pages 1–3. IEEE, 2011.

[45] C. Delerablée. Identity-based broadcast encryption with constant size ciphertexts and private keys. In K. Kurosawa, editor, *ASIACRYPT*, volume 4833 of *Lecture Notes in Computer Science*, pages 200–215. Springer, 2007.

[46] W. Diffie, P. C. van Oorschot, and M. J. Wiener. Authentication and authenticated key exchanges. *Des. Codes Cryptography*, 2(2):107–125, 1992.

[47] Y. Dodis and N. Fazio. Public key broadcast encryption for stateless receivers. In J. Feigenbaum, editor, *Digital Rights Management Workshop*, volume 2696 of *Lecture Notes in Computer Science*, pages 61–80. Springer, 2002.

[48] L. Ducas. Anonymity from asymmetry: New constructions for anonymous hibe. In J. Pieprzyk, editor, *CT-RSA*, volume 5985 of *Lecture Notes in Computer Science*, pages 148–164. Springer, 2010.

[49] J. Dwyer. Four nerds and a cry to arms against Facebook. May 11, 2010. `http://nyti.ms/1hc60kv`. Accessed: Dec 3, 2013.

[50] N. Fazio and I. M. Perera. Outsider-anonymous broadcast encryption with sublinear ciphertexts. *IACR Cryptology ePrint Archive*, 2012:129, 2012.

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

[52] A. Fiat and M. Naor. Broadcast encryption. In D. R. Stinson, editor, *CRYPTO*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491. Springer, 1993.

[53] M. Fischetti. Data theft: Hackers attack. *Scientific American*, 305(100), 2011.

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

[55] D. M. Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In H. Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 44–61. Springer, 2010.

[56] A. Freier, P. Karlton, and P. Kocher. The Secure Sockets Layer (SSL) Protocol Version 3.0. RFC 6101 (Historic), Aug. 2011.

[57] 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.

[58] T. E. Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[59] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Secure distributed key generation for discrete-log based cryptosystems. *J. Cryptology*, 20(1):51–83, 2007.

[60] C. Gentry. Practical identity-based encryption without random oracles. In S. Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464. Springer, 2006.

[61] C. Gentry and B. Waters. Adaptive security in broadcast encryption systems. *IACR Cryptology ePrint Archive*, 2008:268, 2008.

[62] 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.

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

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

[65] M. T. Goodrich, J. Z. Sun, and R. Tamassia. Efficient tree-based revocation in groups of low-state devices. In M. K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 511–527. Springer, 2004.

[66] M. Groves. MIKEY-SAKKE: Sakai-Kasahara Key Encryption in Multimedia Internet KEYing (MIKEY). RFC 6509 (Informational), Feb. 2012.

[67] M. Groves. Sakai-Kasahara Key Encryption (SAKKE). RFC 6508 (Informational), Feb. 2012.

[68] S. Guha, K. Tang, and P. Francis. Noyb: Privacy in online social networks. In *Proceedings of the First Workshop on Online Social Networks*, WOSN '08, pages 49–54, New York, NY, USA, 2008. ACM.

[69] D. Halevy and A. Shamir. The lsd broadcast encryption scheme. In M. Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 47–60. Springer, 2002.

[70] Y. Hanaoka, G. Hanaoka, J. Shikata, and H. Imai. Identity-based hierarchical strongly key-insulated encryption and its application. In B. K. Roy, editor, *ASIACRYPT*, volume 3788 of *Lecture Notes in Computer Science*, pages 495–514. Springer, 2005.

[71] F. Hess, N. P. Smart, and F. Vercauteren. The eta pairing revisited. *IACR Cryptology ePrint Archive*, 2006:110, 2006.

[72] R. Housley. Using AES-CCM and AES-GCM Authenticated Encryption in the Cryptographic Message Syntax (CMS). RFC 5084 (Proposed Standard), Nov. 2007.

[73] 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.

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

[75] A. Kate, Y. Huang, and I. Goldberg. Distributed key generation in the wild. *IACR Cryptology ePrint Archive*, 2012:377, 2012.

[76] L. Krzywiecki, P. Kubiak, and M. Kutylowski. A revocation scheme preserving privacy. In H. Lipmaa, M. Yung, and D. Lin, editors, *Inscrypt*, volume 4318 of *Lecture Notes in Computer Science*, pages 130–143. Springer, 2006.

[77] K. Lee and D. H. Lee. New techniques for anonymous hibe with short ciphertexts in prime order groups. *TIIS*, 4(5):968–988, 2010.

[78] A. B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In D. Pointcheval and T. Johansson, editors, *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 318–335. Springer, 2012.

[79] A. B. Lewko, A. Sahai, and B. Waters. Revocation systems with very small private keys. *IACR Cryptology ePrint Archive*, 2008:309, 2008.

[80] B. Libert, K. G. Paterson, and E. A. Quaglia. Anonymous broadcast encryption: Adaptive security and efficient constructions in the standard model. In M. Fischlin, J. Buchmann, and M. Manulis, editors, *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 206–224. Springer, 2012.

[81] B. Libert and J.-J. Quisquater. Efficient revocation and threshold pairing based cryptosystems. In E. Borowsky and S. Rajsbaum, editors, *PODC*, pages 163–171. ACM, 2003.

[82] M. M. Lucas and N. Borisov. flybynight: mitigating the privacy risks of social networking. In L. F. Cranor, editor, *SOUPS*, ACM International Conference Proceeding Series. ACM, 2009.

[83] W. Luo, Q. Xie, and U. Hengartner. Facecloak: An architecture for user privacy on social networking sites. In *CSE (3)*, pages 26–33. IEEE Computer Society, 2009.

[84] L. Martin and M. Schertler. Using the Boneh-Franklin and Boneh-Boyen Identity-Based Encryption Algorithms with the Cryptographic Message Syntax (CMS). RFC 5409 (Informational), Jan. 2009.

[85] T. Matthews. Suggestion for random number generators in software. *RSA Laboratories Bulletin*, 1:1–4, 1996.

[86] C. Matyszczyk. If your account is subpoenaed, Facebook sends police, well, everything. `http://preview.tinyurl.com/facebook-subpoena`, 2012.

[87] 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.

[88] 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.

[89] D. A. McGrew and J. Viega. The security and performance of the galois/counter mode of operation (full version). *IACR Cryptology ePrint Archive*, 2004:193, 2004.

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

[91] D. Naor, M. Naor, and J. B. Lotspiech. Revocation and tracing schemes for stateless receivers. *IACR Cryptology ePrint Archive*, 2001:59, 2001.

[92] N. I. of Standards and Technology. FIPS PUB 180-4: Secure Hash Standard. RFC 6101 (Historic), 2012.

[93] T. P. Pedersen. A threshold cryptosystem without a trusted party (extended abstract). In D. W. Davies, editor, *EUROCRYPT*, volume 547 of *Lecture Notes in Computer Science*, pages 522–526. Springer, 1991.

[94] W. Post. NSA slides explain the PRISM data-collection program. June 6, 2013 `http://wapo.st/J2gkLY`. Accessed Sept. 6, 2013.

[95] M. K. R Sakai, K Ohgishi. Cryptosystem based on pairing over elliptic curve (in Japanese). In *The 2001 Symposium on Cryptography and Information Security, Oiso, Japan, January*, 2001.

[96] E. Rescorla. HTTP Over TLS. RFC 2818 (Informational), May 2000. Updated by RFC 5785.

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

[98] A. Sahai and B. Waters. Fuzzy identity based encryption. *IACR Cryptology ePrint Archive*, 2004:86, 2004.

[99] R. Sakai and J. Furukawa. Identity-based broadcast encryption. *IACR Cryptology ePrint Archive*, 2007:217, 2007.

[100] R. Sakai and M. Kasahara. Id based cryptosystems with pairing on elliptic curve. *IACR Cryptology ePrint Archive*, 2003:54, 2003.

[101] M. Scott. Miracl–multiprecision integer and rational arithmetic c/c++ library. *Shamus Software Ltd, Dublin, Ireland, URL*, 2003.

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

[103] 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.

[104] V. Shoup, editor. *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*. Springer, 2005.

[105] N. P. Smart. Efficient key encapsulation to multiple parties. In C. Blundo and S. Cimato, editors, *SCN*, volume 3352 of *Lecture Notes in Computer Science*, pages 208–219. Springer, 2004.

[106] B. Waters. Efficient identity-based encryption without random oracles. In Cramer [42], pages 114–127.

[107] B. Waters. Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In S. Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.

[108] A. Whitten and J. D. Tygar. Why johnny can't encrypt: A usability evaluation of pgp 5.0. In G. W. Treese, editor, *USENIX Security*. USENIX Association, 1999.

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

[110] S. Yu, K. Ren, and W. Lou. Attribute-based on-demand multicast group setup with membership anonymity. *Computer Networks*, 54(3):377–386, 2010.

[111] X. Zhang and K. Wang. Fast symmetric pairing revisited. In Cao and Zhang [36], pages 131–148.

Master thesis filing card

Student: Stijn Meul

Title: Practical Identity-Based Encryption for Online Social Networks

UDC: 621.3

Abstract:

Nowadays Online Social Networks (OSNs) constitute an important and useful communication channel. At the same time, coarse-grained privacy preferences protect the shared information insufficiently. Cryptographic techniques can provide interesting mechanism to protect privacy of users in OSNs. However, this approach faces several issues, such as, OSN provider acceptance, user adoption, key management and usability. We suggest a practical solution that uses Identity Based Encryption (IBE) to simplify key management and enforce confidentiality of data in OSNs. Moreover, we devise an outsider anonymous broadcast IBE scheme to disseminate information among multiple users, even if they are not using the system. Finally, we demonstrate the viability and tolerable overhead of our solution via an open-source prototype.

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