Combining ABCs with ABE

T. R. van de Kamp

Combining ABCs with ABE

Privacy-Friendly Key Generation

for

Smart Card Based
Attribute-Based Encryption

UNIVERSITY OF TWENTE.

2014

Master's thesis Computer Science, *Services, Cybersecurity, and Safety* research group.
Faculty of Electrical Engineering, Mathematics and Computer Science at the University of Twente.

UNIVERSITY OF TWENTE.

This thesis was in part done in the context of the THeCS project of the COMMIT program.

Author: T. R. van de Kamp
Main supervisors: dr. M. H. Everts (University of Twente), dr. J. H. Hoepman (Radboud University), G. Alpár (Radboud University)
Graduation committee: dr. M. H. Everts (University of Twente), dr. A. Peter (University of Twente), dr. J. H. Hoepman (Radboud University), G. Alpár (Radboud University)

Abstract

Attribute-Based Credential (ABC) schemes provide a privacy-friendly method to perform authentication. In such a system the user does not necessarily have to identify himself, but may reveal only partial information about him, i.e., attributes the user possesses. The use of this technique is a proper solution for several kinds of authentication where no full identification is required, e.g., buying liquor at the liquor store or opening the door to an office building. However, using Attribute-Based Encryption (ABE) provides some advantages of ABCs in the case of data protection. Ciphertext-Policy ABE schemes allow a user to define an access policy over an encrypted file, so that only the individuals possessing the right attributes can decrypt the file. The data authorization takes place when a user tries to decrypt a file; data access does not involve an on-line party such as would be required by an ABC system.

A smart card implementation of the Identity Mixer (idemix) credentials system exists, making it feasible to implement the ABC system. Little progress has been made to create an ABE scheme that is suitable to run in a similar environment. Most ABE schemes require computationally complex decryption algorithms that take too much time to run on current smart cards. Moreover, many multi-authority ABE schemes violate the user's privacy by requiring the user to reveal his unique identifier, enabling authorities to profile its users.

We create an overview of different types of ABE schemes and describe several schemes in terms of security and efficiency. Using our classification, we select the [LW11] decentralized multi-authority ABE scheme that we can adapt to meet our requirements. We propose a Blind Key Generation protocol that provides a way to do privacy-friendly key issuance without the user having to reveal his identifier. We prove this protocol to be secure against three different types of attackers using the security definitions introduced by Green and Hohenberger [GH07]. Additionally, we propose an Off-card Decrypt protocol. This protocol enables us to outsource some of the most complex operations to a trusted device, yet safely store the decryption keys on the smart card and never reveal them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since the start of the Information Age, we use more and more digital equipment in our day-to-day life. This digitalization certainly has advantages over their analogue counterparts, as it often makes operations easier or user-friendlier. However, with the use of such new techniques, additional negative side effects are easily introduced. In particular privacy is an often overlooked aspect. A well designed digital system is built from the start with all the possible privacy concerns in mind.

1.1 Background

This thesis is about designing a practical and privacy-friendly encryption scheme for the IRMA ecosystem. The work combines Attribute-Based Credential (ABC) techniques with Attribute-Based Encryption (ABE) to create such a scheme.

1.1.1 Privacy-Friendly Authentication

The IRMA (*I Reveal My Attributes*) project[1] has developed an efficient smart card implementation of a privacy-friendly authentication system. Most current authentication systems require the user to identify oneself with a unique name or number. However, we can think of many cases where the *identity* of a person is actually irrelevant, but only certain *properties* of the user are relevant. For example, a student might only need to show that he is a student to get a discount at a museum, yet another person only needs to show his museum season ticket to enter the museum. So, instead of revealing your identity it can be more relevant to reveal *attributes* of yourself. These attributes can show what you are, e.g., a student, or what you possess, e.g., a museum season ticket. Moreover, attributes can be identifying, e.g., a Social Security Number, or anonymous, e.g., age or hometown.

With the *IRMA card* one can authenticate oneself by selectively revealing attributes. The near field communication (NFC) enabled smart card communicates with another device to prove the possession of attributes. One could use the card in a liquor store to show that he is over eighteen—old enough to buy alcohol—by holding the card in front of a terminal computer that will flash a

[1] `https://www.irmacard.org`

green light if one is old enough. In addition, an IRMA card could be used to log in on a website. If the user does not own a NFC card reader, this process can be facilitated with an NFC enabled smart phone. The card reader or smart phone acts as a communication medium to establish a connection from the card to the web server. Whatever scenario you might think of, the cryptographic techniques used by the IRMA project ensure that the verifying party learns nothing more than whether the user possesses the attribute or not.[2]

1.1.2 Encryption

Public-key cryptography makes it possible to do asymmetric cryptography. Using the *public key*, anyone can encrypt messages that can only be decrypted if one possesses the *private key* or decryption key. A special form of public-key cryptography, called *Attribute-Based Encryption*, allows users to decrypt messages if their decryption key satisfies the access policy defined in the ciphertext. Such an access policy could for example specify that you should be over eighteen *and* live in the Netherlands to be able to decrypt the ciphertext.

Encryption is a valuable addition to the IRMA ecosystem. By using encryption, data can be protected against unauthorized access without the need of an on-line verifier authorizing data requests. Such an on-line verifier is needed when we use an ABC scheme.

1.2 Goals

To extend the possibilities of an IRMA card, encryption can be added to the IRMA ecosystem. The main objective of this Master's thesis is to create a suitable encryption scheme for the IRMA ecosystem. For a scheme to be suitable in the IRMA ecosystem, it should make use of attributes, be efficient enough to run on a smart card, and be privacy-friendly.

With the encryption scheme added to the IRMA ecosystem, we want to be able to decrypt messages using an IRMA card. We want to securely store the user's decryption keys on his smart card and use these keys to decrypt the ciphertext he is allowed to. We do not explicitly require the encryption algorithm to run on the smart card: it may run on any device capable of doing the computations.

1.3 Approach

First we will argue why ABE is a good candidate for an encryption scheme in the IRMA ecosystem. Next, we will discuss how ABE works and which types of ABE schemes exist, to allow us to pick a good ABE scheme that suits the IRMA ecosystem. We will select a scheme and tackle any major drawbacks of the scheme to create a modified version of the ABE scheme that is privacy-friendly and is efficient enough to run on a smart card. When we adapt the ABE scheme, we will use other IRMA techniques that are already available in the ecosystem.

[2] It is possible that other communication layers reveal more information. However, this would also be the case if other techniques are used.

1.4 Document Structure

We will start by providing the necessary mathematical and cryptographic background in Chapter 2. Chapters 3 and 4 explain how the ABC and ABE systems work. The different types of ABE schemes are discussed in Chapter 5. At the end of the chapter we will have an overview of the current ABE schemes and we will be able to choose a specific scheme that will match our needs. In Chapters 6 and 7 we adapt the scheme to create a practical and privacy-friendly encryption scheme for the IRMA ecosystem.

The conclusions and recommendations are discussed in Chapter 8. The final chapter provides a summary of the thesis.

Chapter 2

Mathematical Background

This chapter provides all the necessary preliminary work to understand the remainder of the thesis.

2.1 Notations

The notation $r \in_R S$ is used for expressing that r is picked uniformly at random from the finite set S. A formula with a question mark above an operator, e.g., $\stackrel{?}{=}$, denotes that the formula should be true if both the left-hand side and the right-hand side are properly constructed. A return value $\perp$ of a protocol indicates that the protocol rejected the input. $\mathcal{S}$ represents the set of all attributes. The set of attributes belonging to a single authority A_j is defined as $\mathcal{S}_{A_j} \subseteq \mathcal{S}$. $\mathcal{S}_U$ denotes the set of attributes that the user U possess.

Additional new notations are introduced where they are needed.

2.2 Commitment

Alice regularly enters into a heated discussion with Bob. When Alice and Bob cannot succeed in convincing each other using their argumentation, they usually decide to flip a coin. The coin toss will then determine who has won the argument. However, Alice and Bob sometimes debate over the telephone. If Alice would flip a coin, Bob won't trust Alice to tell the actual outcome, and vice versa. Luckily, cryptographic techniques can be used to settle their disputes over the telephone. Coin flipping by telephone works as follows. Both Alice and Bob flip a (physical) coin. Alice won't tell Bob the outcome of her coin flip, but *commits* to the outcome and shows Bob her commitment. Bob now reveals his outcome of the coin flip to Alice. Finally, Alice responds by revealing her outcome of the coin flip and telling the *opening* of her commitment so that Bob can verify that she initially committed to the same value. Alice wins the argument if both outcomes were the same, otherwise Bob wins.

Next, we give a formal definition of a commitment scheme.

Definition 1 (Commitment)**.** A commitment scheme consists of three algorithms, Setup, Commit, and Reveal. Commit and Reveal are both protocols between two parties, the sender and receiver.

Setup This algorithm determines the public parameters of the system.

Commit The sender commits to a message M by choosing a random value u and calculating the commitment $c = \mathsf{Commit}(M, u)$.

Reveal The sender reveals the value he committed to by publishing the message M and the opening u. The receiver can verify the commitment c he received by checking if $c \stackrel{?}{=} \mathsf{Commit}(M, u)$ is indeed the committed value.

There are two properties defined on a commitment scheme that must be satisfied. A commitment must be *binding*, meaning that the sender—Alice in our example—cannot reveal another message than she originally committed to. In addition, a commitment must be *hiding*, meaning that the receiver—Bob—cannot learn to what message the sender committed to. We subdivide both properties into two types, computationally binding/hiding versus information-theoretically binding/hiding. A scheme is *computationally binding* if no polynomial time algorithm exists that generates the same commitment for two different messages. That is, there is no efficient algorithm that can find M_1, M_2, u_1, and u_2, such that $\mathsf{Commit}(M_1, u_1) = \mathsf{Commit}(M_2, u_2)$ and $M_1 \neq M_2$. We call a scheme *information-theoretically binding* if it is impossible—even with a computer of unlimited power—to generate such a commitment. A scheme is *computationally hiding* if the statistical distribution of all possible committed messages are computationally indistinguishable from each other. The strongest type of hiding is the *information-theoretical hiding*, meaning that those distributions must be statistically indistinguishable. We note that the *impossibility result* tells us that a scheme that is both information-theoretical binding *and* information-theoretical hiding cannot exists.

The commitment scheme described below is computational binding (under the Discrete Logarithm assumption), and information-theoretical hiding.

Scheme (Pedersen's commitment scheme [Ped92])**.** Pedersen's commitment scheme can be used to commit to a value $x \in \mathbb{Z}_q$.

Setup Let p and q be two large primes, such that $q \mid p-1$. Let $\langle g \rangle$ be the unique subgroup of order q of group $\mathbb{Z}_p^*$. Pick $h \in_R \langle g \rangle \setminus \{1\}$ such that $\log_g h$ is unknown to any party.

Commit Pick a number $u \in_R \mathbb{Z}_q$. The commitment to value x is $c = g^u h^x$.

Reveal The commitment can be opened by revealing the values x and u. The receiver can verify the commitment by checking $c \stackrel{?}{=} g^u h^x$.

2.3 Zero-Knowledge Proofs of Knowledge

Assume Peggy wants to convince Victor that she knows a secret value. She does not want Victor to learn her secret value itself or any information about it. For example, Peggy may have generated a public-private key pair and wants to prove to Victor that she knows the private key of the corresponding public key. In order to do so, Victor could encrypt a random message using the public key, send the ciphertext over to Peggy and ask her to decrypt the message. If Peggy could decrypt the message she probably has the private key. However, Peggy now functions as a decryption oracle that can be misused in a chosen ciphertext

attack. Luckily, there are other cryptographic techniques that provide a better solution.

A *zero-knowledge proof* is a protocol between a prover, Peggy, and a verifier, Victor. It can be used to prove knowledge of a secret value—called a *witness*—and, at the same time, ensure that no information about this witness is leaked. We will focus on a special class of zero-knowledge proofs, called Σ-*protocols*. Σ-protocols have the following fixed protocol structure. The prover starts by committing to a *nonce*, a freshly chosen, random number, and sends this commitment t to the verifier. The verifier returns a challenge c to the prover, who finally responds with a value s. We call this conversation between the prover and the verifier *accepting*, if the verifier accepts the proof provided by the prover. Besides this imposed structure, Σ-protocols also satisfy three properties. First, if both parties act honestly, then verification will always succeed. Second, if a cheating prover can create an accepting conversation for more than one challenge, then she basically knows the witness. Finally, the zero-knowledge property: nothing is learned from obtaining accepting conversations because every accepting conversation can be simulated by any party on its own.

Before we formally state the definition of a Σ-protocol, we will mathematically define what a prover wants to prove to the verifier. Consider a binary relation $R \subseteq V \times W$ consisting of pairs (v, w) where $v \in V$ denotes a public input and $w \in W$ denotes a private witness. In a zero-knowledge protocol the prover wants to prove knowledge of the witness w, that satisfies $(v, w) \in R$ for the public input v.

Definition 2 (Σ-protocol [CDN01])**.** A Σ-protocol is a protocol with the previously described conversation (t, c, s) satisfying the following three properties.

Completeness An honest verifier always accepts as long as the prover uses a witness w such that $(v, w) \in R$.

Special soundness A cheating prover can correctly answer only a single challenge. More precisely, there exists an efficient algorithm that computes (*extracts*) witness w such that $(v, w) \in R$ from any public input v and any pair of accepting conversations (t, c, s), (t, c', s') where $c \neq c'$.

Special honest-verifier zero-knowledgeness When given a challenge c, accepting conversations can be created (*simulated*) with the same probability distribution of a conversation between the honest prover and honest verifier where the challenge c is used.

An example of a Σ-proof is shown in Figure 2.1. The figure displays Schnorr's protocol [Sch91]. Schnorr's protocol uses a multiplicative group of prime order q, generated by g, to prove knowledge of a discrete logarithm. The prover proves that he knows a value α such that $A = g^\alpha$.

Lemma 1. *Schnorr's protocol [Sch91] as depicted in Figure 2.1 is a Σ-protocol.*

Proof. We will prove the three properties that make this protocol a Σ-protocol.

Completeness The protocol is complete:

$$g^s = g^{r+c\alpha} = g^r (g^\alpha)^c = tA^c.$$

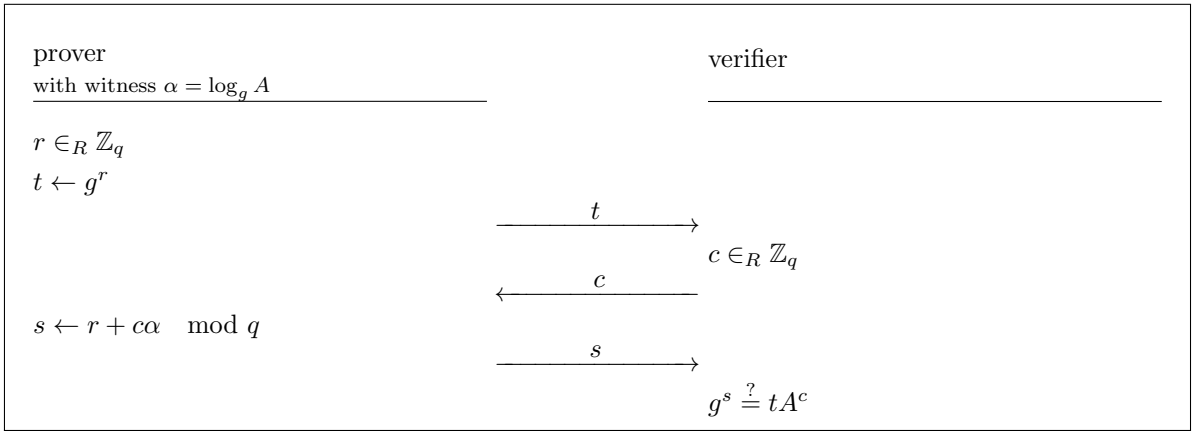

Figure 2.1: Schnorr's protocol [Sch91].

Special soundness Given two accepting conversations (t, c, s), (t, c', s') with $c \neq c'$, we can extract the witness α by computing

$$\alpha = \frac{s - s'}{c - c'} \mod q.$$

This equation holds because we have $g^s = tA^c$ and $g^{s'} = tA^{c'}$ which imply

$$g^{s-s'} = A^{c-c'} \qquad \text{and} \qquad A = g^{\frac{s-s'}{c-c'}}.$$

Special honest-verifier zero-knowledgeness The stochastic distribution of an accepting conversation between an honest prover and an honest verifier, given an arbitrary challenge c, is

$$\big\{(t, c, s) : \ r \in_R \mathbb{Z}_q, t \leftarrow g^r, s \leftarrow r + c\alpha \mod q\big\}.$$

This notation denotes how the values t, c, and s are chosen, and how they depend on the randomness. We note that valid conversations (t, c, s) occur with probability $\frac{1}{q}$.

A simulator can simulate, again for arbitrary challenge c, a conversation with distribution

$$\big\{(t, c, s) : \ s \in_R \mathbb{Z}_q, t \leftarrow g^s A^{-c}\big\}.$$

For valid conversations (t, c, s) the outcome of both the 'real' as well the simulated conversations will occur with probability $\frac{1}{q}$. □

A neat feature of Σ-protocols is that new, more advanced, Σ-protocols can be systematically constructed from other Σ-protocols. By combining protocols in a specific way we can obtain zero-knowledge proofs of knowledge for more complex relations. For example, the AND-composition enables us to prove that we know multiple witnesses. The OR-composition allows us to prove that we know one witness or another. Several other compositions exist, most importantly the

parallel composition and the EQ-composition (used to express equality of witnesses). The structure of the resulting Σ-protocol of such a composition of two Σ-protocols will remain the same. However, instead of one commitment t several commitments $\mathbf{t}$ may be required, just like multiple responses $\mathbf{s}$ can occur.

Because Σ-protocols can be composed in such a systematic way, we can use a simpler notation for the protocols. Camenisch and Stadler [CS97] introduce an efficient notation where only the witnesses and *relations* are listed. This notation has the advantage that we can define a complete protocol on a single line, without specifying the exact messages that are sent. The Σ-protocol in Figure 2.1 can be written as $\mathrm{PK}\{(\alpha) : A = g^{\alpha}\}$. As a convention, we will write all private witnesses with Greek symbols and all other public values in Roman script. For example, the notation $\mathrm{PK}\{(\alpha, \beta, \gamma) : A = g^{\alpha}h^{\gamma} \wedge B = h^{\beta}\}$ denotes a zero-knowledge proof of knowledge of witnesses α, β, and γ, such that $A = g^{\alpha}h^{\gamma}$ and $B = h^{\beta}$ hold.

2.3.1 Fiat–Shamir Heuristic

Σ-protocols are not very efficient due to their interactive nature. It takes time to send messages back and forth, but more importantly, it requires the prover and verifier to be on-line at the same moment. Fiat and Shamir [FS87] propose a heuristic to turn an interactive zero-knowledge proof into a non-interactive one. The Fiat–Shamir heuristic turns a zero-knowledge protocol into a signature of knowledge. A *signature of knowledge* is a signature that can be send to a verifier in order to provide a zero-knowledge proof of knowledge. The signature has the additional property that it can be stored and the same signature can be verified by other verifiers. When the Fiat–Shamir heuristic is applied to a Σ-protocol, we will refer to it as a Σ-*proof*.

A zero-knowledge protocol can be turned into a signature of knowledge by defining the challenge value c as the hash value of the commitments $\mathbf{a}$ together with a message M. The Fiat–Shamir heuristic is described in Algorithm 1.

Algorithm 1 (Fiat–Shamir Heuristic)**.** The Fiat–Shamir heuristic can be applied to any Σ-protocol. The resulting signature of knowledge will be a signature on a message M.

Signature generation The prover determines commitments $\mathbf{t}$ exactly the same way as in a Σ-protocol. The challenge value c is calculated by applying a cryptographic hash function $\mathcal{H}$ to the commitments $\mathbf{t}$ and the message M, $c \leftarrow \mathcal{H}(\mathbf{t}, M)$. Now, the responses $\mathbf{s}$ can be determined in the same way as in the Σ-protocol.

The signature on M is the pair $(c, \mathbf{s})$.

Signature verification The verifier can determine all the commitments $\mathbf{t}$ using the same simulation technique that we have used in the special honest-verifier zero-knowledge proofs. The verifier gets the fixed challenge c and the responses $\mathbf{s}$ as input. He accepts the proof if the challenge value c is the same as calculated by the verifier using the, by the simulation technique determined, commitments $\mathbf{t}$. The verifier checks $c \stackrel{?}{=} \mathcal{H}(\mathbf{t}, M)$.

The above described heuristic will become clearer by an example in which we turn a Σ-protocol into a Σ-proof. In a Σ-proof the message M is replaced by the public input v of the relation R.

Example 2.1 (Σ-proof)**.** We apply the Fiat–Shamir heuristic to the Σ-protocol of Figure 2.1. The prover executes the Proof generation, the verifier runs the Proof verification. The prover wants to prove knowledge of witness α using public input $v = A = g^\alpha$.

Proof generation The prover picks a $r \in_R \mathbb{Z}_q$ and sets $t = g^r$. The challenge is calculated as $c = \mathcal{H}(t, A)$ and the response is set to $s = r + c\alpha$.

The Σ-proof on the relation $A = g^\alpha$ is the pair (c, s).

Proof verification The verifier will accept the proof if

$$c = \mathcal{H}(g^s A^{-c}, A)$$

holds. Note that this equation holds, as long as $t = g^s A^{-c} = g^{r+c\alpha} A^{-c} = g^r (g^\alpha)^c A^{-c} = g^r$ will hold.

There exists a simple notation for signatures of knowledge, just like for zero-knowledge proofs. For example, a signature on a message M based on the interactive zero-knowledge protocol $\mathrm{PK}\{(\alpha, \beta) : A = g^\alpha \wedge B = h^\beta\}$ can be written as $\mathrm{SPK}\{(\alpha, \beta) : A = g^\alpha \wedge B = h^\beta\}(M)$.

2.4 Secret Sharing

Secret sharing schemes have the goal to split up a secret into several parts, where those separate parts do not leak any information on the secret. A sharing scheme consists of two algorithms, a distribution or Setup algorithm, and a reconstruction or Pooling algorithm. The Setup algorithm splits the secret into several *shares* that can be used in the Pooling algorithm to reconstruct the secret. A simple secret sharing scheme is the unanimous consent control by modular addition scheme.

Scheme (Unanimous Consent Control by Modular Addition [MOV96, §12.7.1])**.** This scheme can split a secret s, where $0 \leq s < m$, in n shares.

Setup Create $n - 1$ shares, s_i, by assigning them random numbers from $\mathbb{Z}_m$; $\forall i \in \{1, \ldots, n-1\} : s_i \in_R \mathbb{Z}_m$. The final share is defined as

$$s_n = s - \sum_{i=1}^{n-1} s_i \mod m.$$

Pooling The secret can be reconstructed using modular addition of all shares,

$$s = \sum_{i=1}^{n} s_i \mod m.$$

More advanced secret sharing schemes are *threshold* sharing schemes. Such a scheme divides a secret into n shares, where only a subset of at least t shares, with $t \leq n$, is required to reconstruct the secret. Shamir [Sha79] proposes the first threshold sharing scheme.

Scheme (Shamir's Secret Sharing Scheme [Sha79])**.** A secret s can be split into n shares. The secret can be reconstructed with t or more shares.

Setup Let $p > \max(s, n)$ be a prime. Set $a_0 = s$ and select $t-1$ coefficients $a_1, \ldots, a_{t-1} \in_R \{0, \ldots, p-1\}$. Define the $t-1$ degree polynomial $q(x) = \prod_{i=0}^{t-1} a_i x^i$. Finally, compute $s_i = q(i) \mod p$ for $i \in \{1, \ldots, n\}$.

The shares are the tuples (i, s_i) for $i \in \{1, \ldots, n\}$.

Pooling The polynomial $q(x)$ can be reconstructed with t or more shares. Let each distinct share equal (x_i, y_i) for $i \in \{1, \ldots, t\}$. Define the Lagrange coefficients $\Delta_i(x) = \prod_{1 \leq j \leq t; i \neq j} \frac{x - x_j}{x_i - x_j}$, as well as the polynomial $q'(x) = \sum_{i=1}^{t} y_i \cdot \Delta_i(x)$.

The secret can be now be determined, we find $q'(0) = q(0) = a_0 = s \pmod p$.

2.5 Algebraic Preliminaries

In this thesis we use several abstract algebraic principles. The RSA related algebra is needed for discussing Attribute-Based Credentials. The various definitions of a bilinear map are needed for the discussion of Attribute-Based Encryption.

2.5.1 RSA Related Algebra

Definition 3 (Safe primes [CL03])**.** A prime number p is a *safe prime* if there is another prime number p' (called a *Sophie Germain prime*) such that $p = 2p' + 1$.

Definition 4 (Quadratic residues)**.** An integer a is a *quadratic residue* modulo n if there exists an integer $b \in \mathbb{Z}_n^*$ such that $b^2 \equiv a \pmod n$. The *set of quadratic residues* modulo n, $QR_n \subseteq \mathbb{Z}_n^*$, contains all the quadratic residues modulo n.

2.5.2 Bilinear Map

We will formally state the definition of a bilinear map, as this will later form an important building block for the Attribute-Based Encryption schemes.

Definition 5 (Bilinear map using a prime order group [SW05; CC09])**.** Let $\mathbb{G}_1$, $\mathbb{G}_2$ be cyclic multiplicative groups of prime order p, generated by g_1 and g_2 respectively. We say $(\mathbb{G}_1, \mathbb{G}_2)$ has an admissible asymmetric bilinear map, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, into $\mathbb{G}_T$ if the following two conditions hold.

1. The map is bilinear; $\forall a, b \in \mathbb{Z}_p : \ e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.
2. The map is non-degenerate; this requires $e(g_1, g_2) \neq 1$.

If $\mathbb{G}_1 = \mathbb{G}_2$ and $g_1 = g_2$, then we say that e is a symmetric bilinear map. Unless otherwise stated, we will refer to an admissible and computationally efficient symmetric bilinear map simply as a bilinear map.

We note that, in practice, $\mathbb{G}_1$ and $\mathbb{G}_2$ are elliptic curve groups and $\mathbb{G}_T$ is an extension field, so all groups are abelian.

Definition 6 (Bilinear map using a composite order group [LW11])**.** Let $\mathbb{G}$, $\mathbb{G}_T$ be cyclic multiplicative groups of composite order $\Gamma = p_1 p_2 p_3$, where p_1, p_2, and p_3 are distinct primes. The map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a composite order bilinear map if the following two conditions hold.

1. The map is bilinear; $\forall g, h \in \mathbb{G},\ a, b \in \mathbb{Z}_\Gamma :\ e(g^a, h^b) = e(g, h)^{ab}$.
2. The map is non-degenerate; this requires a generator g such that the order of the element $e(g, g) \in \mathbb{G}_T$ equals Γ, the order of group $\mathbb{G}_T$.

Lewko and Waters [LW10] note that when g is an element of the subgroup of order p_i, and h is an element of a different subgroup of order p_j $(i \neq j)$, then $e(g, h) = 1$ will hold. To show this, they suppose that g_1 and g_2 are elements of the subgroup of order p_1 ($\mathbb{G}_{p_1}$) and the subgroup of order p_2 ($\mathbb{G}_{p_2}$), respectively. Now, let g be a generator of the group $\mathbb{G}$. Note that $g^{p_2 p_3}$ will be a generator of the subgroup $\mathbb{G}_{p_1}$, $g^{p_1 p_3}$ a generator of $\mathbb{G}_{p_2}$, and $g^{p_1 p_2}$ of $\mathbb{G}_{p_3}$. We can thus write $g_1 = (g^{p_2 p_3})^{r_1}$ and $g_2 = (g^{p_1 p_3})^{r_2}$ for some r_1 and r_2. We then have

$$e(g_1, g_2) = e(g^{r_1 p_2 p_3}, g^{r_2 p_1 p_3}) = e(g^{r_1}, g^{r_2 p_3})^{p_1 p_2 p_3} = 1.$$

2.6 Security Definitions

This last section of the chapter contains the definition of an access structure and collects several assumptions that some results rely on. A short introduction to attack models is provided at the end of this section.

2.6.1 Access Structure

Access structures are used to define which users have access to which resources. In the case of attribute-based authentication, attributes determine the authorization level of the user. An access structure can be regarded as a collection of sets of attributes. Each single set describes which attributes are needed to be granted access. As long as the user's attributes satisfy at least one set in the collection, the user is granted access.

There are two kinds of access structures: monotonic and non-monotonic. Monotonic access structures ensure that whenever a user would be granted access based on a subset of his attributes, he will be granted access based on all his attributes. This means that no negations of attributes are possible. Non-monotonic access structures do allow such negation of attributes. Here, the possession of an extra attribute may deny you access. For example, having the attribute 'name:Joe' will cause you to fail to gain access to a resource associated with the access structure "role:manager AND ¬name:Joe".

The following definition defines an access structure formally. We will use the terms access policy and access structure interchangeably.

Definition 7 (Access Structure [Bei96])**.** Let $\{P_1, \ldots, P_n\}$ be a set of parties. A collection $\mathcal{A} \subseteq 2^{\{P_1, \ldots, P_n\}}$ is monotone if $\forall Y, Z :\ Y \in \mathcal{A}$ and $Y \subseteq Z \implies Z \in \mathcal{A}$.

An *access structure* (respectively, *monotone access structure*) is a collection (respectively, monotone collection) $\mathcal{A}$ of non-empty subsets of $\{P_1, \ldots, P_n\}$, i.e., $\mathcal{A} \subseteq 2^{\{P_1, \ldots, P_n\}} \setminus \{\emptyset\}$. The sets in $\mathcal{A}$ are called the *authorized sets*, and the sets not in $\mathcal{A}$ are called the *unauthorized sets*.

We say that the set of attributes $\mathcal{S}_U$ of user U satisfies the access structure $\mathcal{A}$, written as $\mathcal{S}_U \models \mathcal{A}$, if and only if $\mathcal{S}_U$ is in the authorized sets.

2.6.2 Complexity Assumptions

Security games are a commonly used method to prove that a cryptographic scheme is secure. Such a proof will show that, when an adversary is able to gain a non-negligible advantage in the security game, he is also able to gain a non-negligible advantage in solving a hard problem. We look at several problems that we will assume hard to solve for polynomial-time adversaries. Instead of formulating a hard problem, we state an *assumption* that an adversary $\mathcal{A}$ cannot solve the problem in polynomial-time.

Two well-known assumptions are the Discrete Logarithm (DL) assumption and the Decisional Diffie–Hellman (DDH) assumption.

Definition 8 (Discrete Logarithm assumption)**.** Given a random group element $h = g^x$ from the group $\langle g \rangle$, it is hard to compute the value x.

Definition 9 (Decisional Diffie–Hellman assumption)**.** Given two arbitrary group elements $A = g^a$ and $B = g^b$ from the group $\langle g \rangle$, it is hard to distinguish g^{ab} from a random group element $Z = g^z$.

More formally: let $\mathcal{G} = (g, g^a, g^b)$; the advantage of an adversary $\mathcal{A}$ in distinguishing g^{ab} from Z,

$$\left|\Pr[\mathcal{A}(\mathcal{G}, g^{ab}) = 1] - \Pr[\mathcal{A}(\mathcal{G}, Z) = 1]\right|,$$

is negligible.

The type of ABC scheme that the IRMA project uses, relies on the *Strong RSA assumption*. This assumption is stronger than the *RSA assumption*, as it allows the adversary to choose *any* $e > 1$ to calculate the eth root of h.

Definition 10 (Strong RSA assumption [CL03])**.** Given a random group element $h \in \mathbb{Z}_n^*$ of an RSA group with modulus n, it is hard to find values $e > 1$ and g such that $g^e \equiv h \pmod{n}$.

More formally: let $\mathcal{G} = (n, h)$; the advantage of an adversary $\mathcal{A}$ in finding values e and g,

$$\Pr[(g, e) \leftarrow \mathcal{A}(\mathcal{G}) : \ e > 1 \wedge g^e \equiv h \pmod{n}],$$

is negligible.

One of the standard assumptions in ABE proofs is the *Decisional Bilinear Diffie–Hellman assumption*. The Decisional Bilinear Diffie–Hellman (DBDH) assumption is used by many ABE schemes.

Definition 11 (Decisional Bilinear Diffie–Hellman assumption [SW05])**.** Given a generator g of the bilinear group $\mathbb{G}$ of prime order p and three arbitrary group elements $A = g^a$, $B = g^b$, and $C = g^c$, it is hard to distinguish $e(g, g)^{abc}$ from a random group element $Z = e(g, g)^z$.

More formally: let $\mathcal{G} = (p, \mathbb{G}, \mathbb{G}_T, e, A, B, C)$; the advantage of an adversary $\mathcal{A}$ in distinguishing $e(g, g)^{abc}$ from Z,

$$\left|\Pr[\mathcal{A}(\mathcal{G}, e(g, g)^{abc}) = 1] - \Pr[\mathcal{A}(\mathcal{G}, Z) = 1]\right|,$$

is negligible.

The following four assumptions have to hold so that we can securely use an ABE scheme that plays an important role in this thesis. We use the notation $\mathbb{G}_x$ to denote the subgroup of order x of the composite order bilinear group $\mathbb{G}$.

Definition 12 (Subgroup decision assumption for 3 primes [LW11])**.** Given a composite bilinear group of order $\Gamma = p_1p_2p_3$ with the map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ and a generator g_1 of the subgroup $\mathbb{G}_{p_1}$, it is hard to distinguish a generator h of the group $\mathbb{G}$ from a generator h_1 of the subgroup $\mathbb{G}_{p_1}$.

More formally: let $\mathcal{G} = (\Gamma, \mathbb{G}, \mathbb{G}_T, e, g_1)$; the advantage of an adversary $\mathcal{A}$ in distinguishing h from h_1,

$$\left|\Pr[\mathcal{A}(\mathcal{G}, h) = 1] - \Pr[\mathcal{A}(\mathcal{G}, h_1) = 1]\right|,$$

is negligible.

Definition 13. (From [LW11].) Given a composite bilinear group of order $\Gamma = p_1p_2p_3$ with the map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ and a generator g_1 of the subgroup $\mathbb{G}_{p_1}$, a generator g_3 of the subgroup $\mathbb{G}_{p_3}$, and a generator $g_{1,2}$ of the subgroup $\mathbb{G}_{p_1p_2}$, it is hard to distinguish a generator h_1 of the subgroup $\mathbb{G}_{p_1}$ from a generator $h_{1,2}$ of the subgroup $\mathbb{G}_{p_1p_2}$.

More formally: let $\mathcal{G} = (\Gamma, \mathbb{G}, \mathbb{G}_T, e, g_1, g_3, g_{1,2})$; the advantage of an adversary $\mathcal{A}$ in distinguishing h_1 from $h_{1,2}$,

$$\left|\Pr[\mathcal{A}(\mathcal{G}, h_1) = 1] - \Pr[\mathcal{A}(\mathcal{G}, h_{1,2}) = 1]\right|,$$

is negligible.

Definition 14. (From [LW11].) Given a composite bilinear group of order $\Gamma = p_1p_2p_3$ with the map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ and a generator g_1 of the subgroup $\mathbb{G}_{p_1}$, a generator $g_{1,3}$ of the subgroup $\mathbb{G}_{p_1p_3}$, and a generator $g_{2,3}$ of the subgroup $\mathbb{G}_{p_2p_3}$, it is hard to distinguish a generator $h_{1,2}$ of the subgroup $\mathbb{G}_{p_1p_2}$ from a generator $h_{1,3}$ of the subgroup $\mathbb{G}_{p_1p_3}$.

More formally: let $\mathcal{G} = (\Gamma, \mathbb{G}, \mathbb{G}_T, e, g_1, g_{1,3}, g_{2,3})$; the advantage of an adversary $\mathcal{A}$ in distinguishing $h_{1,2}$ from $h_{1,3}$,

$$\left|\Pr[\mathcal{A}(\mathcal{G}, h_{1,2}) = 1] - \Pr[\mathcal{A}(\mathcal{G}, h_{1,3}) = 1]\right|,$$

is negligible.

Definition 15. (From [LW11].) Let $a, b, c, d \in_R \mathbb{Z}_\Gamma$. Now, given a composite bilinear group of order $\Gamma = p_1p_2p_3$ with the map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ and a generator g_1 of the subgroup $\mathbb{G}_{p_1}$, a generator g_2 of the subgroup $\mathbb{G}_{p_2}$, a generator g_3 of the subgroup $\mathbb{G}_{p_3}$, and the numbers $g_1^a, (g_1g_3)^b, g_1^c, g_1^{ac}g_3^d$, it is hard to distinguish the value $e(g_1, g_1)^{abc}$ from a random generator h of the group $\mathbb{G}$.

More formally: let $\mathcal{G} = (\Gamma, \mathbb{G}, \mathbb{G}_T, e, g_1, g_2, g_3, g_1^a, (g_1g_3)^b, g_1^c, g_1^{ac}g_3^d)$; the advantage of an adversary $\mathcal{A}$ in distinguishing $e(g_1, g_1)^{abc}$ from h,

$$\left|\Pr[\mathcal{A}(\mathcal{G}, e(g_1, g_1)^{abc}) = 1] - \Pr[\mathcal{A}(\mathcal{G}, h) = 1]\right|,$$

is negligible.

The next complexity assumption is needed for one of our own constructions later on.

Definition 16. Let $\rho, \bar{\rho}, \varsigma, z \in_R \mathbb{Z}_\Gamma$. Now, given a group of composite order $\Gamma = p_1 p_2 p_3$ and a generator g_1 of the subgroup $\mathbb{G}_{p_1}$, a generator h of the entire group $\mathbb{G}$, and the value $X = g_1^\rho h^\varsigma$, it is hard to distinguish an element of the subset generated by choosing different ρ in $g_1^\rho h^\varsigma$ from a random group element $Z = h^z$.

More formally: let $\mathcal{G} = (\Gamma, \mathbb{G}, g_1, h, X)$; the advantage of an adversary $\mathcal{A}$ in distinguishing $g_1^{\bar{\rho}} h^\varsigma$ from Z,

$$\left|\Pr[\mathcal{A}(\mathcal{G}, g_1^{\bar{\rho}} h^\varsigma) = 1] - \Pr[\mathcal{A}(\mathcal{G}, Z) = 1]\right|,$$

is negligible.

Several other assumptions are used by other ABE schemes. For example, the scheme by Chen, Zhang, and Feng [CZF11] uses the decisional n-Bilinear Diffie–Hellman Exponent (BDHE) assumption from [BBG05b]. These and other assumptions can be found in Appendix A.

2.6.3 Attack Model

The power and the goal of an attacker are modeled by an *attack model*. Well known attack models include the chosen plaintext attack (CPA) and the chosen ciphertext attack (CCA). In the CPA model, an attacker is assumed to be able to freely choose *plaintexts* and obtain the corresponding encrypted ciphertexts. In the CCA model, an attacker is assumed to be able to freely choose *ciphertexts* and obtain the corresponding decrypted plaintexts.

Chapter 3

Attribute-Based Credentials

People have used identity cards for identification and authentication for decades. Initially, such identity cards were handwritten or printed documents containing hallmarks or watermarks to guarantee their authenticity. More recently, identity cards are being equipped with a chip to provide a digital authentication method. Just like the analogue identification method, the digital equivalent must also contain a proof of authenticity.

A typical electronic identity (eID) card uses a Public Key Infrastructure (PKI) to guarantee the authenticity of the card. In a PKI setup users are given a smart card with a personalized public-private key pair. The card issuer digitally signs the public-key to guarantee its authenticity. The public-key together with the signature is called a *certificate*. For a user to prove his identity, he simply shows his certificate and proves that he knows the corresponding private key.

By using this classical PKI approach, we introduce some privacy related issues which were not present when we used the analogue identification method. This is mainly because digital data can easily be recorded and stored in large databases. Using the PKI card we will always identify ourselves with our personal certificate, which will often contain our name or other personal information. However, even if the certificate only contains a pseudonym we still lose some of our anonymity. This is due to the fact that the unique certificate can be stored in a database every time we show it. Because each time we identify ourselves with the same certificate, this reveals information about the usage pattern. For example, if we use the identity card to prove that we are old enough to buy liquor, then the liquor store can recognize the identity card and use it to see if you are a returning customer or have never been there before. We say that the user has become *linkable*.

3.1 Attribute-Based Credentials

With an Attribute-Based Credential (ABC) scheme, authentication takes place using attributes instead of identities. Using these attributes, a user can prove properties of his identity. For example, a user might prove that he is a student from the Netherlands or Belgium. This statement can be proven if the user

possesses the attributes for 'student' and 'Dutchman' or 'Belgian'.

Attributes can be seen as a key-value pair where the attribute key describes the corresponding value. It is good to realize that a lot of properties can be expressed as an attribute in this way. For example, your name, Citizen Service Number, hometown, loyalty card for a certain company, railroad ticket, or subscription to a service; all can be expressed as attributes. The attributes are grouped together in a *credential*. A user can possess multiple distinct credentials.

Statements can prove something about the attribute value, e.g., the value of 'loyalty status' is 'gold' and the value of 'student' equals 'true', or the possession of a credential, e.g., the user has the credential issued by the company 'Acme'. Moreover, statements can be combined using logical operators like AND, OR, EQ, or NOT. Using these operators, complex statements about the user can be formulated, e.g., 'hometown' EQ 'birthplace' AND ('age:43' OR 'age:65').

ABCs can be used as an access control mechanism using those statements. A verifier can create a statement that the user must meet in order to be granted access to some resource. The user will be authorized after he has successfully authenticated using the required statement. However, we do not want that multiple users can *collude* and combine their attributes so that their combined attributes satisfy the requirements that they would not have met individually.

In order to prove the authenticity of an attribute in a statement, digital signatures are used. A *digital signature* can be seen as the digital equivalent of a handwritten signature. However, there are two major differences between a digital and an analogue signature. Firstly, a digital signature is easily copied from one location to another. Secondly, signatures on distinct messages, signed by the same person, look differently.

Using such a digital signature that signs a credential, users can authenticate to a verifier. The verifier can check, by examining the signature, whether the attributes that the prover reveals are correct or fraudulent. Apart from the user and the verifier, we can distinguish another party, the *issuer* or *attribute authority* (AA). The issuer can sign user's credentials whose authenticity can be verified by a verifier.

3.2 The IRMA Project

The IRMA project started off by developing two practical implementations of two different ABC systems. Their first implementation uses Microsoft's U-Prove [MV12], whereas the second uses IBM's idemix [VA13].[1] Both implementations focus on practical usability by providing efficient smart card implementations of the schemes. Although the latter implementation is slower, it is preferred by the IRMA project because it is more privacy friendly. As a consequence, the project now focuses more on building an ecosystem around their idemix implementation. In 2013, a pilot using the idemix smart card implementation was started to obtain the first results of a larger scale hands-on experience.

IBM's Identity Mixer (idemix) is based on the Camenisch–Lysyanskaya signature scheme [CL01; CL03]. An important feature of this system is that it

[1] The source code of both implementations can be found on `https://github.com/credentials/`.

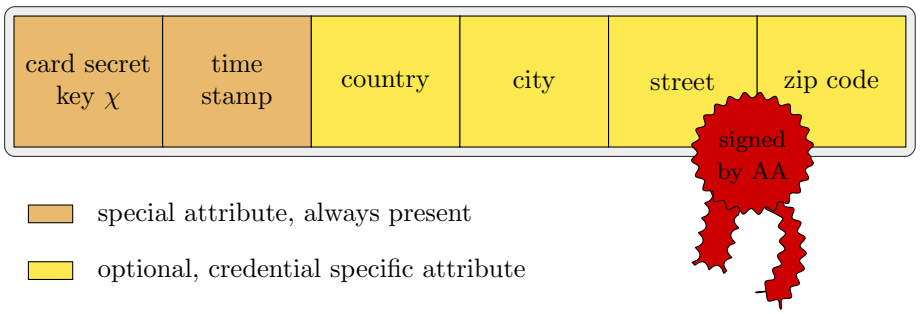

Figure 3.1: Graphical representation of the "address" credential.

provides *multi-show unlinkability*. This means that when you reveal your non-identifying attribute more than once to the same verifier, the verifier would not be able to tell if the attribute came from the same user or from a different user. This implies that, for example, a liquor store will not be able to learn from the ABC system whether you also bought a bottle of liquor the other day.

The IRMA project implements not only a subset of the idemix technology, but has also extended the technology with other technicalities to create a practical implementation of an ABC scheme. An idemix credential can be issued by any party, no central registration of the issuer is imposed. However, the issuer's public key of a signature needs to be published somewhere, in order to let a verifier check the credential. Due to the decentralized setup, the IRMA card is a perfect candidate for an eID card that is issued by the government where, besides the government, commercial companies could also issue new credentials to the card.

For efficiency reasons, an IRMA credential consists of no more than six attributes of which the last four are free to choose. The first attribute is used to store a card specific master secret key that will prevent collusion attacks, and the second one stores a validity time stamp. An IRMA credential "address" may contain, not only the two required attributes, but additionally contain the "country", "city", "street", and "zip code" attributes. Figure 3.1 shows a graphical representation of this credential.

Despite the great progress made in creating an efficient smart card implementation of idemix [VA13], some use cases, e.g., public transportation or money transactions, require an even faster implementation than the current one. The current implementation requires just over 1 second to reveal a single attribute and drops just below a second when all four attributes and the time stamp are disclosed [VA13]. A transaction time of less than 350 milliseconds is required in most public transportation scenarios [HJV10]. The fact that current cryptographic smart cards do not allow direct access to most of the underling cryptographic primitives presents a challenge to do fast verification or issuance of attributes.

3.3 Identity Mixer

The key concept behind the idemix system that is implemented by the IRMA technology, is the *Camenisch–Lysyanskaya signature* scheme [CL01; CL03]. We will discuss the most important parts of the idemix system, but for an exact description of the setup, the reader is referred to the official idemix specification [IBM13].[2]

The idemix system consists of various algorithms and protocols, we describe only the most relevant.

Authority Setup$(1^k) \to (\mathbf{PK}, \mathbf{SK})$ This algorithm initializes the system by creating a private-public key pair for a single attribute authority (AA) based on a security parameter k.

Issue Credential$(\mathbf{PK}, \mathbf{SK}, \chi, (t_1, \ldots, t_l)) \to (A, e, v)$ New credentials with attributes $(t_1, \ldots, t_l)$ can be issued by an AA with private key (SK) to a user with a master secret key χ, by creating a signature on the attributes and the user's master key χ via the issuance protocol. Note that, although the user never reveals his master secret key χ—not even to the AA—credentials are bound to the key χ. This binding prevents collusion attacks.

Disclose Attributes$(\mathbf{PK}, \chi, (t_1, \ldots, t_l), (A, e, v))$ Attributes $(t_1, \ldots, t_l)$ of a credential can be selectively disclosed by a user, by showing only a subset of the attributes $(t_1, \ldots, t_l)$ to the verifier. In order to provide multi-show unlinkability, the user can randomize his signature (A, e, v) before it is shown. The verifier checks the signature on the credential corresponding to the disclosed attributes.

A Camenisch–Lysyanskaya signature [CL01] is a special form of an RSA signature [RSA78] that can be defined over $l + 1$ different messages and can be randomized, yet still be proven valid for the messages it signs.

Scheme (Identity Mixer [IBM13])**.** A simplification[3] of the [IBM13] scheme is provided.

Authority Setup Select two safe primes $p = 2p' + 1$ and $q = 2q' + 1$ for the RSA modulus $n = pq$. Select $l + 3$ random elements out of the set of quadratic residues, $Z, S, R_0, \ldots, R_l \in QR_n$. The AA's public key is $\mathrm{PK} = (n, Z, S, R_0, \ldots, R_l)$, its private key is $\mathrm{SK} = (p, q)$.

Issue Credential A credential, consisting of several attributes $t_1, \ldots, t_l$, is issued by AA to a user with a master secret key χ. The issuer starts the protocol by choosing a random nonce n_1 and sends this to the user. The user responds by choosing a random value ν' and calculating $U = S^{\nu'} R_0^{\chi} \mod n$. He sends the value U together with the signature of knowledge

$$\Sigma_1 = \mathrm{SPK}\{(\nu', \chi) : U \equiv S^{\nu'} R_0^{\chi} \pmod{n}\}(U, n_1),$$

[2] Due to space constrains, we omit the precise zero-knowledge proofs. We do not discuss the required interval checks, nor discuss how such a proof works while the order of the group is kept secret.

[3] The idemix cryptographic library contains more proofs than the ones that we discuss here. The other proofs are not relevant for this work.

proving that U was properly constructed, and a freshly picked nonce n_2 over to the AA. The AA validates the proof and chooses random v'' and random prime e if the proof verifies correctly. Using the selected random numbers it determines[4] $d = e^{-1} \mod p'q'$ and calculates

$$A = \left(\frac{Z}{US^{v''} \prod_{i=1}^{l} R_i^{t_i}} \right)^{d} \mod n.$$

It also generates a signature of knowledge to prove that this A is well-formed,

$$\Sigma_2 = \mathrm{SPK}\{(\delta) : A \equiv \left(\frac{Z}{US^{v''} \prod_{i=1}^{l} R_i^{t_i}} \right)^{\delta} \pmod n\}(A, n_2).$$

The AA sends the 3-tuple (A, e, v'') and the signature of knowledge back to the user.

The user stores the signature on the credential as the 3-tuple $(A, e, v = \nu' + v'')$. He can verify if the signature is correct by checking

$$Z \stackrel{?}{\equiv} A^e S^v R_0^{\chi} \prod_{i=1}^{l} R_i^{t_i} \pmod n.$$

The protocol is graphically described in Figure 3.2.

Disclose Attributes A user determines a set of attribute indices $\mathcal{D} \subseteq \{1, \ldots, l\}$ that he wants to reveal (the master secret key with index 0 should never be revealed). Let the set $\overline{\mathcal{D}}$ contain the indices of the attributes that the user does not want to reveal.

The issuer gives the user a random nonce n_1 to sign, to guarantee freshness of the proof. Next, the user chooses a random value r to randomize his Camenisch–Lysyanskaya signature (A, e, v) with. His new randomized signature is the 3-tuple $(A' = AS^r \mod n, e, \nu = v - er)$. In order to prove the correctness of the signature, a signature of knowledge proof is constructed. The user has to prove the equality

$$Z \prod_{i \in \mathcal{D}} R_i^{-t_i} \equiv A'^e S^{\nu} R_0^{\chi} \prod_{i \in \overline{\mathcal{D}}} R_i^{t_i} \pmod n,$$

the equality will hold only if the signature is correct.

Figure 3.3 shows the run of the verification protocol where the first two attributes, t_1 and t_2, are disclosed. All the other attributes remain hidden to the verifier.

A general idea on why the signature is unforgeable, is that the modular inverse of e can only be calculated if $|QR_n| = p'q'$ is known[5]. The complete security proof of the scheme relies on the Strong RSA assumption and the Decisional Diffie–Hellman (DDH) assumption modulo a safe prime product [CL01].

[4] Note that $e^{-1} \mod \phi(n)$ may also be used. However, calculating modulo $p'q'$ is more efficient and yields the same final result since $A \in QR_n$.

[5] An attacker that can determine $\phi(n) = (p-1)(q-1)$ can also break the scheme using the calculation from Footnote 4.

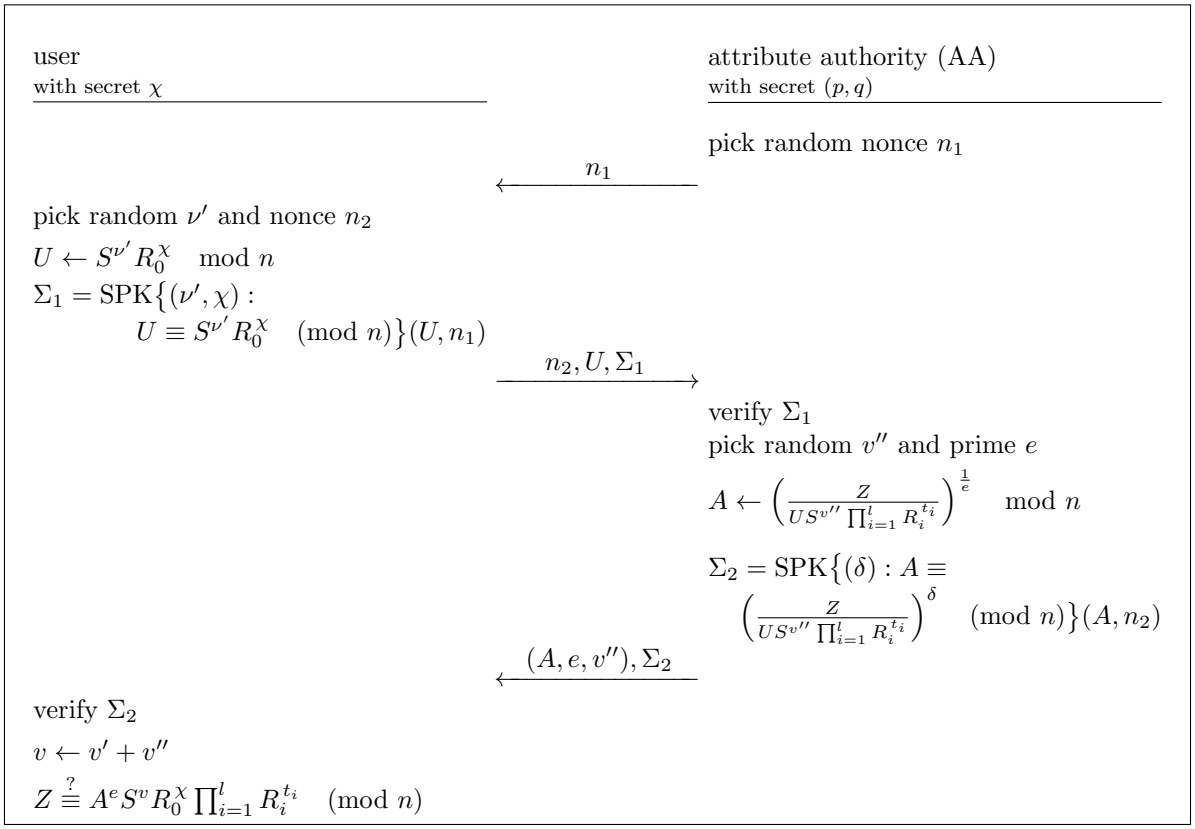

Figure 3.2: Issuance protocol for an idemix credential (simplified).

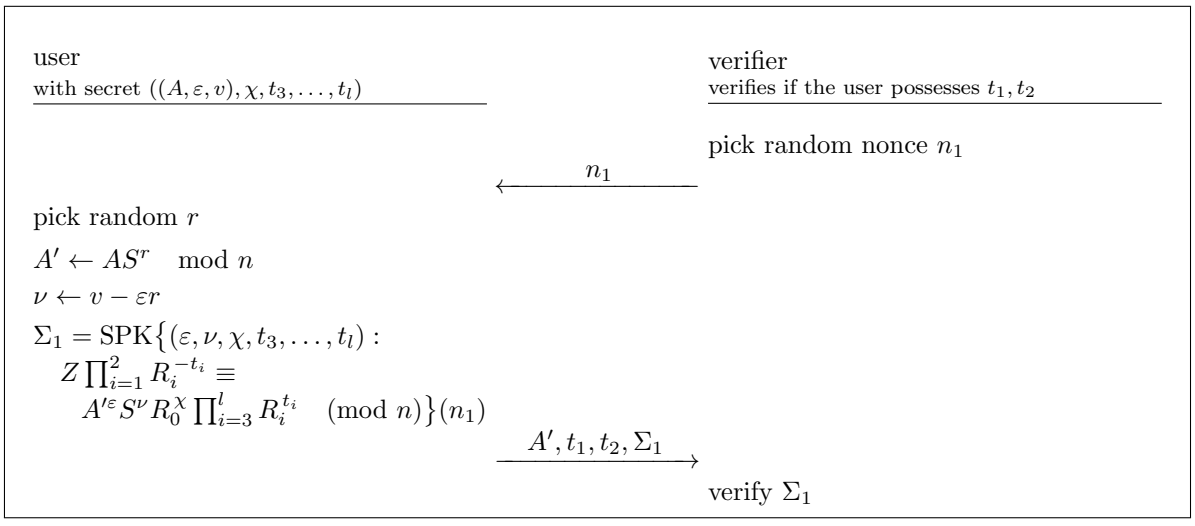

Figure 3.3: Verification protocol of attributes t_1, t_2 of an idemix credential (simplified).

Collusion resistance Assume the following access structure that a user needs to satisfy in order to get a discount for a museum ticket bought online: 'nationality:Dutch' AND ('occupation:student' OR 'loyalty card:gold'), where 'nationality' and 'occupation' are attributes from the same credential, but 'loyalty card' is an attribute from a different credential. Alice and Bob, good friends of each other, like to go to the museum with a discount, however, neither of them has the right attributes in order to apply for the discount. Alice only has the attribute 'nationality:Dutch', whereas Bob only has the attributes 'occupation:student' and 'loyalty card:gold'. Alice and Bob, tech savvy as they are, try to combine their attributes in order to satisfy the access structure. They start off by trying to fool the system by combining their 'nationality:Dutch' and 'occupation:student' attributes to match the policy. However, because those attribute keys are from the same credential—something they could not achieve by combining the two credentials—they quickly decide that they could just focus on combining the attributes 'nationality:Dutch' and 'loyalty card:gold' as the challenge would be similar.

No matter what Alice and Bob try, they would not succeed in proving that the credential with master secret key χ_{Alice}, that also contains attribute 'nationality:Dutch', is identical to the credential with master secret key χ_{Bob}, that contains attribute 'loyalty card:gold'. Thus, the museum web shop just needs to require in its access structure that the used master secret keys of different credentials are the same. Note that this can easily be implemented with the EQ-composition, without the need of a verifier to learn the master secret key.

Chapter 4

Attribute-Based Encryption

Encryption-based access control has several advantages over classical access control. In a classical setup, as depicted in Figure 4.1, data is stored unencrypted on the server and the user needs to authenticate each time she wants to retrieve data from the server. The server is required to authorize the user's request before it sends the plaintext data to the user. Letting the server authorize the user's requests allows for flexible and fine-grained access control. However, the server needs to be trusted and well-protected. In case the server gets compromised, all the plaintext data becomes available to the attacker. Besides the security issue of storing plaintext data on a server, another practical problem exists. The authentication server needs to be online to handle each request, making the server vulnerable to Denial of Service attacks.

Instead of storing the data in plaintext format on the server, one could encrypt the data and store this on the server. This has the advantage that the server is not burdened with the authorization and authentication of users. Moreover, the data can be stored on many—even untrusted—servers, as it is encrypted anyway. Not all encryption types can be used to obtain flexible and fine-grained access control. If one could use attributes—like the ones used in ABC—and define access policies over them, fine-grained access control could be obtained. Attribute-Based Encryption (ABE) is a type of public-key encryption where decryption keys are associated with attributes. By using ABE, flexible and fine-grained access control can be obtained.

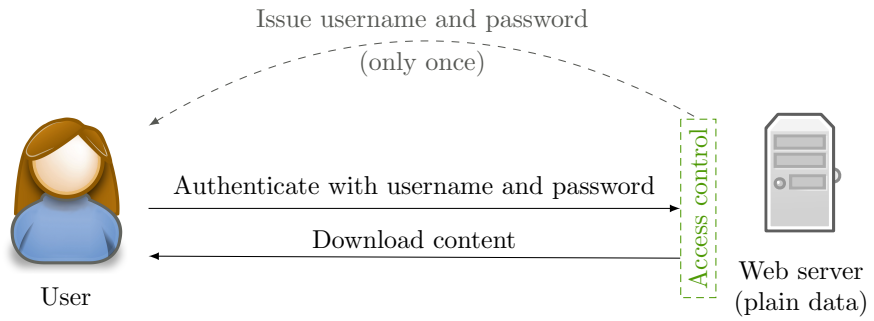

Figure 4.1: Classical access control.

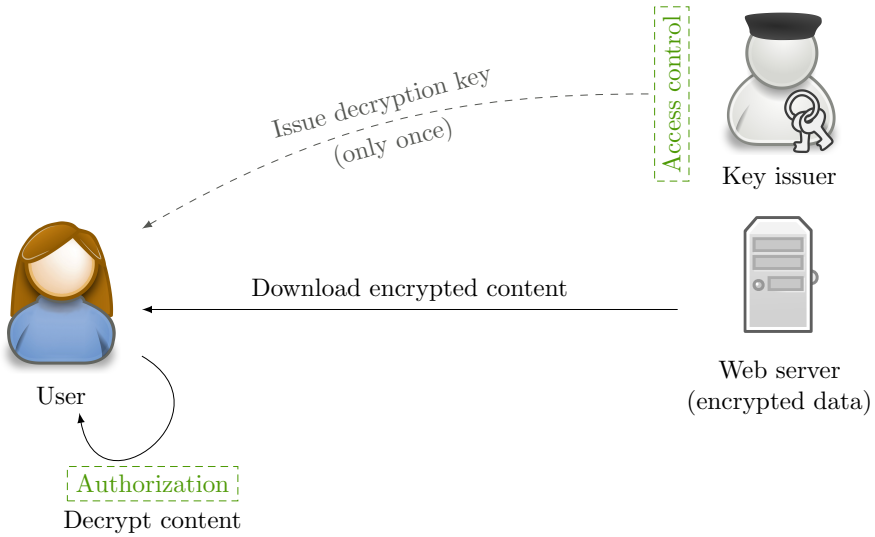

Figure 4.2: Encryption-based access control.

In Figure 4.2 the encryption-based access control method is schematically shown. Just like in the classical method, the user first needs to register before she can request access to the data. Instead of choosing a username and password, the user is issued a decryption key associated with several attributes. The main difference with the classical access control is that all files are stored encrypted on the server and that *anyone* can download the data. The ciphertexts are associated with an access policy, determining who can decrypt which ciphertext. Thus, the authorization will take place when the user tries to decrypt the ciphertext. Typically, this decryption takes place on the client side.

4.1 Why ABE?

Attribute-Based Encryption is a variation on Identity-Based Encryption (IBE). Identity-Based Encryption [Sha85] is a special type of asymmetric cryptography where no public key certificate of the recipient is needed when encrypting a message. For example, Alice can encrypt a message to Bob using the publicly available system parameters and Bob's e-mail address as his 'public key'. Bob can request the private key for his identity—the e-mail address in this case—from a Trusted Third Party (TTP). The TTP verifies that Bob is the owner of the e-mail address and, if verification succeeds, provides Bob with a private key derived from his e-mail address.

The difference between ABE and IBE is that in ABE an *access policy* can be defined on who may decrypt the ciphertext. This cryptographically enforced access policy lends itself to the use of *attributes* instead of *identities*. Although IBE could be used with attributes instead of identities, the use of IBE becomes impractical when one tries to mimic the use of complex access policies. For simple policies, like "male AND (over-18 OR student)", a message could be encrypted using IBE. For example, we could emulate the policy in IBE by creating two

ciphertexts, the message encrypted to the attributes/identities "over-18" and "student", and encrypting those resulting ciphertexts again with the attribute "male" to create two final ciphertexts. Now, when both final ciphertexts are published, a male student can obtain the message by first decrypting one of the ciphertexts with his key for the attribute "male" and subsequently decrypt the obtained result with his key for the attribute "student". However, Alice, a female student, could decrypt the ciphertext too, as long as she colludes with another male. In order to do so, she first asks Bob to decrypt the ciphertext—which he can with his "male" attribute key—and she decrypts the result from Bob with her own "student" attribute key.

ABE, in contrast to IBE, *does* provide protection against this collusion attack. We call a scheme *collusion resistant* when two users who cannot decrypt a message individually—with their own private key—cannot combine their keys in order to jointly decrypt the ciphertext.

Definition 17 (Collusion resistance [SW05])**.** No group of users should be able to combine their keys in such a way that they can decrypt a ciphertext that none of them alone could.

In case no policies are needed, i.e., messages will only be encrypted to one single attribute, the advantage of ABE over IBE disappears. The use of IBE still has a slight advantage over public-key cryptography as the public key for each attribute does not have to be determined before one could encrypt to the attribute. If we do not consider this to be a problem, we could even consider to use symmetric encryption. Symmetric cryptography has the advantage of being much faster than asymmetric cryptography. However, using symmetric cryptography has an inherent drawback compared to the asymmetric variant: because the encryption key is identical to the decryption key, the encryptor automatically obtains the privilege to decrypt other data with the same attribute as well.

Since we do not want to make any concessions regarding usability of our encryption scheme, we will further examine ABE.

4.2 History of ABE

The concept of Attribute-Based Encryption (ABE) evolved out of the notion of IBE. Sahai and Waters [SW05] have created an encryption method which they called *Fuzzy Identity-Based Encryption*. The goal of the scheme is to allow a user to encrypt messages to an identity ω and enable the recipient of the ciphertext to decrypt the ciphertext if his identity ω' is close enough to the identity ω. They envisioned two goals for their scheme. One was to allow encryption to a biometric feature, e.g., the feature vector of an iris scan. The second application is what they called *attribute-based encryption*.

Their scheme is the first proposed ABE scheme. In the scheme, the ciphertext and the private keys are associated with attributes (or features of a biometric). The ciphertext can only be decrypted when enough attributes of the private key match with the attributes associated with the ciphertext. This makes the definition of an *access policy* possible. The use of a policy allows more fine-grained access control over the encrypted data. For example, if Alice wants to encrypt a message to all females over 18, she associates the ciphertext with

the attribute set {"female", "over-18"} and requires that the decryptor should possess both attributes—she sets the threshold value $k = 2$. Newer schemes allow more complex policies where propositional logic can be used, i.e., policies like "(female AND over-18) OR (male AND driver's license)" are possible.

Goyal et al. [GPS+06] propose a new scheme which allows the incorporation of policies into the private key. They describe two different types of ABE: Key-Policy ABE (KP-ABE)—for which they also provide a working scheme—and Ciphertext-Policy ABE (CP-ABE). In a KP-ABE scheme, the ciphertext is associated with a set of attributes and on the user's private key a policy is defined. The policy determines which messages can be decrypted with the private key. In a CP-ABE scheme, the situation is reversed. The policy is defined over the ciphertext and the private key is associated with a set of attributes. The first CP-ABE scheme was created by Bethencourt, Sahai, and Waters [BSW07]. Both the work of Goyal et al. and Bethencourt, Sahai, and Waters allow the creation of policies by combining attributes with the use of AND, OR and—the generalization—k-OUT-OF-n operators. We will elaborate on the differences between KP-ABE and CP-ABE in the next section.

4.3 ABE Use Cases

The main advantage of ABE is that the encryptor does not need to know the precise identity of the person to whom he encrypts the data. Moreover, ABE enables one to distribute data via a cloud network in order to assert high availability, without having to worry about access control or a compromised or corrupt server. Attribute-Based Encryption has practical use cases in several fields, e.g., military or other hierarchical structures, and even in the field of personal health records. The policy over a set of attributes defines who is able to decrypt what data. Thus the choice between KP-ABE and CP-ABE boils down to the choice who must define the access policy.

For instance, in a military scenario, it might be desirable for the military leaders, i.e., the TTP, to determine who can access which files. A KP-ABE scheme suites this scenario best, allowing the military leaders to create an access policy for each user. Soldiers can then encrypt data by associating it with the document's metadata, e.g., the creation time or location and the squadron or rank of the author. If the encrypted data are associated with the correct attributes, only the soldiers granted access by the military leaders will be able to decrypt the ciphertext. The encryptor does not need to know who is granted access to which files. *Broadcast encryption* is also an example where KP-ABE can be used [GPS+06]. Premium channels can be broadcast encrypted and clients are given only the decryption keys for the content they paid for. The broadcasts are associated with attributes and clients are given a key in which a client specific policy resides. A policy in the form of "package:sport OR (package:movie AND genre:action)" will allow a client to watch action movies and sport broadcasts.

In contrast, a CP-ABE scheme is useful in a scenario where the encryptor is also the owner of his data. In this case, *the user*—and not the TTP—has to be able to determine who may decrypt the resulting ciphertext. In a centralized personal health record, a patient should be able to determine whom he gives access to his data. By using a CP-ABE scheme, the patient can encrypt his data in such a way that he is the one who determines which doctors have access

	KP-ABE	CP-ABE
Key	describes a policy	describes a set of attributes
Ciphertext	associated with a set of attributes	associated with a policy
TTP	determines the policy	determines the attributes
Encryptor	determines the attributes	determines the policy

Table 4.1: Differences between KP-ABE and CP-ABE.

to which parts of his personal health record.

The differences between KP-ABE and CP-ABE are summarized in Table 4.1.

4.4 Intuition Behind General ABE Schemes

In order to get a feeling of how the ABE schemes work, we will discuss the typical ABE scheme on an abstract level. In each scheme there are—at least—three parties: the encryptor, the decryptor and a TTP. We will call the TTP in this scenario the *key generation authority* (KGA). The KGA will initiate the system and issue the private keys to the decryptors. We can distinguish at least four different algorithms in every scheme: Setup, Encrypt, Key Generation, and Decrypt.

Setup This algorithm is run before all other algorithms and determines the public parameters (PK) and a master key (MK) for the KGA. The PK determines the set of all possible attributes and all user keys will be derived from the MK.

Encrypt Anyone who has the PK can encrypt their data and associate the resulting ciphertext with attributes (in the case of KP-ABE) or an access policy (in the case of CP-ABE) that will determine which users can decrypt the ciphertext.

The ciphertext consists of multiple parts. One of these parts is a randomly chosen secret number operating on the plaintext. The other parts are needed to reconstruct this secret number. Using a secret sharing scheme that splits the secret number into various parts, the access structure is enforced by using these in parts of the ciphertext.

Key Generation The KGA can create new decryption keys for users using its MK. A user's private key (SK) is derived from the MK by randomizing

the MK in such a way that the user cannot convert the SK back to the MK.

To prevent user collusion, each SK is randomized by a unique, user-specific number, or, the key is bound to a fixed global identifier (GID) of the user.

Decrypt The decryptor can check the ciphertext to see if he is able to decrypt the ciphertext. If the decryptor's attributes satisfy the access structure, then he is able to reconstruct the secret number used to encrypt the plaintext. He can do so by operating on several other parts of the ciphertext together with his SK. The recovered secret number can be used to invert the encryption and obtain the plaintext.

4.4.1 Enforcing Access Policies

The key aspect of the Encrypt and Decrypt algorithms is the cryptographic enforcement of the access structure. Different approaches exist to enforce the access policy, but all approaches use a form of secret sharing. Several early ABE schemes [GPS+06; BSW07; ITH+09] use *Shamir's Secret Sharing Scheme* (SSSS) to convert an access structure into an access tree. Ibraimi et al. [ITH+09] also use the *Unanimous Consent Control by Modular Addition Scheme* instead of SSSS to improve the computational efficiency of the access tree. Another approach of incorporating the policy, is to use *Linear Secret Sharing Scheme* (LSSS) matrices (or equivalently a monotone span program) [LW11]. We will describe the use of SSSS and LSSS to cryptographically enforce the access structure in the next two algorithms. Later, we will provide an example for both algorithms in Examples 4.1 and 6.1.

An access structure $\mathcal{A}$ can be represented by an *access tree* $\mathcal{T}$ through SSSS (Figure 4.3 and Example 4.1 will clarify how to do so). The general idea is to split the secret value s over all the leaf nodes of the tree, where the leaf nodes represent the attributes in the access policy. The structure of the tree determines which combinations of leaf nodes are required to be able to reconstruct the secret value. When a user wants to decrypt the ciphertext, he needs to combine the leaf nodes to reconstruct the secret. However, the ABE scheme forces him to use only the leaf nodes that represent the attributes that he possesses.

Example 4.1 will explain how the access structure can be converted into an access tree.

Algorithm 2 (Access Tree)**.** Convert each AND and OR operator in the access structure to a k-OUT-OF-n operator. Note that the k-OUT-OF-n operator is a generalization of both the AND and the OR operator. An AND operator can be converted to a k-OUT-OF-n operator by setting $k = n$ (so, n-OUT-OF-n) and an OR operator can be converted to a k-OUT-OF-n operator by setting $k = 1$ (so, 1-OUT-OF-n).

Write the access structure as a tree $\mathcal{T}$, describe each non-leaf node by a k-OUT-OF-n operator, and let each leaf node represent an attribute. The value k_x for each non-leaf node x determines the threshold k of the k-OUT-OF-n operator. The value n_x represents the number of children of the node x. Assign each node x in the tree a uniquely identifying number index(x). Let parent(x) return the parent node for each node x, except for the root node. Let the function attr(x) return the associated attribute for each leaf node x in the access tree.

Associate each node x in the tree with a polynomial q_x of degree $k_x - 1$ for each non-leaf node and a constant polynomial for each leaf node. Pick a secret value $s \in_R \mathbb{Z}_p$, let the root node polynomial satisfy $q_{\text{root}}(0) = s$ and pick all the other coefficients randomly. For all other nodes set $q_x(0) = q_{\text{parent}(x)}(\text{index}(x))$ and assign all other coefficients a random value.

The process of converting an access structure into an access tree is further illustrated with an example in Section 4.5.

Instead of creating a tree, we could also convert an access structure into an *access matrix*. Here, the idea is to create a matrix where several rows are needed to create a linear combination that results in a vector of the form $\begin{pmatrix}1 & 0 & \cdots & 0\end{pmatrix}$. Each row in the access matrix represents an attribute from the access policy. When a user wants to decrypt the ciphertext, he needs to select several rows such that $\begin{pmatrix}1 & 0 & \cdots & 0\end{pmatrix}$ is in the subset spanned by the rows, i.e., there exists a linear combination of the rows such that $\begin{pmatrix}1 & 0 & \cdots & 0\end{pmatrix}$ is the result. However, the ABE scheme forces him to use only the rows that represent attributes that he possesses.

To convert an access structure into an access matrix, one could use the algorithm by Lewko and Waters [LW10] or by Liu and Cao [LC10]. Liu and Cao claim that their algorithm generates the smallest matrices possible. For policies with threshold gates, i.e., the k-OUT-OF-n operator, they indeed generate a smaller LSSS matrix than the Lewko and Waters [LW10] construction. However, since the algorithm by Lewko and Waters is simpler and results in small matrices in most cases anyway, we state here only their algorithm.

Algorithm 3 (Access Matrix)**.** Write the access structure as a tree, describe each non-leaf node by an AND or OR operator, and let each leaf node represent an attribute. Let c represent a counter value for the AND nodes and initialize it to 1. Each node is labeled with a vector determined by its parent node; the root node is assigned the vector $\begin{pmatrix}1\end{pmatrix}$. The children of a node described by an OR operator inherit the same vector as their parent. The vector v of a node described by an AND operator is first padded (if necessary) with zeros until its length equals c. The two children the node are labeled with the padded vector v' concatenated with the number 1 and with the zero-vector of the size of v' concatenated with the number -1. Increment the counter value by 1. Note that the two vectors sum to v' concatenated with 0. When every node is labeled with a vector, create equally sized vectors by padding the shorter ones with zeros to the end. Now, stack the vectors of every leaf node on top of each other to create the LSSS matrix.

An example of such a conversion of an access structure into an access matrix can be found in Section 6.1.

4.5 An ABE Scheme Explained

Before we dive into the construction of an ABE scheme, we first take a look at an example on how to convert an access structure into an access tree.

Conversion to an access tree The next example converts the policy "(data analyst AND (mathematician OR senior manager)) OR executive board" into an access tree using Algorithm 2.

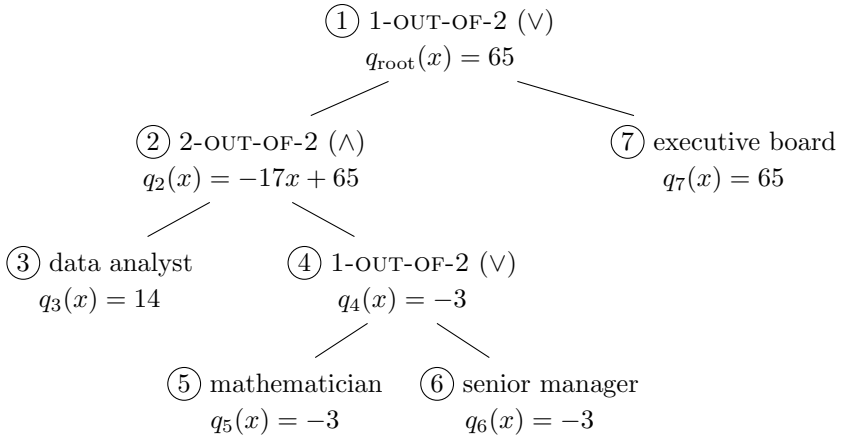

Figure 4.3: A graphical representation of the accesses tree for the policy "(data analyst AND (mathematician OR senior manager)) OR executive board".

Example 4.1 (Access Tree)**.** We start by converting the AND and OR operators from the access structure to a k-OUT-OF-n operator. The obtained access structure looks a bit cluttered, but it boils down to the same access structure we had before: "1-OUT-OF-2{2-OUT-OF-2{data analyst, 1-OUT-OF-2{mathematician, senior manager}}, executive board}".

We now begin with the creation of the tree. The second OR in the original access structure becomes the root node with "data analyst AND (mathematician OR senior manager)" and "executive board" as its children. The first child node can be further converted into a subtree, where the second child node becomes a leaf node. We continue with this construction until the whole tree has been built. Next, we assign to each node a unique number. A graphical representation of the complete access tree is shown in Figure 4.3. For the sake of readability, we will sometimes refer to node x as the number assigned to node x, i.e., $\text{index}(x)$.

We now start with the selection of the polynomials q_x for each node x. We pick a 'random' secret $s = 65$ and associate the root node with a polynomial of degree $k_1 - 1 = 0$, i.e., a constant polynomial, so $q_{\text{root}}(x) = 65$. Leaf node 7 just inherits the same value because $q_7(0) = q_{\text{parent}(x)}(\text{index}(x)) = q_{\text{root}}(7) = 65$. Node 2 will be associated with a polynomial of degree $k_2 - 1 = 1$, so we determine the polynomial by 'randomly' picking the coefficient -17 and setting $q_2(0) = q_{\text{parent}(x)}(\text{index}(x)) = q_{\text{root}}(2) = 65$. We obtain $q_2(x) = -17x + 65$. The polynomial for leaf node 3 is obtained by calculating $q_3(0) = q_2(3) = 14$.

The rest of the selection of the polynomials is done analogously. The final result can be found in Figure 4.3.

The polynomial of each non-leaf node can be reconstructed when enough values of the child nodes are available. The reconstruction is identical to the Pooling of SSSS: one uses Lagrange interpolation in order to determine the polynomial. For example, the value $q_2(0)$ can be determined if the values of the node's children $q_3(0)$ and $q_4(0)$ are known. In this case we have $(3, q_3(0) = 14)$ and $(4, q_4(0) = -3)$ as shares. The Lagrange coefficients are $\Delta_3(0) = \frac{0-4}{3-4} = 4$ and $\Delta_4(0) = \frac{0-3}{4-3} = -3$. Using these Lagrange coefficients, the value $q_2(0) = 65$

can be obtained, $q_2(0) = 14\Delta_3(0) + -3\Delta_4(0) = 14 \cdot 4 + -3 \cdot -3 = 65$.

A simple ABE scheme In order to provide some intuition on how a general ABE scheme works, we describe a simplification[1] of the Bethencourt, Sahai, and Waters [BSW07] CP-ABE scheme.

Scheme (Bethencourt, Sahai, and Waters [BSW07])**.** The scheme consists of the usual four algorithms. Those algorithms are described one by one.

Setup Select a bilinear group $\mathbb{G}_1$, with mapping $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$, of prime order p and generator g. Next, select $\alpha, \beta \in_R \mathbb{Z}_p$. Determine a hash function $\mathcal{H}$ which maps arbitrary attributes to a number in $\mathbb{G}_1$. The public parameters are $\text{PK} = (\mathbb{G}_1, e, p, g, h = e(g,g)^\alpha, u = g^\beta, \mathcal{H})$. The master key is $\text{MK} = (g^\alpha, \beta)$.

Encrypt The algorithm encrypts a message $M \in \mathbb{G}_T$ with the PK under the access structure $\mathcal{A}$. Convert the access structure $\mathcal{A}$ to an access tree $\mathcal{T}$ according to Algorithm 2. Recall that the root node polynomial satisfies $q_{\text{root}}(0) = s$ for an $s \in_R \mathbb{Z}_p$.

Let Y be the set of leaf nodes in $\mathcal{T}$. The ciphertext is published as

$$\text{CT} = (\mathcal{T}, \tilde{C} = Mh^s, C = u^s, \\ \forall y \in Y : \; C_y = g^{q_y(0)}, C'_y = \mathcal{H}(\text{attr}(y))^{q_y(0)}).$$

Key Generation The algorithm, executed by the KGA, derives a private key from the MK for a set of attributes $\mathcal{S}_U$ of a user U. Select $r \in_R \mathbb{Z}_p$ and $\forall i \in \mathcal{S}_U : \; r_i \in_R \mathbb{Z}_p$. The private key for user U is

$$\text{SK} = (D = g^{\frac{\alpha + r}{\beta}}, \forall i \in \mathcal{S} : \; D_i = g^r \mathcal{H}(i)^{r_i}, D'_i = g^{r_i}).$$

Decrypt Determine if the attributes of SK satisfy the access structure $\mathcal{A}$, corresponding to the given access tree $\mathcal{T}$. If the attributes do not satisfy the access policy, stop and output $\perp$, otherwise continue. Let $\mathcal{S} \subseteq \mathcal{S}_U$ be a smallest set of attributes that satisfy the policy (more than one of such a set may exist). Determine for each $i \in \mathcal{S}$ the node x such that $\text{attr}(x) = i$ and calculate

$$\begin{aligned} F_x &= \frac{e(D_i, C_x)}{e(D'_i, C'_x)} \\ &= \frac{e(g^r \mathcal{H}(i)^{r_i}, g^{q_x(0)})}{e(g^{r_i}, \mathcal{H}(\text{attr}(x))^{q_x(0)}))} \\ &= e(g^r, g^{q_x(0)}) \frac{e(\mathcal{H}(i)^{r_i}, g^{q_x(0)})}{e(g^{r_i}, \mathcal{H}(i)^{q_x(0)}))} \\ &= e(g,g)^{r q_x(0)}. \end{aligned}$$

Before we continue with the algorithm, we introduce some new notations. Let $\text{child}(x)$ be the set of children of node x in the access tree. Define the Lagrange coefficient for $i \in \mathbb{Z}_p$ and some set S, containing elements

[1] We leave out their Delegate algorithm, as it is not relevant for this work.

of $\mathbb{Z}_p$, as $\Delta_{i,S}(x) = \prod_{j \in S; i \neq j} \frac{x-j}{i-j}$. Recall that we can evaluate a k-degree polynomial using Lagrange interpolation,

$$q_x(0) = \sum_{i \in S} q_x(i) \cdot \Delta_{i,S}(0),$$

if the set S contains at least $k+1$ distinct values. Using this interpolation, we can evaluate the polynomial of node p in the tree, as long as we know the value of F_z for all the node's children z,

$$\begin{aligned} F_p &= \prod_{z \in \text{child}(p)} F_z^{\Delta_{\text{index}(z),\{\text{index}(y) | y \in \text{child}(p)\}}(0)} \\ &= \prod_{z \in \text{child}(p)} \left(e(g,g)^{rq_z(0)}\right)^{\Delta_{\text{index}(z),\{\text{index}(y) | y \in \text{child}(p)\}}(0)} \\ &= \prod_{z \in \text{child}(p)} (e(g,g)^{rq_p(\text{index}(z))\Delta_{\text{index}(z),\{\text{index}(y) | y \in \text{child}(p)\}}(0)} \\ &= e(g,g)^{rq_p(0)}. \end{aligned}$$

Continue this calculation all the way up to the root of the tree to obtain $F_{\text{root}} = e(g,g)^{rq_{\text{root}}(0)} = e(g,g)^{rs}$. Finally, retrieve the plaintext by calculating

$$\tilde{C} \frac{F_{\text{root}}}{e(C,D)} = \tilde{C} \frac{e(g,g)^{rs}}{e\left(u^s, g^{\frac{\alpha+r}{\beta}}\right)} = \tilde{C} e(g,g)^{rs-(\alpha+r)s} = M.$$

The [BSW07] scheme prevents user collusion by randomizing the user's SK with a factor r and binds each attribute that the user possesses to this same r. When two different users try to combine their attributes, i.e., combine their different tuple (D_i, D_i') in a policy, decryption will fail. The intermediate result F_x for each attribute will be bound to the randomization factor r of the user. Because of this user specific randomization factor, the Lagrange interpolation fails for the combination of two different F_x of different users. The parent F_p of the F_{x_1} of a user with randomization factor r and the F_{x_2} of a user with randomization factor $\tilde{r}$ will contain an exponent of the form $rq_{x_1}\Delta_{x_1}(0) + \tilde{r}q_{x_2}\Delta_{x_2}(0)$ which does not simplify, whereas if $r = \tilde{r}$ this simplifies to $r(q_{x_1}\Delta_{x_1}(0) + q_{x_2}\Delta_{x_2}(0)) = rq_p(0)$.

Chapter 5

Choosing a Suitable ABE Scheme for the IRMA Ecosystem

This chapter discusses different types of ABE and enables us to make a proper decision on what kind of ABE to choose.

5.1 Types of ABE

ABE schemes can be classified based on several properties of a scheme. Policy placement is a well-known classification, as explained in the previous chapter. The policy can reside in the user's SK (KP-ABE) or in the ciphertext (CP-ABE). Independent of this partition, the schemes could also be grouped by the availability of the number of attributes. In a *small universe construction* the PK grows linearly in size with the amount of possible attributes and the attributes need to be explicitly defined in the Setup algorithm. However, this does not necessarily mean that it is always the case that no new attributes can be added after the Setup algorithm has been run. In a *large universe construction* one does not need to explicitly define the set of attributes during Setup. Any attribute that is uniquely mapped to a specific group, that is fixed in the Setup algorithm, can be used. The large universe construction is generally computationally more expensive. However, the construction may be more efficient in storage space, as the storage space of the PK in the large universe construction is independent of the number of attributes in the universe, in contrast to the storage space required by the PK in the small universe construction. The [BSW07] scheme described in Section 4.5 is an example of a large universe construction.

In addition to the other two properties, we can describe the schemes by the allowed number of authorities in the system, since in some situations it is deemed impractical to have a single KGA issuing all the SKs to each user. Moreover, the KGA will be able to decrypt any message encrypted to whatever policy. This is called the *key escrow problem*, a problem that also exists in IBE. If one likes to have multiple KGAs and still wants to be able to encrypt documents with an access policy spanning those multiple authorities, a

multi-authority ABE scheme can be used. The first published *multi-authority* scheme [Cha07]—based on KP-ABE—uses a central authority (CA), that is still able to decrypt *any* encrypted message. Each KGA is assigned a subset of the attribute universe by the CA and can issue a SK for any user. In order to maintain the collusion resistance, all KGAs should agree on a unique global identifier (GID) for each user, instead of randomizing the SK. Chase and Chow [CC09] propose a different approach to maintain collusion resistance where they used a pseudo random number generator whose output always sums to zero for any particular input. Since the setting where a CA is able to decipher any message might be undesirable, a second type of multi-authority schemes are proposed. In a *decentralized* multi-authority scheme no CA is needed, and a KGA is only able to decrypt a ciphertext if the attributes created by that authority satisfy the policy associated with the ciphertext.

Additionally, we describe the schemes by their expressiveness of the possible access structures, as different schemes allow different access policies. For example, there exists a scheme that only allows the use of one AND operator [CN07], while several other schemes allow the nesting of many k-OUT-OF-n operators in a policy [GPS+06; BSW07; LW11]. Besides the number or types of operators that can be used in a policy, we distinguish two types of access policies: monotonic and non-monotonic. *Monotonic* access policies do not allow negations of attributes, *non-monotonic* ones do allow such negations. Excluding groups is trivial in a non-monotonic policy, "manager AND ¬location:Chicago" is such an example.

In summary, we have seen several independent divisions in ABE. Schemes can be classified by policy placement (KP-ABE or CP-ABE), size of the attribute universe (small or large universe construction), control over the attributes (single authority or multi-authority setup) and the expressiveness of policies (monotonic or non-monotonic policies). A multi-authority setup can be further characterized by its requirement on a CA (centralized or decentralized setup).

Attribute revocation The support for attribute revocation does not fit well in our classification. However, since direct attribute revocation is problematic in all ABE schemes, we do mention it here.

It is difficult to create an ABE scheme that allows immediate access revocation without introducing any other drawback. Since the attributes are issued one time during Key Generation and decryption can take place off-line, the only way to prevent data access is to re-encrypt every file. Re-encryption requires the issuance of new keys, such that every user who is still granted access can decrypt. The scheme of Ibraimi et al. [IPN+09] introduces an on-line semi-trusted mediator to allow direct revocation, at the expense of requiring an on-line party.

5.2 Combining ABCs with ABE

The next few subsections discuss the types of ABE that suit the IRMA ecosystem best given the classification of the previous section. In this discussion, we have to keep in mind that the IRMA project focuses on simple access structures. Therefore, focus should be placed on access policies with two or three attributes joined together by one or two operators. Note that monotone OR

access policies can easily be emulated by encrypting the plaintext twice with different attributes. However, all other operators that we want to use, have to be incorporated in the scheme.

We will have a clear understanding of the requirements of our scheme at the end of this section. Concrete, existing schemes will be discussed on their privacy friendliness and computational efficiency in the next sections.

5.2.1 KP-ABE vs. CP-ABE

Although CP-ABE and KP-ABE both have their advantages in different scenarios. The CP-ABE variant would probably better fit the IRMA ecosystem because the idea behind the IRMA project is to place the *user* in the center, instead of an attribute authority or key generation authority. The user should be in full control of the secrecy of his message, not some authority. The encryptor will not be able to determine who has access to the plaintext in a monotone KP-ABE scheme, whereas in a CP-ABE scheme, the encryptor can determine who can decipher the obtained ciphertext.

As a side note, we observe that when the key issuer is also the encryptor of *all* messages, e.g., in the case of broadcast encryption, the choice between KP-ABE and CP-ABE is irrelevant. This is because the key issuer (KP-ABE) or the encryptor (CP-ABE) determines the access policy, and the policy placement *itself* does not effect the decryption capabilities of the decryptor.

5.2.2 Small vs. Large Universe Construction

To decide which construction is preferred, we need to estimate how many attributes will be used by a key generation authority (KGA). If each KGA uses a small set of attributes, the small universe seems more desirable, as it is in general less computationally intensive. However, if a KGA uses many attributes, key storage might become a problem and a large universe construction becomes more attractive. The large universe construction is far more desirable if we want to create many specific attributes. For example if we create an attribute for every zip code or every family name.

In a small universe construction, an attribute needs to be defined before a policy can be defined over that attribute. Thus, in contrast to the large universe construction, encryption can only take place after the attribute is explicitly defined. This is an insignificant difference if the small universe construction generates a fixed set of attributes during Setup. However, if new attributes can be created after Setup and the KGA would add new attributes to the system on request[1], privacy related problems may arise. This is because, in such a scenario, a user could not possibly store every attribute, as new attributes can be added at any time. Instead, the user would request the public or private key for an attribute—existing or not—when needed. This poses a privacy threat since the KGA could keep a record of all the requested attributes to see how popular each attribute is. In the case of an identifying attribute, e.g., a Social Security Number, this becomes even more a privacy threat, as the KGA now learns how many people request the public key of that specific person.

[1] Note that the security model has to allow this. Currently, most models do not allow this.

Finally, we note that the ABC system used by the IRMA project can be regarded as a large universe construction: the set of possible attributes does not have to be explicitly defined.

5.2.3 Single-Authority vs. Multi-Authority Setup

In a practical scenario there will not be a single KGA controlling all the attributes, but instead there will be multiple KGAs each responsible for their own set of attributes. If each KGA has nothing to do with any of the other KGAs, then every KGA can implement their own ABE scheme, i.e., each authority runs the Setup algorithm individually. However, if attributes are issued by different KGAs and a policy is defined across these attributes, a multi-authority setup is required. An example case for such a multi-authority setup is a movie rental service that encrypts the movies with their own 'license' attribute, but requires some age attribute issued by the government to check if the decryptor is old enough to watch the movie.

For efficiency reasons, it is important that new KGAs could easily join the system and that every KGA does not have to be aware of other KGAs, just like this is the case with attribute authorities (AAs) in the ABC setup of the IRMA project. A decentralized setup is desired in order to prevent the need for a CA where every KGA needs to register.

5.2.4 Monotonic vs. Non-Monotonic Access Structures

Most of the non-monotonic schemes are inefficient; typically they are computationally intensive, require a lot of storage space, or have other limitations. Efficient non-monotonic schemes do exist, but those have other drawbacks. For example, in the non-monotonic scheme of Chen, Zhang, and Feng [CZF11] only one AND node is allowed in the access tree and in the non-monotonic scheme of Li et al. [LXZ+13] the computations are still quite intensive, although the storage space for the ciphertext is made constant.

Putting these drawbacks aside, there is also a practical problem of deciding whether one should possess an attribute or not. It is easy to prove that you are a student, but how do you prove that you are *not* a student? Clearly, a statement from all universities that the user is not a student would do so, but it is highly impractical to obtain.

Currently, the extra expressiveness of policies in monotonic schemes does not add much compared to the efficient non-monotonic schemes. Moreover, the use of a monotonic access structure suffices in most practical scenarios.

5.2.5 Requirements of a Suitable Scheme

The most important requirement for an ABE scheme in the IRMA ecosystem is the use of a decentralized multi-authority setup. Additionally, a CP-ABE scheme is highly preferred, as it allows the users to take full control of the secrecy of their data. The use of a monotonic scheme suffices, although an efficient non-monotonic scheme also could be used. Finally, we do not have a marked preference for the type of attribute universe: both the small and the large universe construction can be used.

5.3 Comparison of Different ABE Schemes

Since the work of Sahai and Waters [SW05] in 2005 many new ABE schemes have been developed. We summarize several features of a selection of interesting ABE schemes in different tables. This selection is made subjectively based on the most innovative and characteristic ABE schemes. A further consideration is the scheme's potential usefulness in the IRMA environment.

In Table 5.1 the classification of Section 5.1 is used to arrange the different schemes. An overview of the complexity assumptions for the schemes is given in Table 5.2 and Table 5.3 lists the computational and storage costs.

It would be interesting to compare the findings from the literature review to a survey paper or other recapitulating document of another author. Sadly, to the best of my knowledge, only one survey paper on ABE has been published. Lee, Chung, and Hwang [LCH13] focus on six different aspects of an ABE scheme: data confidentiality, fine-grained access control, scalability, user accountability, user revocation, and collusion resistance. They discuss five different schemes, [SW05; GPS+06; BSW07; OSW07; WLW+11], on these aspects. It is hard to compare their findings with ours, as they discuss different papers on different aspects. However, I disagree with them on some key points. For example, they state that the [BSW07] scheme allows direct revocation or that [SW05] does not provide data confidentiality, whereas I think that the [BSW07] scheme does not allow *immediate* revocation (this is only possible with an on-line party) and [SW05] certainly provides data confidentiality (an unauthorized user definitely cannot decrypt ciphertexts).

5.3.1 Security of Different ABE Schemes

Almost every paper defines its own specific security game, so it is hard to compare the security of schemes. However, we can see in Table 5.2 that most of the discussed schemes use the Decisional Bilinear Diffie–Hellman (DBDH) assumption or a similar assumption to prove the security of the scheme. Several newer schemes try to provide stronger security models without becoming much more inefficient [ITH+09; LW11; RW13].

The mentioned complexity assumptions can be found in Section 2.6.2 or Appendix A.

5.3.2 Computational and Storage Costs

Table 5.3 lists the computational and storage costs of various schemes. The symbols are explained on Page 44, right after the table. The computational costs are expressed in the number of different operations that are used. The storage costs are expressed in the number of elements that need to be stored.

The Encryption cost $E_{\mathbb{G}_T} + M_{\mathbb{G}_T}$, required in *every* scheme for the computation of $Me(g,g)^s$, is left out in the table, in order to save space. Also, all polynomial calculations, e.g., the evaluation of $y(0)$ in $Y^{y(0)}$, are left out to increase readability.

The precise computational costs depend on the exact implementation, e.g., if intermediate results are stored or recomputed, and on the access structure, e.g., the policy $A \wedge (B \vee C)$ is similar to $(A \wedge B) \vee (A \wedge C)$, but the first might be

more efficient to implement. The table tries to give a good, extensive overview; however, the original papers of the table should be consulted for the exact costs.

It is interesting to note that the schemes [SW05] II, [GPS+06] II, and [Cha07] II are so inefficient since they require the evaluation of a computationally heavy function $T(x)$ for each attribute. These schemes implement a large universe construction. In general, large universe constructions require relatively more heavy computations. All listed large universe construction schemes require at least $2|\omega_d|P$.

The most efficient decryption schemes in this overview are [CZF11] and [RD13]: they only require a fixed number of bilinear pairings for any access structure.

scheme	type	universe	policy	multi-auth.	points of interest
[SW05] I	n/a	small	threshold	no	n/a
[SW05] II	n/a	large	threshold	no	n/a
[GPS+06] I	KP	small	monotonic k-OUT-OF-n access tree	no	policy
[GPS+06] II	KP	large	monotonic k-OUT-OF-n access tree	no	policy
[BSW07]	CP	large	monotonic k-OUT-OF-n access tree	no	CP
[CN07] I	CP	small	non-monotonic AND statement	no	non-monotonic
[CN07] II	CP	small	non-monotonic AND statement	no	computational efficiency
[Cha07] I	KP	small	threshold	centralized	multi-authority
[Cha07] II	KP	large	monotonic k-OUT-OF-n access tree	centralized	multi-authority, policy
[CC09]	KP	small	threshold	decentralized	generic protocol, decentralized setup
[ITH+09] I	CP	small	monotonic AND, OR tree	no	computational efficiency
[IPN+09] I	CP	small	monotonic AND, OR tree	no	delegation and revocation
[IPN+09] II	CP	small	monotonic AND, OR tree	centralized	multi-authority
[LW11]	CP	small	monotonic LSSS access matrix	decentralized	decentralization
[CZF11]	CP	small	non-monotonic AND statement	decentralized	non-monotonic, efficiency
[RW13] I	CP	large	monotonic LSSS access matrix	no	large universe, efficiency, security in the standard model
[RW13] II	KP	large	monotonic LSSS access matrix	no	large universe, efficiency, security in the standard model
[RD13]	CP	small	monotonic access structure	decentralized	fast decryption

Table 5.1: Classification of different ABE schemes.

scheme	based on	complexity assumption	model
[SW05] I	IBE	Decisional Modified Bilinear Diffie–Hellman	fuzzy selective ID (sid)
[SW05] II	IBE	DBDH	fuzzy sid
[GPS+06] I	[SW05] I	DBDH	selective-set (CPA)
[GPS+06] II	[SW05] II	DBDH	selective-set (CPA)
[BSW07]	[GPS+06], [SW05]	generic group heuristic	generic bilinear group (CPA)
[CN07] I	[SW05] II	DBDH	sid (CPA & CCA)
[CN07] II	[CN07] I	DBDH	sid (CPA)
[Cha07] I	[SW05] I	DBDH	sid
[Cha07] II	[GPS+06] II	DBDH	sid[2]
[CC09]	[SW05]	DBDH & q-Decisional Diffie–Hellman Inversion (DDHI)	selective-attribute attack (sAtt)
[ITH+09] I		DBDH	ciphertext indistinguishability (IND)-sAtt-CPA
[IPN+09] I	[CN07]	DL & Diffie–Hellman (DH)	generic group
[IPN+09] II	[Cha07]	No security proof provided	
[LW11]		Subgroup decision for 3 primes, and others (Definitions 13 to 15)	static corruption
[CZF11]		decisional n-BDHE	selective security (CPA & (extension) CCA)
[RW13] I	[LW11]	q-1	selective security
[RW13] II		q-2	selective security
[RD13]	[LW11]	generic group heuristic	generic bilinear group (IND-CPA)

Table 5.2: Security of different ABE schemes.

[2]The security proof should be the full version of the paper; however, the full version is not published.

scheme	Key Generation costs	Encryption costs	Decryption costs	PK storage	SK storage
[SW05] I	$\lvert\omega_o\rvert(E_{\mathbb{G}_1} + I_{\mathbb{Z}_p})$	$\lvert\omega_p\rvert E_{\mathbb{G}_1}$	$\lvert\omega_d\rvert(P + M_{\mathbb{G}_T}) + I_{\mathbb{G}_T}$	$\lvert\Omega\rvert\mathbb{G}_1 + \mathbb{G}_T$	$\lvert\omega_o\rvert\mathbb{G}_1$
[SW05] II	$\lvert\omega_o\rvert(3(2E_{\mathbb{G}_1} + (\lvert\Omega\rvert + 1)(E_{\mathbb{G}_1} + M_{\mathbb{G}_1})) + M_{\mathbb{G}_1})$	$\lvert\omega_p\rvert(E_{\mathbb{G}_1} + (\lvert\Omega\rvert + 1)(E_{\mathbb{G}_1} + M_{\mathbb{G}_1})) + E_{\mathbb{G}_1}$	$\lvert\omega_d\rvert(2P + I_{\mathbb{G}_T} + E_{\mathbb{G}_T} + 2M_{\mathbb{G}_T})$	$(\lvert\Omega\rvert + 2)\mathbb{G}_1 + \mathbb{G}_T$	$2\lvert\omega_o\rvert\mathbb{G}_1$
[GPS+06] I	$\lvert\omega_o\rvert(E_{\mathbb{G}_1} + I_{\mathbb{Z}_p})$	$\lvert\omega_p\rvert E_{\mathbb{G}_1}$	$\lvert\omega_p\rvert P + \lvert\mathcal{T}\rvert E_{\mathbb{G}_T} + (\lvert\mathcal{T}\rvert + 1)M_{\mathbb{G}_T} + I_{\mathbb{G}_T}$	$\lvert\Omega\rvert\mathbb{G}_1 + \mathbb{G}_T$	$\lvert\omega_o\rvert\mathbb{G}_1$
[GPS+06] II	$\lvert\omega_o\rvert(3(2E_{\mathbb{G}_1} + (\lvert\Omega\rvert + 1)(E_{\mathbb{G}_1} + M_{\mathbb{G}_1})) + M_{\mathbb{G}_1})$	$\lvert\omega_p\rvert(E_{\mathbb{G}_1} + (\lvert\Omega\rvert + 1)(E_{\mathbb{G}_1} + M_{\mathbb{G}_1})) + E_{\mathbb{G}_1}$	$\lvert\omega_d\rvert(2P + I_{\mathbb{G}_T} + M_{\mathbb{G}_T}) + \lvert\mathcal{T}\rvert E_{\mathbb{G}_T} + (\lvert\mathcal{T}\rvert + 1)M_{\mathbb{G}_T} + I_{\mathbb{G}_T}$	$(\lvert\Omega\rvert + 2)\mathbb{G}_1$	$2\lvert\omega_o\rvert\mathbb{G}_1$
[BSW07]	$\lvert\omega_o\rvert(2E_{\mathbb{G}_1} + M_{\mathbb{G}_1}) + 2E_{\mathbb{G}_1}$	$(2\lvert\omega_p\rvert + 1)E_{\mathbb{G}_1}$	$\lvert\omega_d\rvert(2P + I_{\mathbb{G}_T} + M_{\mathbb{G}_T}) + P + \lvert\mathcal{T}\rvert E_{\mathbb{G}_T} + (\lvert\mathcal{T}\rvert + 2)M_{\mathbb{G}_T} + I_{\mathbb{G}_T}$	$2\mathbb{G}_1 + \mathbb{G}_T$	$(2\lvert\omega_o\rvert + 1)\mathbb{G}_1$
[CN07] I	$(\lvert 2\Omega\rvert + 1)E_{\mathbb{G}_1}$	$(\lvert\Omega\rvert + 1)E_{\mathbb{G}_1}$	$(\lvert\Omega\rvert + 1)P + (\lvert\Omega\rvert + 1)M_{\mathbb{G}_T} + I_{\mathbb{G}_T}$	$(3\lvert\Omega\rvert + 1)\mathbb{G}_1 + \mathbb{G}_T$	$(2\lvert\Omega\rvert + 1)\mathbb{G}_1$
[CN07] II	$(\lvert 2\Omega\rvert + 1)E_{\mathbb{G}_1} + \lvert\mathcal{T}\rvert E_{\mathbb{G}_1}$	$(\lvert\Omega\rvert + 1)E_{\mathbb{G}_1}$	$(\lvert\Omega\rvert + \lvert\mathcal{T}\rvert)P + (\lvert\mathcal{T}\rvert + 1)M_{\mathbb{G}_T} + I_{\mathbb{G}_T}$	$(3\lvert\Omega\rvert + 1)\mathbb{G}_1 + \mathbb{G}_T$	$(2\lvert\Omega\rvert + \lvert\mathcal{T}\rvert + 1)\mathbb{G}_1 + \mathbb{G}_T$
[Cha07] I	$\lvert\omega_o\rvert(E_{\mathbb{G}_1} + I_{\mathbb{Z}_p})$	$(\lvert\omega_p\rvert + 1)E_{\mathbb{G}_1}$	$\lvert\omega_d\rvert(P + E_{\mathbb{G}_T} + M_{\mathbb{G}_T}) + P + (\lvert\mathcal{A}_d\rvert + 1)M_{\mathbb{G}_T} + I_{\mathbb{G}_T}$	$(\lvert\Omega\rvert + 1)\mathbb{G}_1 + \mathbb{G}_T$	$\lvert\omega_o\rvert\mathbb{G}_1$
[Cha07] II	$\lvert\omega_o\rvert(3(2E_{\mathbb{G}_1} + (\lvert\Omega\rvert + 1)(E_{\mathbb{G}_1} + M_{\mathbb{G}_1})) + M_{\mathbb{G}_1})$	$(\lvert\omega_p\rvert + 1)E_{\mathbb{G}_1}$	$\lvert\omega_d\rvert(2P + I_{\mathbb{G}_T} + M_{\mathbb{G}_T}) + P + 2\lvert\mathcal{T}\rvert E_{\mathbb{G}_T} + (2\lvert\mathcal{T}\rvert + 1)M_{\mathbb{G}_T} + I_{\mathbb{G}_T}$	$(\lvert\Omega\rvert + 3)\mathbb{G}_1 + \mathbb{G}_T$	$2\lvert\omega_o\rvert\mathbb{G}_1$
[CC09]	Not relevant; the paper is interesting because of the protocol				
[ITH+09] I	$(\lvert\omega_o\rvert + 1)E_{\mathbb{G}_1}$	$(\lvert\omega_p\rvert + 1)E_{\mathbb{G}_1}$	$(\lvert\omega_d\rvert + 1)(P + M_{\mathbb{G}_T}) + I_{\mathbb{G}_T}$	$(\lvert\Omega\rvert + 1)\mathbb{G}_1 + \mathbb{G}_T$	$(\lvert\omega_o\rvert + 1)\mathbb{G}_1$
[IPN+09] I	$(2\lvert\omega_o\rvert + 1)E_{\mathbb{G}_1}$	$(\lvert\omega_p\rvert + 1)E_{\mathbb{G}_1}$	$(\lvert\omega_d\rvert + 1)(P + M_{\mathbb{G}_T}) + I_{\mathbb{G}_T}$	$(\lvert\Omega\rvert + 1)\mathbb{G}_1 + \mathbb{G}_T$	$(\lvert\omega_o\rvert + 1)\mathbb{G}_1$
[IPN+09] II	$(2\lvert\omega_o\rvert + 1)E_{\mathbb{G}_1}$	$(\lvert\omega_p\rvert + 1)E_{\mathbb{G}_1}$	$(\lvert\omega_d\rvert + 1)(P + M_{\mathbb{G}_T}) + I_{\mathbb{G}_T}$	$(\lvert\Omega\rvert + 1)\mathbb{G}_1 + \mathbb{G}_T$	$(\lvert\omega_o\rvert + 1)\mathbb{G}_1$

Continues on next page...

scheme	Key Generation costs	Encryption costs	Decryption costs	PK storage	SK storage
[LW11]	$\lvert\omega_o\rvert(2E_{\mathbb{G}_1} + M_{\mathbb{G}_1})$	$\lvert\omega_p\rvert(2E_{\mathbb{G}_T} + M_{\mathbb{G}_T} + 3E_{\mathbb{G}_1} + M_{\mathbb{G}_1})$	$\lvert\omega_d\rvert(2P + E_{\mathbb{G}_T} + I_{\mathbb{G}_T} + 3M_{\mathbb{G}_T}) + I_{\mathbb{G}_T}$	$(\lvert\Omega\rvert+1)\mathbb{G}_1+(\lvert\mathcal{A}\rvert+1)\mathbb{G}_T$	$(\lvert\omega_o\rvert+1)\mathbb{G}_1$
[CZF11]	$\lvert\Omega\rvert(E_{\mathbb{G}_1} + M_{\mathbb{G}_1})$	$2E_{\mathbb{G}_1} + (\lvert\omega_p\rvert - 1)(M_{\mathbb{G}_1} + M_{\mathbb{G}_T})$	$2P + (\lvert\omega_p\rvert - 1)M_{\mathbb{G}_1} + I_{\mathbb{G}_T} + 2M_{\mathbb{G}_T}$	$2\lvert\Omega\rvert(\mathbb{G}_1 + \mathbb{G}_T)$	$(\lvert\omega_o\rvert+1)\mathbb{G}_1$
[RW13] I	$\lvert\omega_o\rvert(4E_{\mathbb{G}_1} + 3M_{\mathbb{G}_1}) + 3E_{\mathbb{G}_1}$	$\lvert\omega_p\rvert(5E_{\mathbb{G}_1} + 2M_{\mathbb{G}_1}) + E_{\mathbb{G}_1}$	$3\lvert\omega_d\rvert(P+E_{\mathbb{G}_T}+M_{\mathbb{G}_T})+P+I_{\mathbb{G}_T}$	$5\mathbb{G}_1 + \mathbb{G}_T$	$(2\lvert\omega_o\rvert+2)\mathbb{G}_1$
[RW13] II	$\lvert\omega_o\rvert(5E_{\mathbb{G}_1} + 2M_{\mathbb{G}_1})$	$\lvert\omega_p\rvert(4E_{\mathbb{G}_1} + 2M_{\mathbb{G}_1}) + E_{\mathbb{G}_1}$	$\lvert\omega_d\rvert(3P+E_{\mathbb{G}_T}+M_{\mathbb{G}_T})+I_{\mathbb{G}_T}$	$4\mathbb{G}_1 + \mathbb{G}_T$	$3\lvert\omega_o\rvert\mathbb{G}_1$
[RD13]	$\lvert\omega_o\rvert(2E_{\mathbb{G}_1} + M_{\mathbb{G}_1})$	$(\lvert\vee\rvert - 1)\big[(\lvert\wedge\rvert - 1)(M_{\mathbb{G}_1}+M_{\mathbb{G}_T})+3E_{\mathbb{G}_1}\big]$	$2P + I_{\mathbb{G}_T} + (\lvert\mathcal{A}_d\rvert - 1)M_{\mathbb{G}_1} + M_{\mathbb{G}_T}$	$\lvert\Omega\rvert(\mathbb{G}_1 + \mathbb{G}_T)$	$\lvert\omega_o\rvert\mathbb{G}_1$

Table 5.3: Computational and storage costs of ABE schemes.

P bilinear pairing $E_{\mathbb{G}}$ exponentiation in group $\mathbb{G}$ $M_{\mathbb{G}}$ multiplication in group $\mathbb{G}$ $I_{\mathbb{Z}}$ modular inversion in group $\mathbb{Z}$

$|\Omega|$ total number of attributes (determined in Setup)

$|\omega_d|$ minimum number of required attributes to match the policy

$|\omega_p|$ number of attributes used in the policy

$|\omega_o|$ number of attributes owned by the user

$|\mathcal{A}|$ total number of different authorities (determined in Setup)

$|\mathcal{A}_d|$ number of different authorities required to match the policy

$|\cdot|$ number of $\cdot$ operators (AND or OR) defined in the policy

$|\mathcal{T}|$ a number dependent on the structure of the access tree

$\mathcal{T}$ Access tree (policy)

$\mathcal{S}$ Attribute set

PK The public parameters of the scheme

SK A decryption key for a user of the scheme

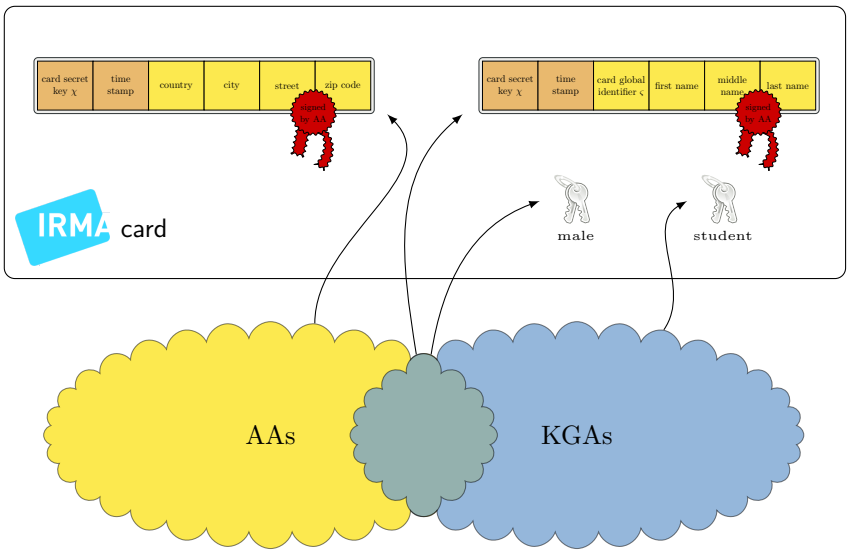

Figure 5.1: Diagram of the with ABE extended IRMA ecosystem.

5.4 The IRMA Ecosystem with ABE

Section 5.2 made clear what the requirements are for an ABE scheme in the IRMA ecosystem. The most important requirement is to have a multi-authority setup. Generally, only decentralized setups where KGAs operate independently are practical in large scale implementation, due to scalability problems with centralized setups. In this section, we will sketch what the IRMA ecosystem may look like, when enriched with a decentralized multi-authority CP-ABE scheme.

We can distinguish two types of authorities, attribute authorities (AAs) and key generation authorities (KGAs). The AAs can issue credentials to the IRMA card that can be used for ABC authentication. The KGAs can issue private keys, associated with an attribute, to the IRMA card that can be used to decrypt ABE messages. A single authority can act as both an AA and a KGA. However, the use of an attribute by some AA can be completely independent of the use by some KGA. For example, a university might use the 'student' attribute from the government's AA to provide access to their library, but use the 'student' attribute issued by their own KGA to encrypt their online courses with. Figure 5.1 schematically depicts the described IRMA ecosystem.

If multiple universities use an attribute labeled as 'student', one might think that this will create a conflicting situation. Luckily, this is not the case in the cryptographic part of the scheme. To encrypt to a KGA attribute we need the public key of that attribute. This public key is merely derived from a randomly chosen master key and the label 'student' is just the name for the public key. Thus, when several 'student' attributes exist from different KGAs, they just point to different public keys from different authorities. The choice of the public key determines which 'student' can decrypt the ciphertext.

5.5 Suitable ABE Schemes for the IRMA Ecosystem

Many papers have been published on multi-authority ABE, although not all papers are sufficiently practical. We briefly describe the most prominent papers published on multi-authority ABE (MA-ABE).

Chase [Cha07] is the first to affirm the existence of a MA-ABE scheme. Her scheme uses a unique global identifier (GID) for each user in order to prevent user collusion. It also requires a central authority (CA) which is, unfortunately, able to decrypt all the messages. The scheme of Lin et al. [LCL+08] removes the CA, but also fixes the set of authorities at Setup. Furthermore, their scheme provides only limited collusion resistance.

Chase and Chow [CC09] propose a new multi-authority scheme that eliminates the need of the CA and the disclosure of the user's GID. By not disclosing the user's GID, their scheme becomes more privacy-friendly as we will explain in Chapter 6. They created an anonymous ABE key issuing protocol which uses *secure two-party computations* and *zero-knowledge proofs*. However, the algorithms in their scheme are inefficient and Key Generation requires communication between all key generation authorities. Moreover, no new KGA can be added to their system after the Setup. Li et al. [LRR+13] extend the work of Chase and Chow [CC09] by introducing an on-line Semi-Trusted Authority (STA) which accounts for many of the communications and computations. However, since their work does not substantially change the scheme of [CC09], many of the drawbacks in [CC09] apply to [LRR+13] too.

The scheme by Müller, Katzenbeisser, and Eckert [MKE09a; MKE09b] offers the first multi-authority CP-ABE solution, but those solutions require the use of a single CA which can decrypt any ciphertext. Later, Liu et al. [LCH+11] developed a new scheme that allows multiple CAs and multiple KGAs. The KGAs operate independently of each other, although they still have to register at every CA; besides, users are linkable if some KGAs collude. The first decentralized multi-authority CP-ABE scheme is introduced by Lewko and Waters [LW11]. They created a small universe construction where new KGAs could be added anytime after the initial Global Setup. This scheme is also adaptively secure, in contrast to the more restricted selective security model that is used by most of the previous schemes.[3] However, it does require the use of a GID. The non-monotonic CP-ABE scheme by Chen, Zhang, and Feng [CZF11] can be extended to a decentralized multi-authority setup, by replacing the random value by the user's GID. The issued SK in their scheme is similar to the one used in [LW11]. The work of Li et al. [LXZ+13] shows a constant-size ciphertext KP-ABE scheme, but does not scale very well in the number of KGAs. Moreover, their scheme uses a GID too.

The multi-authority CP-ABE scheme by Rao and Dutta [RD13] further optimizes the [LW11] scheme, described in the full paper [LW10]. The [RD13] scheme allows fast decryption of ciphertext and it seems even possible to se-

[3] In the selective ID (sid) model, the adversary has to determine the attributes—or identity—which he wants to attack and select the corrupted KGAs, before receiving the public parameter. The adaptive security model allows adaptive key queries of statically corrupted KGAs and "additionally allows the adversary to choose the public keys of the corrupted authorities for himself, instead of having these initially generated by the challenger" as in the selective security model [LW11].

curely decrease the Decrypt costs further by outsourcing one of the two required bilinear pairing computations to another device. However, the major drawback of the [RD13] scheme remains that it is not proven secure in a standard model, but in the generic bilinear group heuristic of [BSW07] instead.

Han et al. [HSM+12] were the first to create a practical, privacy-friendly, decentralized KP-ABE scheme based on standard complexity assumptions. They took an interesting approach where a secure two-party computation is used to compute the private key based on the GID, without the user having to reveal his GID. Gao and Li [GL13] also noticed the privacy-violating aspects of the usage of a GID and propose a GID free scheme. However, they fail to design a fully secure scheme against multi-collusion attacks. In the same year, Qian, Li, and Zhang [QLZ13] introduce a privacy-friendly decentralized CP-ABE scheme, where the KGA learns nothing about the user's GID and the access structure from the ciphertext is fully hidden. By hiding the complete access structure, nothing can be deduced from the ciphertext. Their Decrypt algorithm is inefficient as a result of using a hidden access structure. In 2014, Han et al. [HSM+14] propose a new privacy-friendly decentralized CP-ABE scheme in their recently submitted ePrint. They are the first to hide the user's attributes in addition to the user's GID when an authority is requested for the user's decryption key. However, their Decrypt algorithm is compared to other MA-ABE schemes very inefficient as it requires at least 8 bilinear pairings for an access structure that contains only two attributes.

We conclude that there exist different decentralized MA-ABE schemes; some are efficient, others are more privacy-friendly. Practical decentralized ABE schemes, e.g., [LW11; RD13], are not privacy-friendly. They require the user to reveal his unique global identifier before he is issued a decryption key. The multi-authority schemes have to rely on such a GID to prevent user collusion.

5.6 Privacy-Friendly Decentralized MA-ABE

Related to the concept of privacy-friendly decentralized MA-ABE is the concept of blind IBE and blind ABE. In an IBE scheme the user needs to provide the key generation center (KGC) with his identity in order to receive the private key for his identity. In a *blind* IBE scheme, the user can obtain his private key without the KGC learning anything about the user's identity, i.e., the KGC does not know the public key for which it just issued the private key. The first blind IBE scheme is proposed by Green and Hohenberger [GH07]. They use their scheme as a tool for the construction of an *oblivious transfer* protocol. Camenisch et al. [CKR+09] use a blind and anonymous IBE scheme to construct a system of public-key encryption with oblivious keyword search (PEOKS). Their scheme is *anonymous* in the sense that it provides key privacy: given a ciphertext, it is impossible to determine which public-key was used to create that ciphertext [BBD+01]. Xu and Zhang [XZ11] create a blind ABE scheme analogous to the work of [GH07]. They use the scheme to create an oblivious transfer protocol which allows complex attribute-based access control policies. Their KGC does not learn anything about the attributes the user possesses in their Blind Key Generation protocol.

Han et al. [HSM+12] create a privacy-friendly decentralized KP-ABE scheme based on blind IBE. In their work they first describe their basic KP-ABE

scheme and prove it secure under the standard DBDH complexity assumption. It is important to note that their basic scheme is not privacy-friendly: in order to prevent user collusion, the KGA must require the user to reveal his GID before he can receive his private key. In the second part of their paper, they replace the privacy-violating Key Generation algorithm with a new Blind Key Generation protocol. Their Blind Key Generation protocol uses commitments and zero-knowledge proofs to securely compute the user's SK without requiring the user to reveal his GID.

Qian, Li, and Zhang [QLZ13] create a privacy-friendly decentralized CP-ABE scheme with fully hidden access structure. By hiding the access structure, i.e., not including the access structure in the ciphertext, their Decrypt algorithm has become significantly inefficient if the user does not know whether he is allowed to decrypt the ciphertext. Similar to the [HSM$^+$12] scheme, they replace their initial Key Generation algorithm with a Blind Key Generation protocol.

Chapter 6

Privacy-Friendly Decentralized ABE with ABC

In this chapter we will modify the decentralized multi-authority CP-ABE scheme by Lewko and Waters [LW11] to obtain a privacy-friendly MA-ABE scheme that can form a part of the IRMA ecosystem. The unmodified scheme satisfies most of the requirements discussed in Chapter 5 and is proven secure in the standard model. However, the scheme is not privacy-friendly.

The [LW11] scheme violates the user's privacy by requiring the user to authenticate with a unique global identifier (GID) before he is issued a decryption key for any attribute. The user is required to do so, because the Key Generation algorithm has to bind the user's GID to the issued decryption key in order to prevent user collusion. In order to make the scheme privacy-friendly, we will replace the Key Generation algorithm with a privacy-friendly Blind Key Generation protocol between the user and the KGA. This approach is similar to the approach taken by several others [HSM+12; QLZ13; HSM+14], and, to a lesser extent, similar to the [CC09] scheme. The protocol will use the ABC technology that is used by the IRMA project, together with a *secure two-party computation* to bind the user-specific decryption key to an IRMA card. In this way, the IRMA card can be used for Attribute-Based Credentials and Attribute-Based Encryption.

6.1 Decentralized Multi-Authority ABE Scheme

We start off with reviewing the unmodified decentralized multi-authority scheme of Lewko and Waters [LW11]. The scheme does not make use of access trees like the scheme discussed in Section 4.5, but uses access matrices. Just like we did in Section 4.5, we will first give an example how to convert an access structure into an access matrix.

Conversion to an access matrix The next example converts the policy "(data analyst AND (mathematician OR senior manager)) OR executive board"

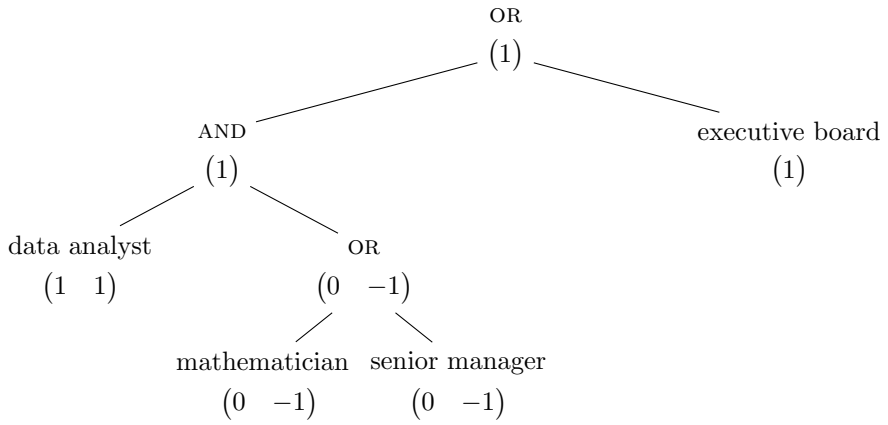

Figure 6.1: A graphical representation of the tree for the policy "(data analyst AND (mathematician OR senior manager)) OR executive board".

into an access matrix using Algorithm 3 on Page 31.

Example 6.1 (Access matrix)**.** We start off by describing the policy as a tree and set the counter value $c = 1$. As root node we have a node described by OR and labeled with (1). Below is the "executive board" leaf node and a node described by AND. Both nodes inherit the vector (1). The AND node has two children: the leaf node "data analyst" and an OR node. The nodes are labeled with $(1 \quad 1)$ and $(0 \quad -1)$, respectively. We increment the AND counter to $c = 2$. Note that $(1 \quad 1) + (0 \quad -1) = (1 \quad 0)$. The rest of the nodes are children of this OR node and those inherit the vector $(0 \quad -1)$. Figure 6.1 shows the tree.

We now pad the shorter vectors with the number zero to obtain vectors of the same length, and stack the vectors of each leaf node on top of each other to obtain the matrix

$$\left.\begin{pmatrix} 1 & 1 \\ 0 & -1 \\ 0 & -1 \\ 1 & 0 \end{pmatrix}\right\} \left\langle \begin{array}{l} \text{data analyst} \\ \text{mathematician} \\ \text{senior manager} \\ \text{executive board} \end{array} \right. .$$

We may define a mapping ρ which associates each row with the associated attribute. Note the vector $(1 \quad 0)$ is in the subset of $\mathbb{R}^2$ spanned by a subset of the rows of this matrix, if and only if the attributes corresponding to these rows satisfy the policy "(data analyst AND (mathematician OR senior manager)) OR executive board".

The access matrix is used in the Encrypt and Decrypt algorithms of the described MA-ABE scheme.

The Lewko and Waters scheme A decentralized multi-authority ABE scheme consists of two Setup algorithms, in contrast to the one used in single-authority ABE schemes. The Global Setup algorithm, which may be run by a TTP, initializes the global public parameters. The Authority Setup, which needs to be executed by every KGA that joins the system, generates a public-private key pair for each attribute of the executing KGA. In order to encrypt a message,

the user needs the public key of each attribute of his access policy—possibly from different KGAs—and the public parameters.

Scheme (Lewko and Waters [LW11])**.** The five algorithms of the scheme are described one by one. The description uses the symbols that are defined in Chapter 2. Recall that $\mathcal{S}_x$ denotes the attribute set of party x.

Global Setup Select a bilinear group $\mathbb{G}$ of composite order of three primes $\Gamma = p_1p_2p_3$, with pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Pick a generator g_1 of the bilinear subgroup $\mathbb{G}_{p_1}$. The system parameters are $(\mathbb{G}, e, \Gamma, g_1)$.[1]

Authority Setup Key generation authority A_j can create a public-private key pair for attribute i. The private key for attribute i is a tuple of two random numbers, $\alpha_i, \beta_i \in_R \mathbb{Z}_\Gamma$. The corresponding public key is $\big(h_i = e(g_1, g_1)^{\alpha_i}, u_i = g_1^{\beta_i}\big)$.

We define the private master key of authority A_j, MK_{A_j}, as the set

$$\mathrm{MK}_{A_j} = \big\{(\alpha_i, \beta_i) \mid \forall i \in \mathcal{S}_{A_j}\big\} \quad \text{with } \alpha_i, \beta_i \in_R \mathbb{Z}_\Gamma.$$

The public key of authority A_j is defined as the set

$$\mathrm{PK}_{A_j} = \big\{\big(h_i, u_i\big) \mid \forall i \in \mathcal{S}_{A_j}\big\} \quad \text{with } h_i = e(g_1, g_1)^{\alpha_i}, u_i = g_1^{\beta_i}.$$

Note that there is no relationship between the different attribute keys from the same authority.

Key Generation The private key SK_{U,A_j} issued by key generation authority A_j to a user U with $\mathrm{GID}_U \in \mathbb{G}$ is the set

$$\mathrm{SK}_{U,A_j} = \big\{g_1^{\alpha_i}\mathrm{GID}_U^{\beta_i} \mid \forall i \in \mathcal{S}'_U\big\},$$

where $\mathcal{S}'_U \subseteq \mathcal{S}_U \cap \mathcal{S}_{A_j}$. So, user U may decide to request only a subset of all attributes the user is entitled to and request the other attributes another time.

Encrypt The algorithm encrypts a message $M \in \mathbb{G}_T$ with the public key under the access structure $\mathcal{A}$. Let A be the access matrix of dimensions $n \times m$, obtained by converting the access structure $\mathcal{A}$ to an access matrix using Algorithm 3. Denote the mapping of the rows of A to their corresponding attributes by the function ρ. Choose $s \in_R \mathbb{Z}_\Gamma$ and the random vectors $v, w \in_R {\mathbb{Z}_\Gamma}^m$, but set the first entry of v to s and the first entry of w to 0. For each row A_x of A, define $\nu_x = A_x v$ and $\omega_x = A_x w$, and choose $r_x \in_R \mathbb{Z}_\Gamma$.

The ciphertext is published as

$$\begin{aligned}\mathrm{CT} = ((A, \rho), C_0 = Me(g_1, g_1)^s,\\ \forall x \in \{1, \ldots, n\} :\ C_{1,x} = e(g_1, g_1)^{\nu_x} h_{\rho(x)}^{r_x},\\ C_{2,x} = g_1^{r_x}, C_{3,x} = u_{\rho(x)}^{r_x} g_1^{\omega_x}).\end{aligned}$$

[1] Note that the order of the group, Γ, is a public value. This is an important difference compared to the RSA construction, where the public modulus is composite and the order of the group is a *private* variable.

Decrypt The ciphertext CT can be decrypted by a user U if $\mathrm{SK}_U = \bigcup_{A_j} \mathrm{SK}_{U,A_j}$ can satisfy a subset of rows A_x of A such that $\begin{pmatrix}1 & 0 & \cdots & 0\end{pmatrix}$ is in the subset spanned by these rows. Let $\tilde{\mathrm{SK}}_{U,\rho(x)} = g_1^{\alpha_{\rho(x)}} \mathrm{GID}_U^{\beta_{\rho(x)}}$ be the user's decryption key for attribute $\rho(x)$ of row x. Select a minimal set of attributes $\tilde{\mathcal{S}}$ that the user possesses to satisfy the policy. Compute for each $x \in \tilde{\mathcal{S}}$

$$\begin{aligned}
\tilde{C}_x &= \frac{C_{1,x} \cdot e(\mathrm{GID}_U, C_{3,x})}{e(\tilde{\mathrm{SK}}_{U,\rho(x)}, C_{2,x})} \\
&= \frac{e(g_1,g_1)^{\nu_x} h_{\rho(x)}^{r_x} \cdot e(\mathrm{GID}_U, u_{\rho(x)}^{r_x} g_1^{\omega_x})}{e(g_1^{\alpha_{\rho(x)}} \mathrm{GID}_U^{\beta_{\rho(x)}}, g_1^{r_x})} \\
&= \frac{e(g_1,g_1)^{\nu_x + \alpha_{\rho(x)} r_x} \cdot e(\mathrm{GID}_U, g_1^{\beta_{\rho(x)} r_x + \omega_x})}{e(g_1^{\alpha_{\rho(x)}} \mathrm{GID}_U^{\beta_{\rho(x)}}, g_1^{r_x})} \\
&= \frac{e(g_1,g_1)^{\nu_x + \alpha_{\rho(x)} r_x} \cdot e(\mathrm{GID}_U, g_1)^{\beta_{\rho(x)} r_x + \omega_x}}{e(g_1,g_1)^{\alpha_{\rho(x)} r_x} e(\mathrm{GID}_U, g_1)^{\beta_{\rho(x)} r_x}} \\
&= e(g_1,g_1)^{\nu_x} e(\mathrm{GID}_U, g_1)^{\omega_x}.
\end{aligned}$$

Next, choose constants $c_x \in \mathbb{Z}_\Gamma$ in such a way that they satisfy $\sum_x c_x A_x = \begin{pmatrix}1 & 0 & \cdots & 0\end{pmatrix}$ and compute

$$\begin{aligned}
\tilde{C} &= \prod_{x \in \tilde{\mathcal{S}}} \tilde{C}_x^{c_x} \\
&= \prod_{x \in \tilde{\mathcal{S}}} e(g_1,g_1)^{\nu_x c_x} e(\mathrm{GID}_U, g_1)^{\omega_x c_x} \\
&= e(g_1,g_1)^s,
\end{aligned}$$

here we have used the fact that $\sum_x c_x \nu_x = \sum_x c_x A_x v = s$ and $\sum_x c_x \omega_x = \sum_x c_x A_x w = 0$, the first element of v and w respectively.

Finally, retrieve the message M by calculating

$$\begin{aligned}
M &= C_0 \cdot \tilde{C}^{-1} \\
&= M e(g_1,g_1)^s e(g_1,g_1)^{-s}.
\end{aligned}$$

We note that the Authority Setup and Key Generation algorithms of the schemes by Chen, Zhang, and Feng [CZF11], and Rao and Dutta [RD13] are nearly identical to the ones used in [LW11]; they all use a similar construction for the user's private key (SK). The only difference is that [LW11] uses a SK ($= g_1^{a_i} \mathrm{GID}_U^{\beta_i}$) that is created by multiplying an element of a subgroup ($g_1^{a_i}$) with an element of the whole group ($\mathrm{GID}_U^{\beta_i}$), whereas the other two schemes do not use a generator g_1 of subgroup ($\mathbb{G}_{p_1}$), but a generator g of the entire group ($\mathbb{G}$) instead.

6.1.1 Security of the Scheme

The composite order construction of the [LW11] scheme is proven secure in a static corruption model under the assumptions that are defined in Definitions 12

to 15. In the static corruption model, all authorities and the complete attribute universe are fixed during the setup phase of the security game. Additionally, the adversary chooses a static set of authorities during the setup phase. The game is not completely static: key queries can be made before and after the challenge phase. The adversary specifies two messages and an access structure in the challenge phase. The challenger encrypts one of the two messages using the given access structure and returns the ciphertext to the adversary. After the second key query phase, the adversary has to guess which message corresponds to the challenge ciphertext.

The security proof assumes that no attribute occurs twice in the same access structure. However, this is not a real restriction. Although access policies like "(student AND male) OR (student AND over-18)" cannot be securely used, they can be modified to circumvent this limitation. The above-mentioned access policy can be modified so that it only includes each attribute at most once, i.e., "student AND (male OR over-18)". Even if some complex access structure cannot easily be converted into a logical formula where each attribute occurs only once, we can still circumvent the problem. Lewko and Waters [LW10] propose that the KGA creates each attribute in duplicate—or in triplicate, in quadruple, etc.—that are all issued to the user if he has the right to possess such an attribute. An encryptor can now enforce an access structure where the attribute occurs multiple times by using a different attribute variant for each occurrence of the attribute. In our scenario we could ask the KGA to create a new attribute, e.g., student$_2$. This attribute has to be issued together with the 'student' attribute if a user is entitled to obtain the 'student' attribute. Now the access policy can be altered so that we obtain "(student AND male) OR (student$_2$ AND over-18)".

6.2 Blind Key Generation

In this section and onwards, we present our contribution. We describe the idea how we can create a privacy-friendly ABE scheme and how it fits in the IRMA ecosystem.

The problem with the Key Generation algorithm above is that the user has to provide his GID to the KGA before he can obtain his personal SK. This GID makes the user linkable and enables the authorities to *trace* its users. A curious authority can store all the GID he sees and discover what other attributes the user has, or compare the GIDs with another malicious KGA to learn what attributes the user has at the other party. To avoid this *profiling*, the Key Generation algorithm is replaced by a Blind Key Generation protocol. The Blind Key Generation protocol is a protocol between a user U and a KGA A_j. The parties jointly compute the SK for user U without revealing the user's GID and without KGA learning SK. The protocol must only succeed in issuing the SK if both parties participated honestly. This means that the KGA cannot learn the GID of the user and the user cannot learn the KGA's MK and is required to use his own GID.

6.2.1 Determining the GID Value

Before we can apply the MA-ABE scheme in practice, all KGAs have to agree on some GID for each user. Here the IRMA infrastructure turns out to be useful as we can now just store some unique random number on the IRMA card as an attribute and use this attribute as a global identifier. So, we use the ABC technique to let every KGA agree on a fixed GID for each user. The GID attribute could be part of the card's root credential and be issued when the card is initialized. Although this GID attribute is different from the card's master secret key χ, it has the same property that it should never be revealed.

To construct a value GID $\in \mathbb{G}$ we map the GID attribute value ς to an element in $\mathbb{G}$. This is done by choosing a generator h that generates the entire group $\mathbb{G}$, and use element $h^{\varsigma} \in \mathbb{G}$ as the GID in the MA-ABE scheme. This construction has the additional advantage that we can easily prove knowledge of ς using a zero-knowledge proof. To assure that the used GID $= h^{\varsigma}$ is unique, we have to select a unique value for ς and additionally require $0 \leq \varsigma < |\mathbb{G}|$ for every ς.

6.2.2 Private Key Issuance

Suppose Alice has an IRMA card capable of decrypting ciphertexts that were generated by the [LW11] ABE scheme. The root credential of her card contains the card's master secret key χ, a unique GID attribute ς, and several other attributes to store her full name. Alice additionally received a credential containing the "student" and "computer science" attribute from her university. Since her university offers online courses encrypted using ABE, she would like to obtain the decryption keys she is entitled to. The university does not issue the decryption keys themselves, but relies on another organization to issue the keys for them. We might think of an organization that issues decryption keys for several universities. Using her root credential and her university credential, she can request a decryption key associated with the "student" attribute. To receive the decryption key in a privacy-friendly manner, she runs the Blind Key Generation protocol with the organization, i.e., the KGA. We can define three distinct steps in our Blind Key Generation protocol that securely computes the SK.

1. First, the user must *blind* her GID using a commitment scheme and send the KGA this blinded value. In the same transfer, she authenticates to the KGA using a Σ-proof. This Σ-proof proves that her commitment belongs to the same identity as the root credential is issued to, and that the identity is entitled to receive the decryption key.

2. The KGA responds in the secure two-party computation by computing and sending a blinded SK if the received Σ-proof is valid. Only the user can unblind this blinded SK, using the opening value of the commitment that was used to blind the GID.

3. Finally, the user opens up the received blinded value and stores the decryption key on her IRMA card.

Now, when Alice has the decryption keys on her IRMA card, she can decrypt all messages with an access policy that can be satisfied by her decryption keys.

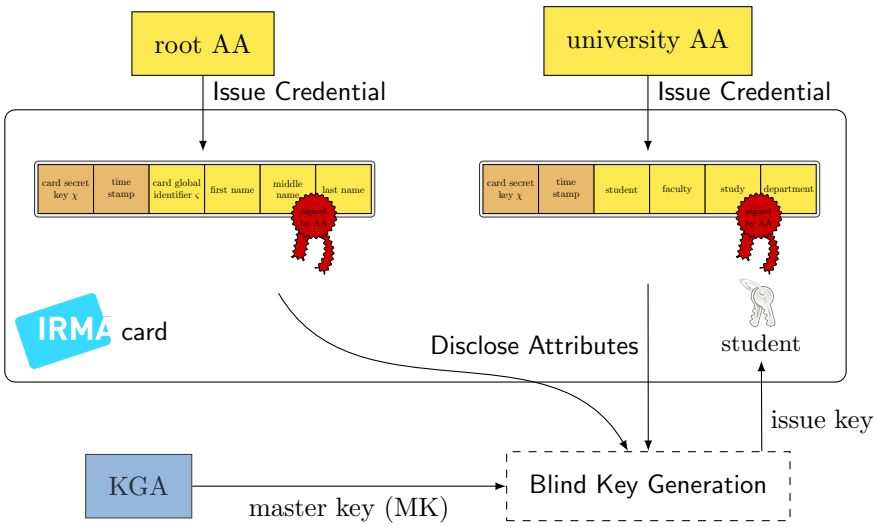

Figure 6.2: An IRMA card containing credentials and a decryption key.

Note that Alice can request other decryption keys, e.g., the "computer science" decryption key, using the same protocol. The users may not request more than one decryption key a time, to prevent the KGA to build a profile of its users. Figure 6.2 depicts Alice's IRMA card and illustrates how she obtained the values stored on her card.

6.2.3 Security Requirements

We first informally state three security requirements for our Blind Key Generation protocol and we give the formal definitions in the next section. The first two requirements protect the KGA against a dishonest user. The last requirement protects the user against a malicious KGA.

Secure authentication (based on [Lin10]) No user should be able to fool the server into issuing him a decryption key for an attribute he does not have.

Leak-freeness (based on [GH07]) A user only learns as much by executing the Blind Key Generation protocol with an honest authority as he would by executing the Key Generation protocol with an honest authority.

Selective-failure blindness (based on [GH07]) An authority learns nothing about an honest user's identity during the Blind Key Generation protocol; moreover, the KGA cannot cause the Blind Key Generation protocol to fail in a manner dependent on the user's GID.

Note that selective-failure blindness implies that only the user should be able to compute his final decryption key for some attribute i, as described by Lemma 2.

Lemma 2 (GID extraction)**.** *If the KGA would be able to calculate the decryption key for the user, it could easily extract the user's GID.*

Proof. Using the MK (α_i, β_i) for attribute i, the KGA could calculate the user's GID from the user's SK,

$$\begin{aligned}\left(\mathrm{SK} \cdot g_1^{-\alpha_i}\right)^{\frac{1}{\beta_i}} &= \left((g_1^{\alpha_i}\mathrm{GID}^{\beta_i}) \cdot g_1^{-\alpha_i}\right)^{\frac{1}{\beta_i}} \\ &= \mathrm{GID}.\end{aligned}$$

□

6.2.4 Security Definitions

Definition 18 (Secure authentication)**.** The authentication method is sound: a malicious user cannot authenticate with a GID other than his own, nor with an attribute he does not possess.

Definition 19 (Leak-freeness [GH07; HSM$^+$12])**.** The algorithm Blind Key Generation is *leak-free*, if for all efficient adversaries $\mathcal{A}$, there exists an efficient simulator $\mathcal{S}$, with special rewind capabilities, such that no efficient distinguisher $\mathcal{D}$ can distinguish whether $\mathcal{A}$ is executing *Real Experiment* or *Ideal Experiment* with non-negligible advantage, where

Real Experiment Run Global Setup and Authority Setup. As many times as the distinguisher $\mathcal{D}$ wants, the adversary $\mathcal{A}$ chooses a GID and executes the algorithm Blind Key Generation with authority A_j.

Ideal Experiment Run Global Setup and Authority Setup. As many times as the distinguisher $\mathcal{D}$ wants, the simulator $\mathcal{S}$ chooses a GID and queries a trusted party to obtain the output of the algorithm Key Generation if the submitted GID is valid, i.e., $\mathrm{GID} \in \mathbb{G}$, and $\perp$ otherwise.

The selective-failure blindness property of a Blind Key Generation protocol is defined as a security game. The adversary, that will play the role of an authority, may choose two users who will request a private key (SK). At the end of the game, the adversary has to match the issued SKs with the right user. He may consult two oracles so he can try to make an educated guess.

We use the symbol ε for the empty string in the following definition.

Definition 20 (Selective-failure blindness [CNs07; HSM$^+$12])**.** A Blind Key Generation protocol is said to be *selective-failure blind* if every adversary $\mathcal{A}$ has a negligible advantage in the following security game.

1. Global Setup is run, outputting the system parameters PK.
2. $\mathcal{A}$ executes the Authority Setup to obtain $\mathrm{MK}_{\mathcal{A}}$ and $\mathrm{PK}_{\mathcal{A}}$.
3. $\mathcal{A}$ publishes his $\mathrm{PK}_{\mathcal{A}}$ and chooses two valid GID attribute values ς_0, ς_1.
4. A bit $b \in_R \{0, 1\}$ is chosen.
5. $\mathcal{A}$ is given the commitment $X_{1,b}$ on ς_b and $X_{1,1-b}$ on ς_{1-b} and black-box access to two oracles: $U(\mathrm{PK}, \mathrm{PK}_{\mathcal{A}}, \varsigma_b)$ and $U(\mathrm{PK}, \mathrm{PK}_{\mathcal{A}}, \varsigma_{1-b})$.
   - The oracle U produces local output SK_b and SK_{1-b}, respectively. We may think of U as an honest user executing the Blind Key Generation protocol with $\mathcal{A}$ acting as KGA.

- If $\mathrm{SK}_b \neq \perp$ and $\mathrm{SK}_{1-b} \neq \perp$, then $\mathcal{A}$ is given the pair $(\mathrm{SK}_0, \mathrm{SK}_1)$[2]; if $\mathrm{SK}_b = \perp$ and $\mathrm{SK}_{1-b} \neq \perp$, then $\mathcal{A}$ is given $(\perp, \varepsilon)$; if $\mathrm{SK}_b \neq \perp$ and $\mathrm{SK}_{1-b} = \perp$, then $\mathcal{A}$ is given $(\varepsilon, \perp)$; if $\mathrm{SK}_b = \mathrm{SK}_{1-b} = \perp$, then $\mathcal{A}$ is given $(\perp, \perp)$.

6. Finally, $\mathcal{A}$ outputs his guess b' on b.

The advantage of $\mathcal{A}$ in this game is defined as $\left|\Pr[b' = b] - \frac{1}{2}\right|$.

6.3 Construction

Before we will look at how we can replace the Key Generation algorithm of the [LW11] scheme by a Blind Key Generation protocol, we will do so for a variant of the main scheme by Lewko and Waters. In their full paper [LW10], they describe a prime order construction, instead of the composite order construction that we described in Section 6.1. Although the [LW10] scheme is only proven secure using the generic group heuristic—instead of the static corruption model—the two constructions do not differ much: it suffices to notice only two differences in the construction. Firstly, the bilinear group is now of prime order p (instead of composite order $N = p_1p_2p_3$). Secondly, we use the generator g of the group $\mathbb{G}$ (instead of the generator g_1 of the subgroup $\mathbb{G}_{p_1}$). This implies that the public key of attribute i now equals the tuple $\left(h_i = e(g,g)^{\alpha_i}, u_i = g^{\beta_i}\right)$, and that the Key Generation algorithm creates keys of the form

$$\mathrm{SK}_{U,A_j} = g^{\alpha_i}\mathrm{GID}_U^{\beta_i}.$$

Note that this variant of the [LW11] scheme is much more similar to the [CZF11] and [RD13] schemes. These schemes use a bilinear group of prime order too and construct the SKs for their users in an identical way.

6.3.1 The Protocol

In order to create a provable Blind Key Generation protocol, we need to extend the Global Setup algorithm. An extra group element $h \in \mathbb{G} \setminus \{1\}$ needs to be determined by a TTP. Note that this h will be a generator too, because every element in $\mathbb{G}$ other than the identity is coprime to p.

Global Setup Select a bilinear group $\mathbb{G}$ of prime order p, with pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Pick a generator g of the bilinear group $\mathbb{G}$. Let $h \in_R \langle g \rangle \setminus \{1\}$ denote a random group element such that $\log_g h$ is secret to all other parties. The system parameters are $(\mathbb{G}, e, p, g, h)$.

Now that the modified Global Setup algorithm is presented, we will move on to the Blind Key Generation protocol. Suppose user U requests a decryption key for attribute t from key generation authority A_j, where $t \in \mathcal{S}_{A_j}$. Without loss of generality, assume $\mathrm{GID}_U = h^{\varsigma}$, where $\varsigma \in \mathbb{Z}_p$, and that the user possesses the root credential containing his card master secret key χ and his GID attribute value ς. In addition, assume that the user possesses a credential containing the attribute t. The user requests a decryption key at authority A_j.

[2] Notice that the ordering of SKs does *not* depend on the choice of b.

As described in Section 6.2.2, the Blind Key Generation protocol consists of three protocol steps. The first step is to blind the user's GID. This is done using the Pedersen commitment scheme. The user commits to his GID attribute value by computing $X = g^{\rho}h^{\varsigma}$ using a number $\rho \in_R \mathbb{Z}_p$. In order to prove to the KGA that the user possesses the attribute t and committed to his own ς, a zero-knowledge proof has to be constructed. The proof, that takes the form of a Σ-proof, consists of three parts. The first part assures that the commitment is well formed. The second part proves the possession of the attribute t, by using the regular ABC signature of knowledge. The last part of the proof is needed to link the two other parts together: it proves that the GID attribute ς used in the commitment is on the same IRMA card as the attribute t. This last part of the proof will thus assure that the decryption key is only issued to a user that is entitled to receive the attribute decryption key. As in the case of the idemix protocols, the user adds a nonce received from the KGA to assure freshness of the proof. The final Σ-proof will look like

$$\Sigma_1 = \mathrm{SPK}\big\{(\rho, \varsigma, \varepsilon_1, \varepsilon_2, \nu_1, \nu_2, \chi) : \; X = g^{\rho}h^{\varsigma} \wedge \\ Z \equiv A_1'^{\varepsilon_1} S^{\nu_1} R_0^{\chi} R_1^{t} \pmod{n} \wedge Z \equiv A_2'^{\varepsilon_2} S^{\nu_2} R_0^{\chi} R_1^{\varsigma} \pmod{n}\big\}(X, n_1).$$

This proof is sent to the KGA together with the commitment and a new fresh nonce n_2 generated by the user. The complete message sent to the KGA is the tuple (n_2, X, Σ_1).

The KGA verifies the proof and responds with $\perp$ if it fails. However, if the proof is correct, it computes $Y = g^{\alpha_t} X^{\beta_t}$ with its secret key (α_t, β_t) for attribute t. Just like the user, A_j also creates a Σ-proof that proves that the response is well-formed,

$$\Sigma_2 = \big\{(\alpha_t, \beta_t) : Y = g^{\alpha_t} X^{\beta_t}\big\}(Y, n_2).$$

The value Y together with the Σ-proof is sent to the user.

The user validates the proof and if it fails, he outputs $\perp$. Otherwise, he computes his decryption key $\mathrm{SK} = Y u_t^{-\rho}$ from the returned value Y and the public key part u_t of the attribute t.

The whole Blind Key Generation protocol is—in its simplest form—described in Figure 6.3. The complete construction of both the Σ-proofs is discussed in Appendix B.

The just described scenario is only for a specific example. However, the Blind Key Generation protocol does not require us to use this limited scenario. For example, another valid scenario may involve a KGA that requires the user to reveal two different attributes t_1 and t_2 (instead of one attribute t) before he is issued a decryption key. There are a few general requirements that a valid scenario must meet. Thus, the user must prove that he committed to the same ς of his smart card and that the revealed attributes (e.g., t_1 and t_2) belong to the same smart card, i.e., the ς and the attributes must come from a credential with the same χ.

6.4 Proof of Security

We will now prove that the Blind Key Generation protocol for the prime order construction of the [LW10] scheme provides secure authentication, is leak-free, and is selective-failure blind. But first, we will check if our protocol is complete.

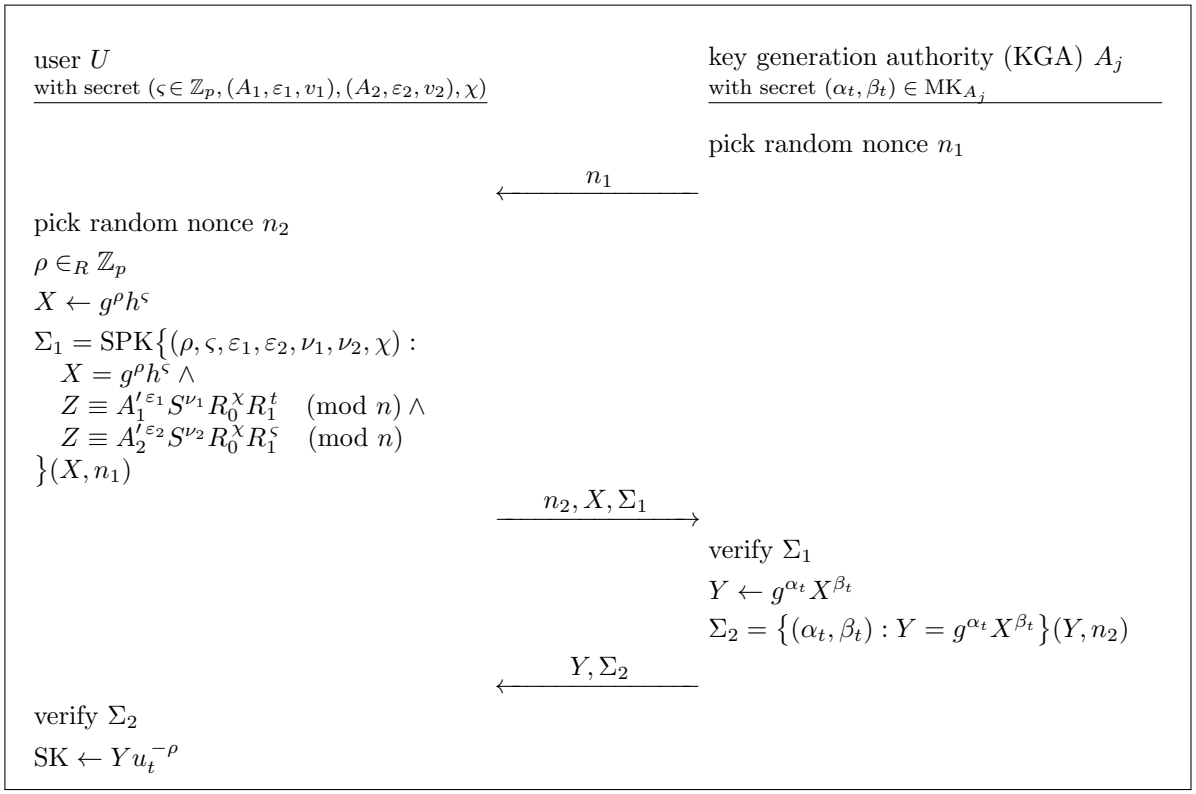

Figure 6.3: Schematic overview of the Blind Key Generation protocol.

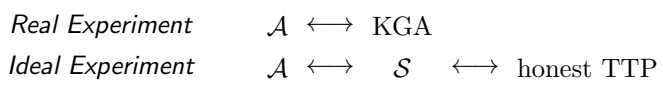

Figure 6.4: Schematic overview of the experiments in the leak-freeness definition.

Lemma 3 (Completeness)**.** *If an honest user U and an honest key generation authority A_j execute the protocol together, then U receives a correctly formed private key (SK).*

Proof.

$$\begin{aligned} \mathrm{SK} &= Y u_t^{-\rho} = Y(g^{\beta_t})^{-\rho} = g^{\alpha_t} X^{\beta_t} g^{-\rho\beta_t} \\ &= g^{\alpha_t} g^{\rho\beta_t} {\mathrm{GID}_U}^{\beta_t} g^{-\rho\beta_t} = g^{\alpha_t} {\mathrm{GID}_U}^{\beta_t} \end{aligned}$$ □

Lemma 4 (Information-theoretically hiding)**.** *The commitment information-theoretically hides the user's identity.*

Proof. The distribution of $X = g^{\rho} h^{\varsigma}$, for fixed ς and random ρ, is uniform in $\mathbb{G}$. In particular, X is statistically independent of the number ς that uniquely determines the user's GID. □

Theorem 1 (Secure authentication)**.** *The authentication in the Blind Key Generation protocol depicted in Figure 6.3 is secure.*

Proof. As long as the special soundness property of the constructed Σ-proof Σ_1 holds, the protocol securely authenticates. □

In order to prove the leak-freeness property, we will create a simulator $\mathcal{S}$ that simulates the new Blind Key Generation protocol, using the old Key Generation algorithm. Next, we will prove that a distinguisher cannot distinguish the simulated Key Generation algorithm from the real one.

Figure 6.4 schematically depicts the two experiments of the leak-freeness property. In the *Real Experiment*, the adversary $\mathcal{A}$ communicates using the Blind Key Generation protocol with the KGA A_j to obtain the decryption key. In the *Ideal Experiment*, the adversary $\mathcal{A}$ communicates using the Blind Key Generation protocol with the simulator $\mathcal{S}$. The simulator $\mathcal{S}$ in its turn, communicates using the Key Generation algorithm with the TTP to obtains the decryption key.

Theorem 2 (Leak-freeness)**.** *The Blind Key Generation protocol described in Figure 6.3 is leak-free. An efficient distinguisher $\mathcal{D}$ cannot distinguish $\mathcal{A}$ from running Real Experiment or Ideal Experiment.*

Proof. When $\mathcal{A}$ executes *Ideal Experiment*, then the simulator $\mathcal{S}$ behaves as follows.

1. The simulator $\mathcal{S}$ receives the public parameters (PK) from the TTP and sends it to $\mathcal{A}$.

2. $\mathcal{A}$ must send to $\mathcal{S}$ a value X and prove knowledge of values (ρ, ς) such that $X = g^{\rho} h^{\varsigma}$. If the proof fails to verify, $\mathcal{S}$ aborts. Since this proof of knowledge is implemented using the extractable techniques, $\mathcal{S}$ can efficiently extract the values (ρ, ς).

3. Next, $\mathcal{S}$ submits GID $= h^{\varsigma}$ to the trusted party, who returns the valid secret key for this identity $\mathrm{SK}_U = g^{\alpha_t} h^{\varsigma \beta_t}$.

4. Finally, $\mathcal{S}$ computes $Y' = \mathrm{SK}_U \cdot u_t^{\rho}$ and returns this value to $\mathcal{A}$.

Observe that the number Y' is exactly the same as the response Y from authority A_j in *Real Experiment*, as long as the trusted party returned the correct key SK_U. Thus, *Real Experiment* and *Ideal Experiment* are indistinguishable to both $\mathcal{A}$ and $\mathcal{D}$. Also note that the pair (ρ, ς) is efficiently extractable, by an extractor with special rewind capabilities not available to A_j, thus the simulator $\mathcal{S}$ is efficient. □

The selective-failure blindness proof basically consists of two parts. One part will prove that we do not gain any extra knowledge of the user's SK by consulting one or both of the oracles. And, because the oracles do not provide us with information on the user, any extra information on the user must come from other parts of the protocol. In the other part of the proof we will argue that neither the rest of the protocol leaks information about the user.

Theorem 3 (Selective-failure blindness)**.** *The Blind Key Generation protocol described in Figure 6.3 is selective-failure blind.*

Proof. Observe that, by executing the Blind Key Generation protocol, a user sends X and a signature of knowledge $\mathrm{SPK}\big\{(\rho, \varsigma, \varepsilon_1, \varepsilon_2, \nu_1, \nu_2, \chi) : X = g^{\rho} h^{\varsigma} \wedge Z \equiv A_1'^{\varepsilon_1} S^{\nu_1} R_0^{\chi} R_1^{t} \pmod{n} \wedge Z \equiv A_2'^{\varepsilon_2} S^{\nu_2} R_0^{\chi} R_1^{\varsigma} \pmod{n}\big\}(X, n_1)$. We know by Lemma 4 that X is distributed uniformly in $\mathbb{G}$.

Suppose that adversary $\mathcal{A}$ runs one or both of his oracles up to this point. We observe that, at this point, $\mathcal{A}$'s views on the two oracles are, computationally indistinguishable. Otherwise, Lemma 4 would be false or the witness indistinguishability property of the zero-knowledge proof will be broken. According to the Blind Key Generation protocol, $\mathcal{A}$ must now respond. Suppose $\mathcal{A}$, using any strategy he wishes, responds with the value $Y \in \mathbb{G}$. Now, $\mathcal{A}$ is able to predict the final output of oracle U, without interaction with the two oracles, with a non-negligible advantage as follows.

1. $\mathcal{A}$ creates the signature of knowledge $\mathrm{SPK}\{(\alpha, \beta) : Y = g^{\alpha} (X_{1,b})^{\beta}\}$. If the proof fails, $\mathcal{A}$ sets $\mathrm{SK}_0 = \perp$, otherwise he computes SK_0 using Key Generation with ς_0.

2. Next, $\mathcal{A}$ generates a different signature of knowledge Y', $\mathrm{SPK}\{(\alpha, \beta) : Y' = g^{\alpha} (X_{1,1-b})^{\beta}\}$. If this proof fails, $\mathcal{A}$ sets $\mathrm{SK}_1 = \perp$, otherwise he computes SK_1 using Key Generation with ς_1.

3. Finally, $\mathcal{A}$ returns his prediction on what the oracle will output:

   - $(\mathrm{SK}_0, \mathrm{SK}_1)$ if $\mathrm{SK}_0 \neq \perp$ and $\mathrm{SK}_1 \neq \perp$[3];

[3]Notice that SK_0 and SK_1 are in the right order again. SK_0 and SK_1 correspond to the GID attribute value ς_0 and ς_1, respectively.

- $(\perp, \varepsilon)$ if $\text{SK}_0 = \perp$ and $\text{SK}_1 \neq \perp$;
- $(\varepsilon, \perp)$ if $\text{SK}_0 \neq \perp$ and $\text{SK}_1 = \perp$;
- $(\perp, \perp)$ if $\text{SK}_0 = \text{SK}_1 = \perp$.

This prediction is correct, because $\mathcal{A}$ is performing the same check as the honest user U. Because $\mathcal{A}$ is able to predict the output of the oracles correctly, his advantage in this security game is the same as the game without this final output of the black-box access to U. Thus, all of the advantage of $\mathcal{A}$ must come from distinguishing the earlier messages of the oracles. However, the oracles only send one uniformly random value $X \in \mathbb{G}$ and a Σ-proof; we know from the security of the underlying proof that $\mathcal{A}$ cannot distinguish between them with non-negligible probability. □

6.5 The Composite Order Scheme

Now that we have created a Blind Key Generation protocol for the prime order constructions of the [LW10], [CZF11], and [RD13] schemes, we will concentrate on creating a similar privacy-friendly key issuance protocol for the [LW11] scheme. Because the [LW11] scheme uses a generator of the subgroup $\mathbb{G}_{p_1}$ and a GID of $\mathbb{G}$—instead of using both the generator and the GID from the *same* group or subgroup as the other schemes do—we can not simply reuse our protocol and apply it to the [LW11] scheme.

A direct consequence of using the composite order construction, is that we have to select $\varsigma, \rho \in_R \mathbb{Z}_\Gamma$ instead of $\mathbb{Z}_p$, simply because Γ—instead of p—is now the order of the bilinear group. Note that when we have to select a random element out of the bilinear subgroup of order p_1, we compute g_1^ρ using the generator g_1 of the subgroup $\mathbb{G}_{p_1}$. This is a bit inefficient because the random number ρ is drawn from the set $\mathbb{Z}_\Gamma$ with cardinality $p_1p_2p_3$, which is significantly larger than the order of the generator g_1, $|g_1| = p_1$, which is secret.

We take two approaches in creating a privacy-friendly key issuance protocol for the [LW11] scheme. In our first approach we modify our Blind Key Generation protocol. The second approach modifies the [LW11] scheme. We conclude with an overview of the advantages and disadvantages of both approaches.

6.5.1 Adapting the Protocol

Since the GID is an element of the bilinear group $\mathbb{G}$ and the generator g_1 generates only the subgroup $\mathbb{G}_{p_1}$, problems with the security proof of the Blind Key Generation protocol arise. The selective-failure blindness property of our original protocol relies on the fact that the value $X = g^\rho h^\varsigma$ does not leak information on the user's GID. We used Pedersen's commitment scheme to construct the commitment X. However, in our modified protocol, we cannot use the generator g anymore, since this would result in an unsound protocol. So, we change the commitment to $X = g_1^\rho h^\varsigma$, where we have used the generator g_1 of the subgroup $\mathbb{G}_{p_1}$. This can be rewritten as

$$X = g_1^\rho h^\varsigma = h^{x\rho} h^\varsigma = h^{x\rho+\varsigma} \quad \text{where } x = \log_h g_1 \text{ is unknown,}$$

because h is a generator of the entire group. For fixed ς and x, and random ρ, X is an element of the subset of $\mathbb{G}$ of cardinality $|\mathbb{G}_{p_1}| = p_1$, because g_1 generates

a subgroup of order p_1. The element X is hidden in the subset determined by ρ: choosing another $\rho \in \mathbb{Z}_{p_1}$ results in another element of the same subset. The subset may be different for other ς; there are $\frac{\Gamma}{p_1} = p_2 p_3$ distinct subsets in total.

The information-theoretical hiding property does not hold any more for this commitment scheme. For a fixed ς, the commitment X is an element in some subset of the group that varies for different values of ς. Assuming two elements are given, it is impossible to (computationally) determine whether the first element is in the same subset as the second element, the modified protocol still (computationally) hides the user's ς. This is the same complexity assumption as listed in Definition 16.

6.5.2 Adapting the Scheme

Instead of trying to modify our Blind Key Generation scheme so that the protocol can be applied to the [LW11] scheme, we could also slightly modify the [LW11] scheme so that it fits our protocol. Our original protocol could be directly applied to the scheme if it were to use a generator of the same group (or subgroup) as the group (respectively, subgroup) of which the GID is from. However, the use of a generator $g_1 \in \mathbb{G}_{p_1}$, that generates the bilinear subgroup of prime order p_1, is a key property of the security proof of the scheme. If we choose a generator of another subgroup, or a generator of the composite order group, a completely new security proof would be needed.

On the other hand, we could pick *the GID* as an element out of the subgroup $\mathbb{G}_{p_1}$ instead of the group $\mathbb{G}$. An important question to ask is, if choosing the GID $\in \mathbb{G}_{p_1}$ undermines the security of the scheme. Luckily, this is not the case. Quite the contrary, Lewko and Waters [LW10] use a GID $\in \mathbb{G}_{p_1}$ in their security proofs of their scheme, but prove that a GID $\in \mathbb{G}$ may be used too. We omit the proof of Lemma 5 here; the interested reader may want to look at the proof provided by [LW10].

Lemma 5 (From [LW10, Lemma 7])**.** *Suppose there exists a polynomial time algorithm $\mathcal{A}$ such that it can distinguish the security game using GID $\in \mathbb{G}_{p_1}$ from the security game using GID $\in \mathbb{G}$ with advantage ε. Then we can construct a polynomial time algorithm $\mathcal{B}$ with advantage ε in breaking the subgroup decision problem for 3 primes, as defined in Definition 12.*

A direct consequence of choosing a GID $\in \mathbb{G}_{p_1}$ is that the issued private key is now an element of the subgroup. That could easily be seen by recalling that $\mathrm{SK} = g_1^{\alpha_t} \mathrm{GID}^{\beta_t}$, where g_1 is a generator of the subgroup $\mathbb{G}_{p_1}$ and GID is now an element of the same subgroup. Thus, using this approach significantly reduces the available key space. The key space reduces by a factor three in this modified scheme. So, to obtaining the same level of security, we need to pick the bit size of the three primes p_1, p_2, and p_3 three times as large as original.

Another consequence is that the GIDs are not necessarily unique anymore even though the values ς *are*. This is due the fact that $g_1^{\varsigma} = g_1^{\varsigma \bmod p_1}$, but $\varsigma \in_R \mathbb{Z}_\Gamma$, $\Gamma = p_1 p_2 p_3$. So, for each ς there are $\frac{\Gamma}{p_1} - 1 = p_2 p_3 - 1$ different ς that have the same GID.

6.5.3 Comparison of Approaches

When we decide to adapt the protocol, we have to introduce a new complexity

assumption. Although this assumption seems reasonable, adding a new assumption might be considered problematic. If we adapt the scheme instead, no new complexity assumptions are required. However, this comes at a great efficiency cost: a much larger key space is required to obtain the same level of security. Moreover, the GID is no longer guaranteed to be unique. Choosing a larger security parameter increases the key space and reduces the chance that several users possess the same GID.

Chapter 7

Practical MA-ABE for IRMA

We would like to be able to decrypt messages using our IRMA card. However, current smart cards are not powerful enough to quickly compute complex operations like the many bilinear pairings needed for decrypting a ciphertext. Though, if we allow communication between the smart card and a trusted device, e.g., a tablet computer or smart phone, we could outsource the complex computations to that trusted device. For security reasons, the trusted device should not learn the private key SK of the user, but only do some auxiliary computations or computations with a randomized key. As the device may still be able to learn part of the user's secret or message, we require the device to be trusted to some extent.

7.1 Off-Card Decryption

The prime order scheme as well as the composite order scheme of [LW11] can be modified to allow a trusted device to partially decrypt a ciphertext with a randomized decryption key obtained from the smart card. The trusted device is not given the private key of the user, but instead a randomized instance of the key. We create a new decryption protocol, consisting of three parts. First, the smart card does some small calculations to randomize its key and the ciphertext to obtain SK' and CT', respectively. The trusted device is given the SK' and CT'. Next, the trusted device executes the original Decrypt algorithm using the randomized ciphertext CT' and the randomized key SK'. This results in a similarly randomized plaintext M'. We rely on $|\mathbb{G}| = |\mathbb{G}_T|$ for the prime order construction [LW10, Appendix E], as well as for the composite order construction [BGN05] to randomize and derandomize the ciphertext and randomized plaintext.

We will now describe the Off-Card Decrypt protocol for the composite order construction where $|\mathbb{G}| = |\mathbb{G}_T| = \Gamma$.

Off-Card Decrypt The smart card establishes a secure channel with the trusted device and picks an $r \in_R \mathbb{Z}_\Gamma^*$ such that r is coprime to Γ. The card

calculates the randomized values

$$\mathrm{CT}' = \Big(C_0' = C_0^r, \\ \forall x \in \tilde{\mathcal{S}} : \ \tilde{\mathrm{SK}}'_{U,\rho(x)} = \tilde{\mathrm{SK}}^r_{U,\rho(x)}, C'_{1,x} = C^r_{1,x}, C_{2,x}, C'_{3,x} = C^r_{3,x}\Big)$$

from his private key and the original ciphertext and sends this over the secure channel to the trusted device. Recall that the set $\tilde{\mathcal{S}}$ denotes a minimal set of attributes that the user possesses to satisfy the access structure.

The trusted device follows the 'normal' decryption steps to calculate

$$\begin{aligned}
\tilde{C}'_x &= \frac{C'_{1,x} \cdot e(\mathrm{GID}_U, C'_{3,x})}{e(\tilde{\mathrm{SK}}'_{U,\rho(x)}, C_{2,x})} \\
&= \frac{e(g_1,g_1)^{\nu_x r} h_{\rho(x)}^{r_x r} \cdot e(\mathrm{GID}_U, u_{\rho(x)}^{r_x r} g_1^{\omega_x r})}{e(g_1^{\alpha_{\rho(x)} r} \mathrm{GID}_U^{\beta_{\rho(x)} r}, g_1^{r_x})} \\
&= \frac{e(g_1,g_1)^{\nu_x r + \alpha_{\rho(x)} r_x r} \cdot e(\mathrm{GID}_U, g_1^{\beta_{\rho(x)} r_x r + \omega_x r})}{e(g_1,g_1)^{\alpha_{\rho(x)} r r_x} e(\mathrm{GID}_U, g_1)^{\beta_{\rho(x)} r r_x}} \\
&= e(g_1,g_1)^{\nu_x r} e(\mathrm{GID}_U, g_1)^{\omega_x r} \\
&= \tilde{C}_x^r
\end{aligned}$$

for all x it receives. Next, using constants $c_x \in \mathbb{Z}_\Gamma$ such that they satisfy $\sum_x c_x A_x = \begin{pmatrix}1 & 0 & \cdots & 0\end{pmatrix}$, it computes

$$\begin{aligned}
\tilde{C}' &= \prod_{x \in \tilde{\mathcal{S}}} \tilde{C}_x'^{c_x} \\
&= e(g_1,g_1)^{sr}
\end{aligned}
\qquad \text{and} \qquad
\begin{aligned}
M' &= C'_0 \cdot \tilde{C}'^{-1} \\
&= M^r e(g_1,g_1)^{sr} e(g_1,g_1)^{-sr} \\
&= M^r.
\end{aligned}$$

Finally, the trusted device sends the value M' to the smart card. The smart card can now extract the message using $d = r^{-1} \mod \Gamma$,

$$\begin{aligned}
M &= (M')^d \\
&= (M^r)^d .
\end{aligned}$$

Note that d can be calculated using the *extended Euclidean algorithm* since r is coprime to Γ. The extended Euclidean algorithm is an efficient algorithm to determine the modular inverse of a given number.

7.2 Efficiency

Using the Off-Card Decrypt protocol, the smart card does not have to compute any bilinear pairing. This is a significant improvement since the [LW11] scheme requires 2 bilinear pairings and one exponentiation per attribute that the decryptor uses to decrypt the ciphertext. Next, we express the computational costs for our off-card decryption protocol using the notation introduced on Page 44. The smart card has to compute $|\omega_d|(2E_{\mathbb{G}} + E_{\mathbb{G}_T}) + E_{\mathbb{G}_T}$ for the randomization and $E_{\mathbb{G}_T} + I_{\mathbb{Z}_\Gamma}$ for the final message extraction. The computational cost for the

trusted device are exactly the same as the total decryption cost of the original Decrypt algorithm, $|\omega_d|(2P + E_{\mathbb{G}_T} + I_{\mathbb{G}_T} + 3M_{\mathbb{G}_T}) + I_{\mathbb{G}_T}$. We stress the fact that *the smart card* does not have to compute any bilinear pairings. For example, if a user has two attributes that jointly satisfy the access structure, then the smart card has to compute 8 exponentiations.

7.3 Trusting the Device

We emphasize that the device should be trusted by the user. The security of the Off-Card Decrypt protocol relies on the randomization of the ciphertext, decryption key, and the resulting plaintext. These values are randomized by raising them to the exponent r that is picked uniformly from $\mathbb{Z}_\Gamma$. Since the ciphertext, decryption key, and plaintext are not necessarily generators of the entire group, they are not fully randomized: an element g out of a group $\mathbb{G}$ is fully randomized by g^r, with $r \in_R \mathbb{Z}_{|\mathbb{G}|}$, if and only if g is a generator of the *entire* group $\mathbb{G}$. The lack of full randomization implies that confidential information could leak.

Note that this is only an issue for the composite order construction, as in the prime order construction every element is a generator of the entire group.

7.3.1 Chosen Ciphertext Attack

The described Off-Card Decrypt protocol is not secure against a chosen ciphertext attack (CCA). To illustrate this, assume that the 'trusted device' cannot be trusted and is indeed a malicious party. Instead of returning the randomized message M' to the smart card, it could return its received randomized user's decryption key $\tilde{\mathrm{SK}}'_{U,i}$ for attribute i. The final computation by the smart card would now return the smart card's own secret key for attribute i $\tilde{\mathrm{SK}}_{U,i}$,

$$\tilde{\mathrm{SK}}'^{d}_{U,i} = \left(\tilde{\mathrm{SK}}_{U,i}{}^{r}\right)^{d} = \tilde{\mathrm{SK}}_{U,i}.$$

The attack can be mitigated by letting the card never return the final plaintext. Instead of returning the plaintext M to the device, the derandomized message M could be used as a secret key for secure symmetric cipher. In this case a message m is encrypted with a symmetric cipher using a random secret key $k_{\text{symmetric}}$ to obtain $c_{\text{symmetric}}$. The random secret key is then encrypted using the Encrypt algorithm from the ABE scheme to obtain c_{ABE}. Decryption of the message works by first recovering the random secret key using the Off-Card Decrypt protocol and then use the result to decipher $c_{\text{symmetric}}$ and finally retrieve m. Now, if a faulty key $k'_{\text{symmetric}}$ is used in the symmetric cipher (due to the device sending a wrong c_{ABE}), pseudorandom data m' will be returned and this m' will not reveal anything about the used key $k'_{\text{symmetric}}$.

Chapter 8

Conclusions and Recommendations

8.1 Conclusions

The Lewko and Waters [LW11] scheme is one of the first practical MA-ABE schemes. Due to its decentralized setup, it fits well in the IRMA ecosystem. Another advantage of the scheme is that it is proven secure in the standard model using realistic assumptions. Although the scheme is one of the best candidates to be used in the IRMA ecosystem, it still suffers from two major drawbacks. The key issuance algorithm is not privacy-friendly and the computations required by the decryption algorithm are too complex for a smart card. However, we have overcome both issues in this thesis.

The privacy-friendly key issuance is made slightly more complicated due to the fact that the scheme uses a generator that generates only a subgroup instead of the complete group. To assure that our solution does not leak information of the used GID, we explored two separate approaches: modifying the protocol or modifying the ABE scheme. Modifying the protocol requires us to introduce an extra complexity assumption. Using this method, the issued decryption key is exactly the same as the one that is issued by the key issue algorithm of the scheme. Modifying the ABE scheme has other advantages and disadvantages. The modified scheme is still provably secure, although the key space significantly reduces. In order to obtain a key space of the same size, much larger primes have to be chosen. This reduces the efficiency of the scheme.

We presented Off-card Decrypt protocol as a solution to decrease the computational complexity to decrypt a ciphertext on the smart card. Although the total amount of computation does not decrease—it slightly increases—the computational complexity *for the smart card* significantly reduces, due to the outsourcing of heavy computations to a more powerful device.

The resulting scheme with our Blind Key Generation and Off-card Decrypt protocols is a practical and privacy-friendly scheme which fits well in the IRMA ecosystem.

8.2 Further Research

We have focused on modifying the [LW11] scheme to suit the IRMA ecosystem. Although the [LW11] scheme is not the most efficient scheme (e.g., [CZF11; RD13] are more efficient), it has a solid security proof and is a practical encryption solution for the IRMA project. Further research could focus on new schemes that are more efficient, yet have the same solid security proof and are suitable to run on a smart card.

Our Blind Key Generation protocol can be applied to several other schemes that use a decryption key of the same form. However, new MA-ABE schemes could construct a totally different looking SK. In such a case, a new Blind Key Generation protocol must be constructed, assuming that the new MA-ABE scheme still requires the user to reveal his GID. An ambitious research project can try to generalize the different types of SKs and create a more general Blind Key Generation protocol that can be applied to various different schemes.

The Off-card Decrypt protocol is not proven secure against various types of attacks. The unmodified version of the protocol is even known to be insecure against a chosen ciphertext attack. A security game and proof should be created before the protocol is put into practice.

Appendix A

Additional Complexity Assumptions

We list here some additional complexity assumptions that occur in Table 5.2. The reader is referred to the paper of Bethencourt, Sahai, and Waters [BSW07] and the full paper of Boneh, Boyen, and Goh [BBG05a] for the generic group heuristic (the generic bilinear group model). Note that the generic group is *not* a complexity assumptions.

Definition 21 (Decisional Modified Bilinear Diffie-Hellman [SW05])**.** Given a generator g of the bilinear group $\mathbb{G}$ of prime order p and three arbitrary group elements $A = g^a$, $B = g^b$, and $C = g^c$, it is hard to distinguish $e(g,g)^{\frac{ab}{c}}$ from a random group element $Z = e(g,g)^z$.

More formally: let $\mathcal{G} = (p, \mathbb{G}, \mathbb{G}_T, e, A, B, C)$; the advantage of an adversary $\mathcal{A}$ in distinguishing $e(g,g)^{\frac{ab}{c}}$ from Z,

$$\left|\Pr[\mathcal{A}(\mathcal{G}, e(g,g)^{\frac{ab}{c}}) = 1] - \Pr[\mathcal{A}(\mathcal{G}, Z) = 1]\right|,$$

is negligible.

Definition 22 (q-Decisional Diffie–Hellman Inversion assumption [CC09])**.** Given a generator g of the bilinear group $\mathbb{G}$ of prime order p and q group elements $y_i = g^{\alpha^i}$ for $i \in \{1, \ldots, q\}$, it is hard to distinguish $g^{\frac{1}{z}}$ from a random group element $Z = g^z$.

More formally: let $\mathcal{G} = (p, \mathbb{G}, \mathbb{G}_T, e, g, y_1, \ldots, y_q)$; the advantage of an adversary $\mathcal{A}$ in distinguishing $g^{\frac{1}{z}}$ from Z,

$$\left|\Pr[\mathcal{A}(\mathcal{G}, g^{\frac{1}{z}}) = 1] - \Pr[\mathcal{A}(\mathcal{G}, Z) = 1]\right|,$$

is negligible.

Definition 23 (n-Bilinear Diffie–Hellman Exponent assumption [BBG05b])**.** Given two generators g and h of the bilinear group $\mathbb{G}$ of prime order p and $2n-1$ group elements $y_i = g^{\alpha^i}$ for $i \in \{1, \ldots, n-1, n+1, \ldots, 2n\}$, it is hard to compute $e(g,h)^{\alpha^n}$.

More formally: let $\mathcal{G} = (p, \mathbb{G}, \mathbb{G}_T, e, g, h, y_1, \ldots, y_{n-1}, y_{n+1}, \ldots, y_{2n})$; the advantage of an adversary $\mathcal{A}$ in finding value x,

$$\Pr\left[x \leftarrow \mathcal{A}(\mathcal{G}) : x = e(g,h)^{\alpha^n}\right],$$

is negligible.

Definition 24 (Decisional n-Bilinear Diffie–Hellman Exponent assumption [BBG05b])**.** Given two generators g and h of the bilinear group $\mathbb{G}$ of prime order p and $2n-1$ group elements $y_i = g^{\alpha^i}$ for $i \in \{1, \ldots, n-1, n+1, \ldots, 2n\}$, it is hard to distinguish $e(g,h)^{\alpha^n}$ from a random group element $Z = e(g,g)^z$.

More formally: let $\mathcal{G} = (p, \mathbb{G}, \mathbb{G}_T, e, g, h, y_1, \ldots, y_{n-1}, y_{n+1}, \ldots, y_{2n})$; the advantage of an adversary $\mathcal{A}$ in distinguishing $e(g,h)^{\alpha^n}$ from Z,

$$\left|\Pr\left[\mathcal{A}\left(\mathcal{G}, e(g,h)^{\alpha^n}\right) = 1\right] - \Pr[\mathcal{A}(\mathcal{G}, Z) = 1]\right|,$$

is negligible.

Definition 25 (First q-type assumption (q-1) [RW13])**.** Let $a, b, c_1, \ldots, c_q \in_R \mathbb{Z}_p$. Given a generator g of the bilinear group $\mathbb{G}$ of prime order p and the following group elements T:

- g^b;
- $g^{a^i}, g^{c_j}, g^{bc_j}, g^{a^i c_j}, g^{\frac{a^i}{c_j^2}} \qquad \forall (i,j) \in \{1, \ldots, q\} \times \{1, \ldots, q\}$;
- $g^{\frac{a^i c_j}{c_k^2}} \qquad \forall (i,j,k) \in \{1, \ldots, 2q\} \times \{1, \ldots, q\} \times \{1, \ldots, q\}$ with $j \neq k$;
- $g^{\frac{a^i}{c_j}} \qquad \forall (i,j) \in \{1, \ldots, 2q\} \times \{1, \ldots, q\}$ with $i \neq q+1$;
- $g^{\frac{a^i b c_j}{c_k}}, g^{\frac{a^i b c_j}{c_k^2}} \qquad \forall (i,j,k) \in \{1, \ldots, q\} \times \{1, \ldots, q\} \times \{1, \ldots, q\}$ with $j \neq k$,

it is hard to distinguish $e(g,g)^{a^{q+1}b}$ from a random group element $Z = e(g,g)^z$.

More formally: let $\mathcal{G} = (p, \mathbb{G}, \mathbb{G}_T, e, g, T)$; the advantage of an adversary $\mathcal{A}$ in distinguishing $e(g,g)^{a^{q+1}b}$ from Z,

$$\left|\Pr\left[\mathcal{A}\left(\mathcal{G}, e(g,g)^{a^{q+1}b}\right) = 1\right] - \Pr[\mathcal{A}(\mathcal{G}, Z) = 1]\right|,$$

is negligible.

Definition 26 (Second q-type assumption (q-2) [RW13])**.** Let $a, b, c, y_1, \ldots, y_q \in_R \mathbb{Z}_p$. Given a generator g of the bilinear group $\mathbb{G}$ of prime order p and the following group elements T:

- $g^a, g^b, g^c, g^{(ac)^2}$;
- $g^{y_i}, g^{acy_i}, g^{\frac{ac}{y_i}}, g^{a^2 c y_i}, g^{\frac{b}{y_i^2}}, g^{\frac{b^2}{y_i^2}} \qquad \forall i \in \{1, \ldots, q\}$;
- $g^{\frac{acy_i}{y_j}}, g^{\frac{by_i}{y_j^2}}, g^{\frac{abcy_i}{y_j}}, g^{\frac{(ac)^2 y_i}{y_j}} \qquad \forall (i,j) \in \{1, \ldots, q\} \times \{1, \ldots, q\}$ with $i \neq j$,

it is hard to distinguish $e(g,g)^{abc}$ from a random group element $Z = e(g,g)^z$.

More formally: let $\mathcal{G} = (p, \mathbb{G}, \mathbb{G}_T, e, g, T)$; the advantage of an adversary $\mathcal{A}$ in distinguishing $e(g,g)^{abc}$ from Z,

$$\left|\Pr\left[\mathcal{A}\left(\mathcal{G}, e(g,g)^{abc}\right) = 1\right] - \Pr[\mathcal{A}(\mathcal{G}, Z) = 1]\right|,$$

is negligible.

Appendix B

Construction of the Σ-Proofs

In this chapter we present the complete construction of both Σ-protocols used in the Blind Key Generation protocol from Section 6.3. The Σ-protocols can be turned into a Σ-proof using the Fiat–Shamir heuristic described in Section 2.3.1.

B.1 Notations

We introduce additional notations to improve the readability of the construction. We use the notation $\{0,1\}^\ell$ to represent the set of integers $\{0,\ldots,2^\ell-1\}$. With $\pm\{0,1\}^\ell$ we represent the set $\{-2^\ell+1,\ldots,2^\ell-1\}$.

Additionally, we define several security parameters, similar to the idemix specification [IBM13]. Table B.1 list the used symbols, their meaning, and recommended bit size by [IBM13].

The ς attribute value should be smaller than or equal to the normal attribute size ℓ_m, otherwise ς would not fit in an attribute value.

symbol	usage ("the size for")	bit size
ℓ_n	RSA modulus	2048
$\ell_\varnothing$	security parameter that governs the statistical zero-knowledge property	80
ℓ_e	e value of the signature	597
ℓ'_e	interval where the e values are taken from	120
ℓ_m	attributes	256
ℓ_v	v value of the signature	2724
$\ell_\mathcal{H}$	domain of the hash function $\mathcal{H}$ used for the Fiat–Shamir heuristic	256

Table B.1: Symbols used in the idemix proofs.

B.2 Σ-Protocol for Σ_1

The complete zero-knowledge proof

$$\mathrm{PK}\big\{(\rho,\varsigma,\varepsilon_1,\varepsilon_2,\nu_1,\nu_2,\chi):\ X=g^{\rho}h^{\varsigma}\wedge Z\equiv {A'_1}^{\varepsilon_1}S^{\nu_1}R_0^{\chi}R_1^{t}\pmod n\wedge Z\equiv {A'_2}^{\varepsilon_2}S^{\nu_2}R_0^{\chi}R_1^{\varsigma}\pmod n\big\}(X,n_1).$$

is described in Figure B.1.

Lemma 6. *The protocol depicted in Figure B.1 is a Σ-protocol.*

Proof. We will prove the three properties that make this protocol a Σ-protocol.

Completeness The protocol is complete:

$$\begin{aligned}
g^{s_\rho}h^{s_\varsigma} &= g^{r_\rho+c\rho}h^{r_\varsigma+c\varsigma} = g^{r_\rho}h^{r_\varsigma}(g^{\rho}h^{\varsigma})^{c} = t_X X^{c};\\
(A'_1)^{s_{\varepsilon_1}}(R_0^{s_\chi})(S^{s_{\nu_1}}) &\equiv (A'_1)^{r_{\varepsilon_1}+c(\varepsilon_1-2^{\ell_e-1})}(R_0^{r_\chi+c\chi})(S^{r_{\nu_1}+c\nu_1})\\
&\equiv \tilde{Z}_1\left[(A'_1)^{\varepsilon_1}(R_0^{\chi})(S^{\nu_1})(A'_1)^{-2^{\ell_e-1}}\right]^{c}\\
&\equiv \tilde{Z}_1\left[Z(A'_1)^{-2^{\ell_e-1}}(R_1^{-t})\right]^{c}\pmod n;\\
(A'_2)^{s_{\varepsilon_2}}(R_0^{s_\chi}R_1^{s_\varsigma})(S^{s_{\nu_2}}) &\equiv (A'_2)^{r_{\varepsilon_2}+c(\varepsilon_2-2^{\ell_e-1})}(R_0^{r_\chi+c\chi}R_1^{r_\varsigma+c\varsigma})(S^{r_{\nu_2}+c\nu_2})\\
&\equiv \tilde{Z}_2\left[(A'_2)^{\varepsilon_2}(R_0^{\chi}R_1^{\varsigma})(S^{\nu_2})(A'_2)^{-2^{\ell_e-1}}\right]^{c}\\
&\equiv \tilde{Z}_2\left[Z(A'_2)^{-2^{\ell_e-1}}\right]^{c}\pmod n.
\end{aligned}$$

The numbers s_{ε_1} and s_{ε_2} lie in the interval

$$\begin{aligned}
&\pm\{0,1\}^{\ell'_e+\ell_\varnothing+\ell_{\mathcal{H}}}+\{0,1\}^{\ell_{\mathcal{H}}}\cdot([2^{\ell_e-1},2^{\ell_e-1+\ell'_e-1}]-2^{\ell_e-1})\\
&\qquad=[-2^{\ell'_e+\ell_\varnothing+\ell_{\mathcal{H}}}+1,2^{\ell'_e+\ell_\varnothing+\ell_{\mathcal{H}}}+(2^{\ell_{\mathcal{H}}}-1)(2^{\ell'_e-1}-1)]\\
&\qquad\subset\pm\{0,1\}^{\ell'_e+\ell_\varnothing+\ell_{\mathcal{H}}+1}.
\end{aligned}$$

The numbers s_χ and s_ς lie in the interval $\pm\{0,1\}^{\ell_m+\ell_\varnothing+\ell_{\mathcal{H}}}+\{0,1\}^{\ell_{\mathcal{H}}}\cdot\{0,1\}^{\ell_m}\subset\pm\{0,1\}^{\ell_m+\ell_\varnothing+\ell_{\mathcal{H}}+1}$.

Special soundness Given two accepting conversations

$$((t_X,\tilde{Z}_1,\tilde{Z}_2),c,(s_{\varepsilon_1},s_{\varepsilon_2},s_{\nu_1},s_{\nu_2},s_\chi,s_\varsigma))$$

and

$$((t_X,\tilde{Z}_1,\tilde{Z}_2),c',(s'_{\varepsilon_1},s'_{\varepsilon_2},s'_{\nu_1},s'_{\nu_2},s'_\chi,s'_\varsigma))$$

with $c\neq c'$, we can extract all witnesses ρ, ς, ε_1, ε_2, ν_1, ν_2, χ. From the verification equations we have

$$\begin{aligned}
g^{s_\rho-s'_\rho}h^{s_\varsigma-s'_\varsigma} &= X^{c-c'},\\
(A'_1)^{s_{\varepsilon_1}-s'_{\varepsilon_1}}(R_0^{s_\chi-s'_\chi})(S^{s_{\nu_1}-s'_{\nu_1}}) &\equiv\left(Z(A'_1)^{-2^{\ell_e-1}}(R_1^{-t})\right)^{c-c'}\pmod n,\\
(A'_2)^{s_{\varepsilon_2}-s'_{\varepsilon_2}}(R_0^{s_\chi-s'_\chi}R_1^{s_\varsigma-s'_\varsigma})(S^{s_{\nu_2}-s'_{\nu_2}}) &\equiv\left(Z(A'_2)^{-2^{\ell_e-1}}\right)^{c-c'}\pmod n,
\end{aligned}$$

$$(s_{\varepsilon_1}-s'_{\varepsilon_1}),(s_{\varepsilon_2}-s'_{\varepsilon_2})\in\pm\{0,1\}^{\ell'_e+\ell_\varnothing+\ell_{\mathcal{H}}+2},\text{ and}$$
$$(s_\chi-s'_\chi),(s_\varsigma-s'_\varsigma)\in\pm\{0,1\}^{\ell_m+\ell_\varnothing+\ell_{\mathcal{H}}+2}.$$

user U (prover)
with secret $((A_1, \varepsilon_1, v_1), (A_2, \varepsilon_2, v_2), \rho, \chi, \varsigma)$

key generation authority A_j (verifier)
verifies if the user authenticates securely

$r_{A_1}, r_{A_2} \in_R \{0,1\}^{\ell_n + \ell_\varnothing}$
$A'_1 \leftarrow A_1 S^{r_{A_1}} \mod n$
$A'_2 \leftarrow A_2 S^{r_{A_2}} \mod n$
$\nu_1 \leftarrow v_1 - \varepsilon_1 r_{A_1}$
$\nu_2 \leftarrow v_2 - \varepsilon_2 r_{A_2}$
$r_\rho \in_R \mathbb{Z}_\Gamma$
$r_{\varepsilon_1}, r_{\varepsilon_2} \in_R \pm\{0,1\}^{\ell'_e + \ell_\varnothing + \ell_\mathcal{H}}$
$r_{\nu_1}, r_{\nu_2} \in_R \pm\{0,1\}^{\ell_v + \ell_\varnothing + \ell_\mathcal{H}}$
$r_\chi, r_\varsigma \in_R \pm\{0,1\}^{\ell_m + \ell_\varnothing + \ell_\mathcal{H}}$
$t_X \leftarrow g^{r_\rho} h^{r_\varsigma}$
$\tilde{Z}_1 \leftarrow (A'_1)^{r_{\varepsilon_1}} (R_0^{r_\chi})(S^{r_{\nu_1}}) \mod n$
$\tilde{Z}_2 \leftarrow (A'_2)^{r_{\varepsilon_2}} (R_0^{r_\chi} R_1^{r_\varsigma})(S^{r_{\nu_2}}) \mod n$

$\xrightarrow{A'_1, A'_2, t_X, \tilde{Z}_1, \tilde{Z}_2}$

$c \in_R \{0,1\}^{\ell_\mathcal{H}}$

$\xleftarrow{c}$

$s_\rho \leftarrow r_\rho + c\rho \mod \Gamma$
$s_{\varepsilon_1} \leftarrow r_{\varepsilon_1} + c(\varepsilon_1 - 2^{\ell_e - 1})$
$s_{\varepsilon_2} \leftarrow r_{\varepsilon_2} + c(\varepsilon_2 - 2^{\ell_e - 1})$
$s_{\nu_1} \leftarrow r_{\nu_1} + c\nu_1$
$s_{\nu_2} \leftarrow r_{\nu_2} + c\nu_2$
$s_\chi \leftarrow r_\chi + c\chi$
$s_\varsigma \leftarrow r_\varsigma + c\varsigma$

$\xrightarrow{s_{\varepsilon_1}, s_{\varepsilon_2}, s_{\nu_1}, s_{\nu_2}, s_\chi, s_\varsigma}$

$g^{s_\rho} h^{s_\varsigma} \stackrel{?}{=} t_X X^c$
$(A'_1)^{s_{\varepsilon_1}} (R_0^{s_\chi})(S^{s_{\nu_1}}) \stackrel{?}{\equiv} \tilde{Z}_1 \left(\frac{Z}{{A'_1}^{2^{\ell_e - 1}} R_1^t} \right)^c \pmod n$
$(A'_2)^{s_{\varepsilon_2}} (R_0^{s_\chi} R_1^{s_\varsigma})(S^{s_{\nu_2}}) \stackrel{?}{\equiv} \tilde{Z}_2 \left(\frac{Z}{{A'_2}^{2^{\ell_e - 1}}} \right)^c \pmod n$
$s_{\varepsilon_1}, s_{\varepsilon_2} \stackrel{?}{\in} \pm\{0,1\}^{\ell'_e + \ell_\varnothing + \ell_\mathcal{H} + 1}$
$s_\chi, s_\varsigma \stackrel{?}{\in} \pm\{0,1\}^{\ell_m + \ell_\varnothing + \ell_\mathcal{H} + 1}$

Figure B.1: Complete zero-knowledge proof of knowledge for Σ_1.

From the third equation we can conclude that under the Strong RSA assumption $(c-c') \mid (s_\varsigma - s'_\varsigma)$ [IBM13], so we can rewrite the first equation as

$$g^{s_\rho - s'_\rho} h^{u(c-c')} = X^{c-c'},$$

with $u(c-c') = (s_\varsigma - s'_\varsigma)$. If $X \in \langle h \rangle$, we have that

$$g^v h^u = X,$$

where $v = \frac{s_\rho - s'_\rho}{c-c'} \mod \Gamma$ (because s_ρ and s'_ρ are computed modulo Γ). Note that we can check whether $X \in \langle h \rangle$ by checking $X^\Gamma \stackrel{?}{=} 1$; in this case we indeed trust the TTP to have chosen h to be a generator of the entire group, i.e., $|\langle h \rangle| = \Gamma = p_1 p_2 p_3$. From the last equation, we additionally know that the prover U knows the secret $\log_g X = \varsigma \in \pm\{0,1\}^{\ell_m + \ell_\varnothing + \ell_\mathcal{H} + 2}$, so the verifier knows the interval where ς is from. We note that if $|\langle h \rangle| \leq 2^{\ell_m + \ell_\varnothing + \ell_\mathcal{H} + 3}$, this would not give any extra information to the verifier.

The witnesses $\varepsilon_1 \equiv \frac{s_{\varepsilon_1} - s'_{\varepsilon_1}}{c-c'}$, $\varepsilon_2 \equiv \frac{s_{\varepsilon_2} - s'_{\varepsilon_2}}{c-c'}$, $\nu_1 \equiv \frac{s_{\nu_1} - s'_{\nu_1}}{c-c'}$, $\nu_2 \equiv \frac{s_{\nu_2} - s'_{\nu_2}}{c-c'}$, and $\chi \equiv \frac{s_\chi - s'_\chi}{c-c'}$, can be extracted in a similar way [IBM13].

Statistical honest-verifier zero-knowledgeness Provided that

$$Z, R_0, R_1, A_1, A_2 \in \langle S \rangle = \mathrm{QR}_n,$$

we have that the protocol is statistical honest-verifier zero-knowledge for sufficiently large $\ell_\varnothing$ [IBM13]. The protocol may be simulated by

$$\begin{aligned}
\big\{((t_X, \tilde{Z}_1, \tilde{Z}_2), c, (s_{\varepsilon_1}, s_{\varepsilon_2}, s_{\nu_1}, s_{\nu_2}, s_\chi, s_\varsigma)) : \\
s_\rho \leftarrow \mathbb{Z}_\Gamma, s_{\varepsilon_1}, s_{\varepsilon_2} \in_R \pm\{0,1\}^{\ell'_e + \ell_\varnothing + \ell_\mathcal{H}}, \\
s_{\nu_1}, s_{\nu_2} \in_R \pm\{0,1\}^{\ell_v + \ell_\varnothing + \ell_\mathcal{H}}, s_\chi, s_\varsigma \in_R \pm\{0,1\}^{\ell_m + \ell_\varnothing + \ell_\mathcal{H}}, \\
t_X \leftarrow g^{s_\rho} h^{s_\varsigma} X^{-c}, \tilde{Z}_1 \leftarrow (A'_1)^{s_{\varepsilon_1}} (R_0^{s_\chi})(S^{s_{\nu_1}}) \left(\frac{Z}{A_1'^{2^{\ell_e - 1}} R_1^t} \right)^{-c} \mod n, \\
\tilde{Z}_2 \leftarrow (A'_2)^{s_{\varepsilon_2}} (R_0^{s_\chi} R_1^{s_\varsigma})(S^{s_{\nu_2}}) \left(\frac{Z}{A_2'^{2^{\ell_e - 1}}} \right)^{-c} \mod n \big\}.
\end{aligned}$$

Where we note that the values $\mathbf{s}$ from the non-simulated protocol, e.g., $s_{\nu_1} = r_{\nu_1} + c\nu_1$, are statistically close to the corresponding interval from their $\mathbf{r}$ values, i.e., $\pm\{0,1\}^{\ell_v + \ell_\varnothing + \ell_\mathcal{H}}$ in the case of s_{ν_1}. □

B.3 Σ-Protocol for Σ_2

The complete zero-knowledge proof

$$\Sigma_2 = \big\{(\alpha_t, \beta_t) : Y = g^{\alpha_t} X^{\beta_t}\big\}(Y, n_2)$$

is described in Figure B.2.

Lemma 7. *The protocol depicted in Figure B.2 is a Σ-protocol.*

Proof. We will prove the three properties that make this protocol a Σ-protocol.

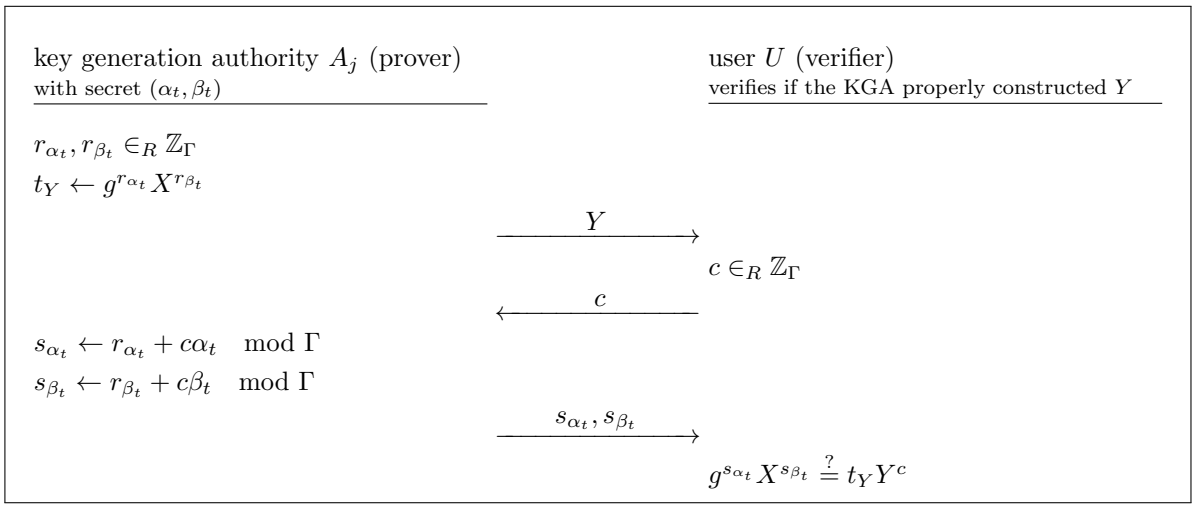

Figure B.2: Complete zero-knowledge proof of knowledge for Σ_2.

Completeness The protocol is complete:

$$g^{s_{\alpha_t}} X^{s_{\beta_t}} = g^{r_{\alpha_t}+c\alpha_t} X^{r_{\beta_t}+c\beta_t} = g^{r_{\alpha_t}} X^{r_{\beta_t}} \left(g^{\alpha_t} X^{\beta_t}\right)^c = t_Y Y^c.$$

Special soundness Given two accepting conversations $(t_Y, c, (s_{\alpha_t}, s_{\beta_t}))$ and $(t_Y, c', (s'_{\alpha_t}, s'_{\beta_t}))$ with $c \neq c'$, we can extract both witnesses α_t and β_t by computing

$$\alpha_t = \frac{s_{\alpha_t} - s'_{\alpha_t}}{c - c'} \mod \Gamma,$$
$$\beta_t = \frac{s_{\beta_t} - s'_{\beta_t}}{c - c'} \mod \Gamma.$$

This equation holds because we have $g^{s_{\alpha_t}} X^{s_{\beta_t}} = t_Y Y^c$ and $g^{s'_{\alpha_t}} X^{s'_{\beta_t}} = t_Y Y^{c'}$ which imply

$$g^{s_{\alpha_t}-s'_{\alpha_t}} X^{s_{\beta_t}-s'_{\beta_t}} = Y^{c-c'} \qquad \text{and} \qquad Y = g^{\frac{s_{\alpha_t}-s'_{\alpha_t}}{c-c'}} X^{\frac{s_{\beta_t}-s'_{\beta_t}}{c-c'}}.$$

Special honest-verifier zero-knowledgeness The stochastic distribution of an accepting conversation between an honest prover and an honest verifier, given an arbitrary challenge c, is

$$\big\{(t_Y, c, s_{\alpha_t}, s_{\beta_t}) : \; r_{\alpha_t}, r_{\beta_t} \in_R \mathbb{Z}_\Gamma, t_Y \leftarrow g^{r_{\alpha_t}} X^{r_{\beta_t}}, \\ s_{\alpha_t} \leftarrow r_{\alpha_t} + c\alpha_t \mod \Gamma, s_{\beta_t} \leftarrow r_{\beta_t} + c\beta_t \mod \Gamma\big\}.$$

We note that valid conversations $(t_Y, c, s_{\alpha_t}, s_{\beta_t})$ occur with probability $\frac{1}{\Gamma^2}$.

A simulator can simulate, again for arbitrary challenge c, a conversation with distribution

$$\big\{(t_Y, c, s_{\alpha_t}, s_{\beta_t}) : \; s_{\alpha_t}, s_{\beta_t} \leftarrow \mathbb{Z}_\Gamma, t_Y \leftarrow g^{s_{\alpha_t}} X^{s_{\beta_t}} Y^{-c}\big\}.$$

For valid conversations $(t, c, s_{\alpha_t}, s_{\beta_t})$ the outcome of both the 'real' as well the simulated conversations will occur with probability $\frac{1}{\Gamma^2}$. □

Summary

We have discussed the advantages of using Attribute-Based Credential (ABC) over 'classical' authentication, mentioning the privacy friendliness as one of the key aspects. The IRMA project implements such an ABC system on a smart card and builds an ecosystem around their setup. We have noted that an encryption scheme could be a valuable addition to the IRMA ecosystem and argued that Attribute-Based Encryption (ABE) is a good candidate for the type of encryption that we need. ABE allows a user to define an access policy on the ciphertext that determines which user can decrypt it. The access policy can be seen as a logical formula with the use of attributes as atoms, e.g., "professor AND CS department". Several types of ABE schemes exist. The schemes can be classified by analyzing four different properties.

- The access policy could reside in the decryption key or in the ciphertext.
- The number of public keys could grow in the number of attributes or remain constant.
- The scheme allows only one single authority or multiple authorities.
- Attributes can be negated in the access policy or not.

Using this classification we were able to select an ABE scheme that fits well in the IRMA ecosystem. However, several ABE schemes lack a privacy-friendly key issuance protocol. By introducing a new key issuance protocol that builds on the ABC system we were able to create a practical and privacy-friendly ABE scheme. Our protocol is proven secure on three points, using reasonable complexity assumptions.

- The user only obtains a decryption key when he has the right to.
- A malicious user learns nothing new from our protocol compared to the original key generation algorithm.
- A malicious authority learns nothing about the user's global identifier that makes the user linkable.

Because the decryption algorithm relies on fairly heavy computations for a smart card, we also created an off-card decryption protocol that allows us to securely outsource the complex calculations to a more powerful trusted device. The trusted device will not learn the user's decryption key, nor the plaintext of the encrypted message, only the smart card will.

List of Acronyms

AA	attribute authority
ABC	Attribute-Based Credential
ABE	Attribute-Based Encryption
BDHE	Bilinear Diffie–Hellman Exponent
CA	central authority
CCA	chosen ciphertext attack
CP-ABE	Ciphertext-Policy ABE
CPA	chosen plaintext attack
DBDH	Decisional Bilinear Diffie–Hellman
DDH	Decisional Diffie–Hellman
DDHI	Decisional Diffie–Hellman Inversion
DH	Diffie–Hellman
DL	Discrete Logarithm
DMBDH	Decisional Modified Bilinear Diffie–Hellman
eID	electronic identity
GID	global identifier
IBE	Identity-Based Encryption
idemix	Identity Mixer
IND	ciphertext indistinguishability
IRMA	I Reveal My Attributes
KGA	key generation authority
KGC	key generation center
KP-ABE	Key-Policy ABE
LSSS	Linear Secret Sharing Scheme

NFC	near field communication
MA-ABE	multi-authority ABE
MK	master key
PK	public parameter
PKI	Public Key Infrastructure
sAtt	selective-attribute
sid	selective ID
SK	private key
SSSS	Shamir's Secret Sharing Scheme
STA	Semi-Trusted Authority
TTP	Trusted Third Party

List of Definitions

Bibliography

[BBD+01] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. "Key-Privacy in Public-Key Encryption." In: *Advances in Cryptology—ASIACRYPT 2001*. Ed. by Colin Boyd. Vol. 2248. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2001, pp. 566–582. ISBN: 978-3-540-42987-6. DOI: 10.1007/3-540-45682-1_33.

[BBG05a] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. *Hierarchical Identity Based Encryption with Constant Size Ciphertext*. Report. Stanford University, 2005. IACR: http://eprint.iacr.org/2005/015.

[BBG05b] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. "Hierarchical Identity Based Encryption with Constant Size Ciphertext." In: *Advances in Cryptology—EUROCRYPT 2005*. Ed. by Ronald Cramer. Vol. 3494. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, pp. 440–456. ISBN: 978-3-540-25910-7. DOI: 10.1007/11426639_26.

[Bei96] Amos Beimel. "Secure Schemes for Secret Sharing and Key Distribution." PhD thesis. Israel Institute of Technology, Technion, Haifa, Israel, June 1996. URL: https://www.iacr.org/phds/index.php?p=detail&entry=548.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. "Evaluating 2-DNF Formulas on Ciphertexts." In: *Theory of Cryptography*. Ed. by Joe Kilian. Vol. 3378. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, pp. 325–341. ISBN: 978-3-540-24573-5. DOI: 10.1007/978-3-540-30576-7_18.

[BSW07] J. Bethencourt, A. Sahai, and B. Waters. "Ciphertext-Policy Attribute-Based Encryption." In: *Security and Privacy, 2007. SP '07. IEEE Symposium on*. May 2007, pp. 321–334. ISBN: 0-7695-2848-1. DOI: 10.1109/SP.2007.11.

[CC09] Melissa Chase and Sherman S.M. Chow. "Improving Privacy and Security in Multi-authority Attribute-based Encryption." In: *Proceedings of the 16th ACM Conference on Computer and Communications Security*. CCS '09. New York, NY, USA: ACM, 2009, pp. 121–130. ISBN: 978-1-60558-894-0. DOI: 10.1145/1653662.1653678.

[CDN01] Ronald Cramer, Ivan Damgård, and Jesper B. Nielsen. "Multiparty Computation from Threshold Homomorphic Encryption." In: *Advances in Cryptology—EUROCRYPT 2001*. Ed. by Birgit Pfitzmann. Vol. 2045. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2001, pp. 280–300. ISBN: 978-3-540-42070-5. DOI: 10.1007/3-540-44987-6_18.

[Cha07] Melissa Chase. "Multi-authority Attribute Based Encryption." In: *Theory of Cryptography*. Ed. by Salil P. Vadhan. Vol. 4392. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, pp. 515–534. ISBN: 978-3-540-70935-0. DOI: 10.1007/978-3-540-70936-7_28.

[CKR+09] Jan Camenisch, Markulf Kohlweiss, Alfredo Rial, and Caroline Sheedy. "Blind and Anonymous Identity-Based Encryption and Authorised Private Searches on Public Key Encrypted Data." In: *Public Key Cryptography—PKC 2009*. Ed. by Stanisław Jarecki and Gene Tsudik. Vol. 5443. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, pp. 196–214. ISBN: 978-3-642-00467-4. DOI: 10.1007/978-3-642-00468-1_12.

[CL01] Jan Camenisch and Anna Lysyanskaya. "An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation." In: *Advances in Cryptology—EUROCRYPT 2001*. Ed. by Birgit Pfitzmann. Vol. 2045. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2001, pp. 93–118. ISBN: 978-3-540-42070-5. DOI: 10.1007/3-540-44987-6_7.

[CL03] Jan Camenisch and Anna Lysyanskaya. "A Signature Scheme with Efficient Protocols." In: *Security in Communication Networks*. Ed. by Stelvio Cimato, Giuseppe Persiano, and Clemente Galdi. Vol. 2576. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2003, pp. 268–289. ISBN: 978-3-540-00420-2. DOI: 10.1007/3-540-36413-7_20.

[CN07] Ling Cheung and Calvin Newport. "Provably Secure Ciphertext Policy ABE." In: *Proceedings of the 14th ACM Conference on Computer and Communications Security*. CCS '07. New York, NY, USA: ACM, 2007, pp. 456–465. ISBN: 978-1-59593-703-2. DOI: 10.1145/1315245.1315302.

[CNs07] Jan Camenisch, Gregory Neven, and abhi shelat. "Simulatable Adaptive Oblivious Transfer." In: *Advances in Cryptology—EUROCRYPT 2007*. Ed. by Moni Naor. Vol. 4515. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, pp. 573–590. ISBN: 978-3-540-72539-8. DOI: 10.1007/978-3-540-72540-4_33.

[CS97] Jan Camenisch and Markus Stadler. "Efficient Group Signature Schemes for Large Groups." In: *Advances in Cryptology—CRYPTO '97*. Ed. by Burton S. Kaliski Jr. Vol. 1294. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1997, pp. 410–424. ISBN: 978-3-540-63384-6. DOI: 10.1007/BFb0052252.

[CZF11] Cheng Chen, Zhenfeng Zhang, and Dengguo Feng. "Efficient Ciphertext Policy Attribute-Based Encryption with Constant-Size Ciphertext and Constant Computation-Cost." In: *Provable Security*. Ed. by Xavier Boyen and Xiaofeng Chen. Vol. 6980. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, pp. 84–101. ISBN: 978-3-642-24315-8. DOI: 10.1007/978-3-642-24316-5_8.

[FS87] Amos Fiat and Adi Shamir. "How To Prove Yourself: Practical Solutions to Identification and Signature Problems." In: *Advances in Cryptology—CRYPTO '86*. Ed. by Andrew M. Odlyzko. Vol. 263. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1987, pp. 186–194. ISBN: 978-3-540-18047-0. DOI: 10.1007/3-540-47721-7_12.

[GH07] Matthew Green and Susan Hohenberger. "Blind Identity-Based Encryption and Simulatable Oblivious Transfer." In: *Advances in Cryptology—ASIACRYPT 2007*. Ed. by Kaoru Kurosawa. Vol. 4833. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, pp. 265–282. ISBN: 978-3-540-76899-9. DOI: 10.1007/978-3-540-76900-2_16.

[GL13] Ang Gao and Zengzhi Li. "Free global ID against collusion attack on multi-authority attribute-based encryption." In: *Security and Communication Networks* 6.9 (2013), pp. 1143–1152. ISSN: 1939-0122. DOI: 10.1002/sec.683.

[GPS+06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. "Attribute-based Encryption for Fine-grained Access Control of Encrypted Data." In: *Proceedings of the 13th ACM Conference on Computer and Communications Security*. CCS '06. New York, NY, USA: ACM, 2006, pp. 89–98. ISBN: 1-59593-518-5. DOI: 10.1145/1180405.1180418.

[HJV10] Jaap-Henk Hoepman, Bart Jacobs, and Pim Vullers. "Privacy and Security Issues in e-Ticketing—Optimisation of Smart Card-based Attribute-proving." In: *Workshop on Foundations of Security and Privacy—FCS-PrivMod 2010*. Ed. by Veronique Cortier, Mark Ryan, and Vitaly Shmatikov. (Informal). July 2010. URL: http://www.cs.ru.nl/~pim/publications/2010_privmod.pdf.

[HSM+12] Jinguang Han, W. Susilo, Yi Mu, and Jun Yan. "Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption." In: *Parallel and Distributed Systems, IEEE Transactions on* 23.11 (Nov. 2012), pp. 2150–2162. ISSN: 1045-9219. DOI: 10.1109/TPDS.2012.50.

[HSM+14] Jinguang Han, Willy Susilo, Yi Mu, Jianying Zhou, and Man Ho Au. *PPDCP-ABE: Privacy-Preserving Decentralized Cipher-Policy Attribute-Based Encryption*. Report. Nanjing University of Finance and Economics, University of Wollongong, Institute for Infocomm Research, June 2014. IACR: http://eprint.iacr.org/2014/470.

[IBM13] IBM Research. *Specification of the Identity Mixer Cryptographic Library*. Report. Version 2.3.43. IBM Research—Zürich, Jan. 2013. URL: http://www.zurich.ibm.com/idemix/.

[IPN+09] Luan Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker. *Ciphertext-policy attribute-based threshold decryption with flexible delegation and revocation of user attributes*. Report. University of Twente, Apr. 2009. URL: http://doc.utwente.nl/65471/.

[ITH+09] Luan Ibraimi, Qiang Tang, Pieter Hartel, and Willem Jonker. "Efficient and Provable Secure Ciphertext-Policy Attribute-Based Encryption Schemes." In: *Information Security Practice and Experience*. Ed. by Feng Bao, Hui Li, and Guilin Wang. Vol. 5451. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, pp. 1–12. ISBN: 978-3-642-00842-9. DOI: 10.1007/978-3-642-00843-6_1.

[LC10] Zhen Liu and Zhenfu Cao. *On Efficiently Transferring the Linear Secret-Sharing Scheme Matrix in Ciphertext-Policy Attribute-Based Encryption*. Report. Shanghai Jiao Tong University, 2010. IACR: http://eprint.iacr.org/2010/374.

[LCH13] Cheng-Chi Lee, Pei-Shan Chung, and Min-Shiang Hwang. "A Survey on Attribute-based Encryption Schemes of Access Control in Cloud Environments." In: *International Journal of Network Security* 15.4 (July 2013), pp. 231–240. URL: http://asiair.asia.edu.tw/ir/handle/310904400/25385.

[LCH+11] Zhen Liu, Zhenfu Cao, Qiong Huang, Duncan S. Wong, and Tsz Hon Yuen. "Fully Secure Multi-authority Ciphertext-Policy Attribute-Based Encryption without Random Oracles." In: *Computer Security—ESORICS 2011*. Ed. by Vijay Atluri and Claudia Diaz. Vol. 6879. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, pp. 278–297. ISBN: 978-3-642-23821-5. DOI: 10.1007/978-3-642-23822-2_16.

[LCL+08] Huang Lin, Zhenfu Cao, Xiaohui Liang, and Jun Shao. "Secure Threshold Multi Authority Attribute Based Encryption without a Central Authority." In: *Progress in Cryptology—INDOCRYPT 2008*. Ed. by Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das. Vol. 5365. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, pp. 426–436. ISBN: 978-3-540-89753-8. DOI: 10.1007/978-3-540-89754-5_33.

[Lin10] Yehuda Lindell. "Anonymous Authentication." In: *Journal of Privacy and Confidentiality* 2.2 (2010), pp. 35–63. URL: http://repository.cmu.edu/jpc/vol2/iss2/4/.

[LRR+13] Fei Li, Yogachandran Rahulamathavan, Muttukrishnan Rajarajan, and Raphael C.-W. Phan. "Low Complexity Multi-authority Attribute Based Encryption Scheme for Mobile Cloud Computing." In: *Service Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on*. Mar. 2013, pp. 573–577. DOI: 10.1109/SOSE.2013.12.

[LW10] Allison Lewko and Brent Waters. *Decentralizing Attribute-Based Encryption*. Report. University of Texas at Austin, 2010. IACR: http://eprint.iacr.org/2010/351.

[LW11] Allison Lewko and Brent Waters. "Decentralizing Attribute-Based Encryption." In: *Advances in Cryptology—EUROCRYPT 2011*. Ed. by Kenneth G. Paterson. Vol. 6632. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, pp. 568–588. ISBN: 978-3-642-20464-7. DOI: 10.1007/978-3-642-20465-4_31.

[LXZ+13] Qinyi Li, Hu Xiong, Fengli Zhang, and Shengke Zeng. "An Expressive Decentralizing KP-ABE Scheme with Constant-Size Ciphertext." In: *International Journal of Network Security* 15.3 (2013), pp. 161–170. URL: http://ijns.femto.com.tw/contents/ijns-v15-n1/ijns-v15-n1.html.

[MKE09a] Sascha Müller, Stefan Katzenbeisser, and Claudia Eckert. "Distributed Attribute-Based Encryption." In: *Information Security and Cryptology—ICISC 2008*. Ed. by Pil Joong Lee and Jung Hee Cheon. Vol. 5461. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, pp. 20–36. ISBN: 978-3-642-00729-3. DOI: 10.1007/978-3-642-00730-9_2.

[MKE09b] Sascha Müller, Stefan Katzenbeisser, and Claudia Eckert. "On Multi-Authority Ciphertext-Policy Attribute-Based Encryption." In: *Bulletin of the Korean Mathematical Society* 46.4 (2009), pp. 803–819. DOI: 10.4134/BKMS.2009.46.4.803.

[MOV96] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. 5th ed. CRC Press, Oct. 1996. 816 pp. ISBN: 0-8493-8523-7. URL: http://cacr.uwaterloo.ca/hac/.

[MV12] Wojciech Mostowski and Pim Vullers. "Efficient U-Prove Implementation for Anonymous Credentials on Smart Cards." In: *Security and Privacy in Communication Networks*. Ed. by Muttukrishnan Rajarajan, Fred Piper, Haining Wang, and George Kesidis. Vol. 96. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer Berlin Heidelberg, 2012, pp. 243–260. ISBN: 978-3-642-31908-2. DOI: 10.1007/978-3-642-31909-9_14.

[OSW07] Rafail Ostrovsky, Amit Sahai, and Brent Waters. "Attribute-based Encryption with Non-monotonic Access Structures." In: *Proceedings of the 14th ACM Conference on Computer and Communications Security*. CCS '07. New York, NY, USA: ACM, 2007, pp. 195–203. ISBN: 978-1-59593-703-2. DOI: 10.1145/1315245.1315270.

[Ped92] Torben Pryds Pedersen. "Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing." In: *Advances in Cryptology—CRYPTO '91*. Ed. by Joan Feigenbaum. Vol. 576. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1992, pp. 129–140. ISBN: 978-3-540-55188-1. DOI: 10.1007/3-540-46766-1_9.

[QLZ13] Huiling Qian, Jiguo Li, and Yichen Zhang. "Privacy-Preserving Decentralized Ciphertext-Policy Attribute-Based Encryption with Fully Hidden Access Structure." In: *Information and Communications Security*. Ed. by Sihan Qing, Jianying Zhou, and Dongmei Liu. Vol. 8233. Lecture Notes in Computer Science. Springer International Publishing, 2013, pp. 363–372. ISBN: 978-3-319-02725-8. DOI: 10.1007/978-3-319-02726-5_26.

[RD13] Y. Sreenivasa Rao and Ratna Dutta. "Decentralized Ciphertext-Policy Attribute-Based Encryption Scheme with Fast Decryption." In: *Communications and Multimedia Security*. Ed. by Bart Decker, Jana Dittmann, Christian Kraetzer, and Claus Vielhauer. Vol. 8099. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, pp. 66–81. ISBN: 978-3-642-40778-9. DOI: 10.1007/978-3-642-40779-6_5.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. "A Method for Obtaining Digital Signatures and Public-key Cryptosystems." In: *Communications of the ACM* 21.2 (Feb. 1978), pp. 120–126. ISSN: 0001-0782. DOI: 10.1145/359340.359342.

[RW13] Yannis Rouselakis and Brent Waters. "Practical Constructions and New Proof Methods for Large Universe Attribute-based Encryption." In: *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*. CCS '13. New York, NY, USA: ACM, 2013, pp. 463–474. ISBN: 978-1-4503-2477-9. DOI: 10.1145/2508859.2516672.

[Sch91] C.P. Schnorr. "Efficient Signature Generation by Smart Cards." In: *Journal of Cryptology* 4.3 (1991), pp. 161–174. ISSN: 0933-2790. DOI: 10.1007/BF00196725.

[Sha79] Adi Shamir. "How to Share a Secret." In: *Communications of the ACM* 22.11 (Nov. 1979). Ed. by R. Rivest, pp. 612–613. ISSN: 0001-0782. DOI: 10.1145/359168.359176.

[Sha85] Adi Shamir. "Identity-Based Cryptosystems and Signature Schemes." In: *Advances in Cryptology*. Ed. by George Robert Blakley and David Chaum. Vol. 196. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1985, pp. 47–53. ISBN: 978-3-540-15658-1. DOI: 10.1007/3-540-39568-7_5.

[SW05] Amit Sahai and Brent Waters. "Fuzzy Identity-Based Encryption." In: *Advances in Cryptology—EUROCRYPT 2005*. Ed. by Ronald Cramer. Vol. 3494. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, pp. 457–473. ISBN: 978-3-540-25910-7. DOI: 10.1007/11426639_27.

[VA13] Pim Vullers and Gergely Alpár. "Efficient Selective Disclosure on Smart Cards Using Idemix." In: *Policies and Research in Identity Management*. Ed. by Simone Fischer-Hübner, Elisabeth de Leeuw, and Chris Mitchell. Vol. 396. IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, Apr. 2013, pp. 53–67. ISBN: 978-3-642-37281-0. DOI: 10.1007/978-3-642-37282-7_5.

[WLW+11] Guojun Wang, Qin Liu, Jie Wu, and Minyi Guo. "Hierarchical attribute-based encryption and scalable user revocation for sharing data in cloud servers." In: *Computers & Security* 30.5 (2011). Advances in network and system security, pp. 320–331. ISSN: 0167-4048. DOI: 10.1016/j.cose.2011.05.006.

[XZ11] Lingling Xu and Fangguo Zhang. "Oblivious Transfer with Complex Attribute-Based Access Control." In: *Information Security and Cryptology—ICISC 2010*. Ed. by Kyung-Hyune Rhee and DaeHun Nyang. Vol. 6829. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, pp. 370–395. ISBN: 978-3-642-24208-3. DOI: 10.1007/978-3-642-24209-0_25.