Bachelor thesis
Computing Science

Radboud University

Using IRMA for small scale digital elections

Author:
J.J.J. Doesburg
s4809327

First supervisor/assessor:
Prof. dr. B.P.F. Jacobs
bart@cs.ru.nl

Second assessor:
Dr. S. Ringers
sringers@cs.ru.nl

January 17, 2020

Abstract

The IRMA framework is a real-life implementation of the IBM Idemix attribute-based credential system [10, 45]. Additionally, the IRMA ecosystem features attribute-based signatures that allow for creation of privacy-friendly digital signatures [26]. Interest in the application of IRMA has grown in the last years in the Netherlands, especially in the public sector. As an example, the municipality of Amsterdam is aiming to apply IRMA in all of its public services. Among these is the OpenStad project that involves small-scale digital citizen consultation and referenda. Digital voting is considered a more accessible alternative for traditional paper voting, which is especially important for small-scale elections.

We see that existing voting schemes for digital elections often turn out to be impractical and remain merely academic [32]. Those rely on classic approaches to identity management. So far, however, there has been no research to solve the problem of digital elections using attribute-based credential systems and attribute-based signatures.

In this research, we identify requirements for voting systems from a legal context. We present a scheme that uses attribute-based signatures to record votes in a verifiable and privacy-friendly manner, using blindly issued voting numbers. Two ways to realize this are considered: either blind double signatures or blind generation of voting numbers. Both solutions rely on minimal changes to the Idemix scheme for credential issuance. Based on this, a voting scheme is presented that can be used to realize online remote elections.

Additionally, we present extensions to, among others, further increase accessibility by weakening restrictions on voter registration and vote casting phases, and we present considerations for implementing a public register of votes, to make the scheme less vulnerable to certain forms of coercion. With this, the voting scheme we present satisfies most desired properties for elections: eligibility and unicity while maintaining secrecy, integrity, and verifiability. However, we also discuss limitations fundamental to online remote voting in general and address practical problems.

We conclude that digital elections are not recommendable for large scale elections with large societal influences, but for small scale elections, the increased accessibility and limited attacker models make it recommendable. We have described a good way to start the development of proof of concept digital elections on IRMA and have adequately identified problems that require further research.

Contents

Chapter 1

Introduction

A question citizens in our modern society might regularly ask themselves, is why despite all technological progress of these times, the general consensus of academia is that voting processes should be performed offline and on paper. In recent years, popular media has increasingly paid attention to this matter, with examples of hackable election systems. Meanwhile, our current society is experiencing an increasing distance between democratic processes and citizens that become more and more independent. Easily accessible forms of citizen participation and consultation might be able to reduce this distance. Online remote elections could offer a solution for this.

1.1 IRMA: an attribute-based credential system

IRMA[1](short for: I Reveal My Attributes) is an attribute-based privacy-enhancing credential system (as defined in [44, 35]), developed in the past 10 years. IRMA is a (restricted) implementation of the IBM Idemix credential system ([10, 45]). The project started as a smartcard implementation [47] but has grown to an ecosystem that uses a smartphone application [4].

Attributed based credential systems such as IRMA, are a different approach to classic identity management. Instead of users revealing their identity to a party (the identity provider) that determines whether the user is authorized, authorization is performed directly without any intervening party, based on attributes that users carry themselves. Those attributes are minimal pieces of information (such as NAME, AGE, or NATIONALITY) that describe an individual. This information is issued to a user, who can then disclose them to any verifier.

[1]More information can be found on `https://privacybydesign.foundation/irma-explanation/`, `https://credentials.github.io` or `https://irma.app/docs/`.

Attributes are not necessarily identifying. Therefore, to be authorized, users do not always need to reveal their full identity. Rather, they can selectively disclose the attributes required. Cryptographic schemes and zero-knowledge proofs assure the validity of the attributes, which are signed by issuing parties, while also maintaining optimal privacy of the attribute owners, by, among others, making usage unlinkable to other usages or the user's full identity [9].

In the past years, the IRMA ecosystem has started supporting several kinds of attributes. Especially in fields where privacy is a sensitive and important topic, such as health care, there is interest in using IRMA. Also, governmental organizations are interested in IRMA. The first real applications are currently appearing.

Meanwhile, several improvements are being implemented to the system to enhance the usage and versatility. The most recent developments are the introduction of attribute-based signatures to the system (as proposed in [26]) and the upcoming introduction of issuer-revocation of attributes. Those additions to the system enable new applications of IRMA that are worth investigating.

1.2 Amsterdam OpenStad and Digital Identity

In 2016, the municipality of Amsterdam started the project 'OpenStad Amsterdam' *(Amsterdam Open City)*. The objective of this project was to get citizens actively involved in the governing of the city. To achieve this goal, (online) tools are being developed that enable citizens to easily participate in decision making. On a governmental level, the OpenStad project made budget available and citizens are being invited to make their own proposals on how to spend that money. On a technological level, the municipality started the development of what they call 'participation tools': applications that enable people to share their ideas, discuss with each other and vote on proposals[2].

More recently, the municipality started the project 'Digitale Stad' *(Digital City)*, where the municipality, among others, started investigating how to administer the digital identity of Amsterdam's citizens. Privacy was one of the main focus points of this project. Because of this, they started investing in IRMA. The goal of the project is to decide on and contribute to a platform for a universal and privacy-friendly credential system for all services related to the city of Amsterdam.

[2]Information in these paragraphs is based on explanation personally provided by project leader Mark Fonds.

One of the use cases for this digital identity system is the OpenStad project. Citizens should be able to use their digital identity within the Open City tools. This must, however, be done in a privacy-friendly manner. This is especially relevant for digital voting because of the privacy aspects related. Even though the elections in the Open City context are rather small and not very influential (and not bound to the election law), privacy can still play an important role (some decisions might still be quite sensitive locally).

Ideally, we would have a way to vote anonymously *by design*, where it is impossible to link a vote to the person who cast it, without having to trust any party.

In offline elections, this is no big problem. Time has proven the usage of paper ballots and the system with voting passes to provide sufficient anonymity. Those elections, however, are quite expensive (millions for nation-wide elections in countries such as the Netherlands[3]). Voting also takes quite a large time investment of citizens, who need to go to a polling station physically. For small scale referenda, like those in the OpenStad project, it is undesirable to have elections similar to these nation-wide general elections, since the time-investment required is just too high. Similar observations are found in [1] and supported by [38].

To keep the threshold of participation in these elections low, it should be able for citizens to vote from home digitally with ease. The OpenStad project, therefore, currently organizes online elections with voting codes sent by (paper) mail. However, this process remains rather expensive and is still not very simple or reliable (since voting passes can get lost or end up at different persons).

To find a better way, the digital identity project started experimenting with online voting using IRMA. Voter identification using IRMA will be used in the voting tools developed for the Open City project. This, however, poses some problems. To have digital elections take place according to *privacy by design* principles, we cannot allow one's identity to be disclosed in the process of casting a vote at all. We only want to reveal the minimal amount of information: this person is eligible to vote. Nevertheless, it is required to ensure that basic election principles (only eligible people can vote, eligible people can vote only once) are met. Even though IRMA provides various privacy benefits, those principles can not natively be realized using basic IRMA application.

Apart from the municipality of Amsterdam, other municipalities in the Netherlands are interested in applying IRMA in the process of citizen participation as well. As an example, the municipality of Groningen aims to

[3]Based on data of the Dutch governmental budget for 2010 (Rijksoverheid).

apply IRMA in the software project Consul[4]. Additionally, various municipalities have started issuing IRMA attributes to their citizens in order to support new initiatives.

1.3 Digital elections

Digital voting has had a tumultuous history in the Netherlands, as described in [30]. Though the Netherlands was among the early adopters of (electronic) voting machines in polling stations, the use was discontinued after several legislative requirements turned out not to be met, after an activist group brought the matter under attention. Despite several attempts to solve the problems, in 2008, the government decided to keep voting on paper for the near future. Other experiments with online voting (e.g., using the RIES system [29]) were also discontinued after several attempts, as described in [30].

Even if the usage of an online system was not bound to the Dutch election law (for example, in certain water board elections [30]), we see that a high level of trust must exist for a technology to be successful. Thus, not adopting a digital or online system is not merely a regulatory case.

There exist legislative requirements for elections that describe the fundamental properties a voting scheme (both offline and online) must meet. In the Netherlands, there is the election law, and many countries worldwide have similar (constitutional) laws with equivalent requirements. Examples of these properties are verifiability, secrecy, and integrity, but also accessibility. Those principles and requirements have been researched in-depth, among others in [25, 37, 48, 2].

In classic (offline) paper elections, these requirements are all met to some extent, either by fundamental technological properties of the medium (it is impossible to look through a folded piece of paper) or organizational measures: the persons at the polling station have different (political) backgrounds, and the person verifying someone's identity card is not the same person handing out the empty ballot sheet. Digital online elections have a fundamentally different setup and therefore require different measures to realize the same properties [5]. Specific offline organizational measures that cannot be implemented online, but might be replaceable with cryptographic solutions that cannot be realized on paper elections.

Although according to exact terminology, digital elections involve both elections using an electronic voting machine and online elections, where people

[4] `http://consulproject.org/en/`

remotely vote via the internet (e.g., from their smartphone), in this research, we will focus on only the last category and ignore other forms of electronic voting.

1.4 Current state of research

Most online voting systems, both experimented within the Netherlands and applied worldwide, make use of some nation-wide electronic identity (such as the Dutch DigiD) for voter identification. This is the classic identity management approach. By applying cryptographic schemes, the votes are then anonymized before they are recorded. This, however, does not provide privacy *by design*: the user does not have the fundamental guarantee their identity is not recorded linkable to their vote. The required properties often rely on organizational measures. An example of a country that has widely adopted digital voting this way is Estonia, which we discuss further in section 3.1.

Though the idea of digital elections in the past has been received with criticism [49], the field of research on electronic voting has become increasingly active in recent years [31]. Papers presented at the International Conference for Electronic Voting (E-Vote-ID), vary a lot, from online remote elections to elections with electronic voting machines, and from fundamental cryptographic schemes to legal or administrative aspects. Nevertheless, we see that many (cryptographic) schemes for electronic elections often turn out to be impractical and remained merely academic [32].

To the best of our knowledge, so far, no research has been conducted to use attribute-based credential systems (such as IRMA) that take a fundamentally different approach to the identification problem for digital elections[5]. The IRMA framework tries to deliver a simple and user-friendly implementation of such a system. Moreover, is it a system with multiple use cases. The adoption of IRMA would not be election-specific, but elections would just be another application of IRMA. Therefore, systems based on IRMA might better meet the requirements. Additionally, attribute-based signatures, with non-identifying attributes, as we will see, form a native and practical basis to record votes in a verifiable yet privacy-friendly manner, further simplifying any solution.

[5]Note that several systems do introduce some voting number attribute used in the votes. However, because those attributes are essentially issued by the same party as the verifying party, we do not consider that a *privacy by design* solution according to the definition of attribute-based credential systems [17, 10, 44, 9].

1.5 Our contribution

This research will aim to investigate to what extent IRMA can be used in digital elections. Although the use case we will discuss most in-depth will be the small scale public elections described in section 1.2, we will also look at general elections.

We investigate what the (regulatory) requirements for (public) elections are, and we draw parallels between offline and online elections. We present a voting scheme that describes how users can acquire attributes and perform IRMA sessions to realize digital elections, that allows only eligible users to vote at most once while maintaining secrecy. Moreover, the scheme results in a transparent election outcome verifiable by anyone. The essence of this is briefly mentioned in [26]. In this research we elaborate on this use case scenario, with many details and variations, including blindly issued attributes. We explore how a proof-of-concept of IRMA-based elections could be realized and what are shortcomings that require further research.

The solutions we present are practical and persist under real-world problems, such as users losing access to their credentials. Based on this, we will conclude to what extent the application of IRMA in (small scale) digital elections is possible and desirable.

In a more general sense, the solutions presented here might partially (with several small adaptations) also be applicable in different attribute-based credential systems, both based on Idemix or different schemes that have a similar setup. However, the proposal will be optimized for IRMA and could, therefore, contain features that in other systems could be implemented more optimally.

Reading guide In chapter 2, the basic principles relevant to the topic will be discussed in more detail. Readers with sufficient background knowledge in RSA, zero-knowledge proofs, the IRMA framework, and attribute revocation using cryptographic accumulators are advised to skip this chapter. In chapter 3, a legal framework of elections will be considered, hence describing the requirements our solution should satisfy. Additionally, several small and practical recommendations for the implementation of electronic voting systems are discussed. In chapter 4, we then describe our main contribution by proposing the voting scheme and the usage within a digital election. The limitations and details that should be taken into account will be discussed in chapter 5. Here we also propose some extensions or adaptations to the scheme to improve performance in practical situations. Finally, in chapter 6 and 7, we propose ideas for further research to improve the proposed scheme further or apply it in other contexts, and we draw a general conclusion respectively.

Chapter 2

Preliminaries

In this chapter, background information on the most important concepts related to this research will be given. This involves a basic explanation of RSA and blind signatures, zero-knowledge proofs, terminology of attribute-based signature schemes, cryptographic accumulators, and a brief technical overview of the IRMA framework.

2.1 RSA and blind signatures

RSA is an asymmetric cryptosystem proposed in 1983 by Rivest, Shamir, and Adleman [43] that has become one of the most commonly used methods for public-key encryption and digital signatures. It also forms the cornerstone of the cryptography used in the schemes we propose. We will give a short overview of the basic principle of RSA and the RSA assumption and how it is used in our field.

Note that many statements about RSA here are overstatements or simplifications. There are several weaknesses in the system and its usage, as is described here. In practice, there are measures that can overcome these problems, for RSA to provide sufficient security. For the purpose of this chapter, we will not consider most of those details, however, and stick to the somewhat simplified view.

2.1.1 Asymmetric cryptography

In asymmetric cryptosystems like RSA, keys for encryption and decryption are not set up symmetrically between two parties. Instead, every party generates a private/public key pair. The public key is published for everyone to see, the private key is kept private to only the owner. Whenever someone wants to encrypt information for someone, he or she retrieves the public key

and uses it to encrypt the data. The data can then only be decrypted with knowledge of the private key.

Additionally, the private key can be used to create digital signatures on data. Those signatures can then be published, and anyone can verify that the signature is indeed valid (by verifying that the signature is created with a private key that is related to the public key that is widely known). Still, the value of the private key itself remains private.

Like every asymmetric cryptosystem, the security of RSA is provided by the use of a so-called trapdoor function: a function that is easy to calculate, but hard to calculate the inverse of, unless with knowledge of some secret information (the private key). A pure mathematical definition of trapdoor functions is given in [39].

2.1.2 RSA assumption

The RSA cryptosystem uses prime number factorization as trapdoor function: computing the product of two integers is rather easy, but finding a number's prime factors is a computationally hard problem, since the only way to calculate it is by iteratively and recursively try dividing by different primes[1].

This trapdoor function, the problem of prime factorization, is then used in the concept of modular exponentiation, forming the so-called strong RSA assumption. The strong RSA assumption states that given the values of y and n it remains infeasible to come up with integers x and d such that

$$x^d \equiv y \pmod{n} \tag{2.1}$$

We can use this property in encryption and decryption with RSA. For a message m encrypted using e, we see that it remains very hard (mathematically intractable) to find d such that

$$(m^e)^d \equiv m \pmod{n} \tag{2.2}$$

The public key here is formed by e (and n) and the private key is formed by d (and n), which is the modular inverse exponent for e within group n (as follows from Equation 2.1 and 2.2).

Encryption When trying to create a ciphertext c using RSA encryption, one simply retrieves the public key (e, n) for the user that should be able to decrypt the message again, and calculates

$$c \equiv m^e \pmod{n} \tag{2.3}$$

[1]This statement is obviously an oversimplification. In practice, there exist various methods that are subtly more optimal, however yet no general efficient algorithm exists to compute a number's prime factors, so for the purpose of this text, we will work with this simplified view.

Decryption Now to decrypt the ciphertext c the receiver should invert the modular exponentiation step, which can only be done efficiently with knowledge of private key d

$$m \equiv (m^e)^d \equiv c^d \pmod{n} \tag{2.4}$$

Key generation When generating our key pair with the values e, d and n, we first choose two arbitrary large primes p and q and calculate their product n. We now pick a value for e, such that e is coprime[2] to $\phi(n)$, where ϕ is known as totient function[3] computing $(p-1)*(q-1)$, for p and q being the two prime factors of n. Now we calculate d to be the modular multiplicative inverse of e using $\phi(n)$, which can be calculated easily since we still know the prime factors of n (p and q). An adversary, however, cannot easily calculate d, since they do not know the prime factors.

After the key pair has been generated, p and q can be discarded obviously.

2.1.3 RSA signatures

Naturally RSA can also be used to create cryptographic signatures. For this purpose often a hash value is used, but this is not a strict mathematical requirement for just RSA (the use of hashes is merely a highly advisable requirement for creating signatures on large messages).

Signature creation To create a signature s on a (hashed) message m, one computes using private key d

$$s \equiv m^d \pmod{n} \tag{2.5}$$

Signature verification To verify a signature s on a (hashed) message m, anyone can verify using the public key e that

$$m \equiv (m^e)^d \equiv (m^d)^e \equiv s^e \pmod{n} \tag{2.6}$$

Note that in practice, it is highly advisable to *not* use the same key pair for creating signatures and encryption/decryption, especially when it comes to blind signatures that will be discussed next, since we do not want to leak information about how to use the private key for decryption.

[2]In practice, there are additional requirements to these values to provide sufficient security and efficient computations. Often for e the value 65,537 is chosen.

[3]The usage of ϕ follows from the original proposal in [43], but in practice often a subtly different function (Carmichael's totient function [14]) is used, which behaves quite similarly and offers equivalent security

2.1.4 Blind signatures

Blind signatures, as firstly proposed by Chaum in [15], are a technique to create signatures for a message, where the party generating the signature does *not* know the actual message itself. Blind signatures can be realized in several (asymmetric) cryptosystems, among which RSA.

The realization of blind signatures in RSA is straightforward, and the process is similar to the creation of regular signatures. The process involves a so-called blinding factor that is applied to the message to hide its contents while it is signed, and is then removed, while the signature remains intact.

Creating blinded message To create a RSA blind signature, one first chooses a random value r that is coprime to n in the public key of the signer. Then the blinded message m' to be signed is computed using

$$m' \equiv mr^e \pmod{n} \tag{2.7}$$

and is sent to the signer. Note that because r is random, it does not leak information about our plain message m. Also note that again here, we will often use a hashed message m.

Creating blinded signature The process of generating a blinded signature s' for our blinded message m' now is similar to generating regular RSA signatures as in Equation 2.5:

$$s' \equiv (m')^d \pmod{n} \tag{2.8}$$

Unblinding blinded signature To retrieve a valid (blind) signature for our original message m from the blinded signature s', we can invert our blinding step from Equation 2.7. For this we need the inverse of our random value r^{-1} (which is easy to calculate). We determine the blind signature s for our message m by calculating

$$s \equiv m^d \equiv m^d r r^{-1} \equiv m^d r^{ed} r^{-1} \equiv (mr^e)^d r^{-1} \equiv (m')^d r^{-1} \equiv s' r^{-1} \pmod{n} \tag{2.9}$$

Hence we are able to recreate a valid signature from our signer's key pair, without learning their private key and without them knowing the content of our message. Note that afterwards our value r should be discarded to prevent our signer from ever getting to know what they signed.

The process of generating blind signatures can best be compared to the usage of carbon paper envelopes, as also explained in [15]: a document to be signed is packed in a carbon paper envelop, such that nobody can see its contents. The signer then puts its signature on the envelope, and the

carbon paper copies the signature to the real document inside. After that, the envelop is removed, and all is left is the original document with a valid signature.

The RSA cryptosystem, and especially the concept of blind signatures, form the basis of more advanced protocols and signature schemes for attribute-based credential systems we will see in section 2.3.

2.2 Zero-knowledge proofs

Another fundamental concept of attribute-based credential systems is zero-knowledge proofs. Zero-knowledge proofs are used to deliver a proof of knowledge of a certain value, without actually revealing that value. Zero-knowledge proofs generally are interactive in the sense that they consist of a challenge-response-like mechanism, in which a prover tries to convince the verifier that they know a certain value.

From a mathematical perspective, we use simple statements and require provers to deliver a proof for them. A statement could for example be $a \geq 18$, where a is somebody's age (in years). Proving this statement would generally be very easy, by just showing the value a (maybe $a = 21$). Everyone now can easily verify that indeed $a \geq 18$.

In practice, of course, such a proof itself is worthless. The knowledge of a such that $a \geq 18$ does not have a real meaning since anyone could come up with some value for a such that $a \geq 18$. For a proof like this to make sense in the real world, we do not want the user to decide the value of a, but we want the user to show that some other party (such as the municipality) provided the value a. This can be done with a signature. Essentially, for these applications, we do not want to prove knowledge of value a, but we want to prove possession of a signature on the value a.

As a first approximation, our statement would look like

$$a \geq 18 \quad \wedge \quad a \equiv s^e \pmod{n}$$

where s is an RSA signature of some municipality and (e, n) is the public key of that municipality. A proof for this new statement would now not only mean somebody *says* he has an age of at least 18, but also that that person knows a signature from the municipality on the value that *confirms* the age[4].

[4]Note that this example is an oversimplified view on what happens in IRMA and only serves the purpose of explaining how zero-knowledge proofs can be applied in bare essence. Among others, the identity of the verifier should be included in the statement and signature, as well as information on the validity and timestamps of the attribute.

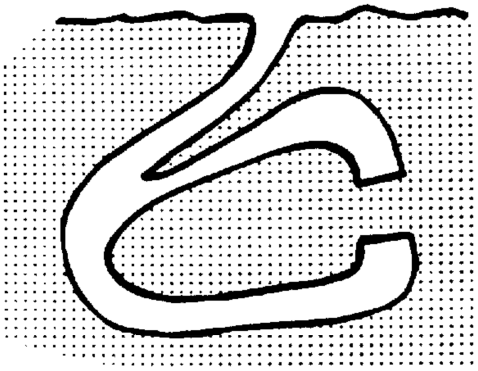


Figure 2.1: The layout of the special cave described in [42].

The straightforward way to deliver a proof for the above statement would be to simply disclose the values of a and s (and e and n) as well. The verifier now, however, learns more information than strictly required: it learns the exact age, not only the fact that the age is at least 18. Moreover, the verifier would also learn s, which he could then use to impersonate the proving party in other sessions.

To improve this, we can use zero-knowledge proofs. Those can be used to prove the above statement (or actually, instead of $a \equiv s^e \pmod{n}$, we can similarly prove $(a \geq 18) \equiv s^e \pmod{n}$, as is done in IRMA), while no more information than strictly required is revealed.

2.2.1 The Ali Baba cave

To explain the concept of zero-knowledge proofs, we will follow the popular example written down in [42], but with several adaptations to shorten the story and make it more insightful.

Consider two people, Peggy (for prover) and Victor (for verifier). They live in a town where there exists a special cave with two corridors that are secretly interconnected by a magic door (as depicted in Figure 2.1). The magic door is normally closed, but upon saying a secret magic word, the door is opened, and a passage is created between the two corridors of the cave.

One day Peggy tells Victor that she knows the magic word. Victor does not believe Peggy and therefore challenges her to prove her statement. The obvious way for this proof would be for Victor to join Peggy, go to the door, and see himself that Peggy knows the magic word. However, this would also reveal the magic word itself to Victor. Peggy does not want that. She wants to disclose as minimal information as possible: only show to Victor that she

knows the magic word, but without revealing the word itself or proving it to somebody else. Therefore Peggy will perform a zero-knowledge proof with Victor of the statement that she knows the magic word.

Instead of both entering the cave, first, Peggy will enter the cave and choose either of the two corridors (randomly). Next, Victor will enter the cave and wait at the entrance, so he can see both corridor exits, but not seeing Peggy in either of the two corridors. He will shout to Peggy the corridor he would like to see her leave the cave from. Now we will have the following:

- In case Peggy would indeed know the magic word, she can always satisfy Victor's request. If she had chosen corridor A, and Victor wants to see her leave from corridor A, she can just walk back (and naturally the same holds for corridor B). If Victor instead wants Peggy to leave the cave via corridor B, she can do so by using the magic word, using the secret passage and go to the other corridor.
- Consequently, if Peggy would not know the magic word, she can only satisfy Victor's requests 50% of the times.

If Peggy and Victor performed these steps several times in a row, Victor would become increasingly more certain that indeed Peggy knows the magic word, until the uncertainty would become negligibly small. This way, a zero-knowledge proof is performed.

Third-party observers One might think of an easier approach to the protocol above. At first sight, it would not be really required for Peggy to choose the first corridor at random. She could just tell Victor and Victor could see Peggy leave from the other exit, immediately knowing for sure Peggy knows the magic word. However, this approach does not satisfy the requirements. Namely, a third party observing Victor would then also be proven the fact that Peggy knows the magic word. Victor could secretly make a recording of the performed steps and use it to unfairly prove to others that he also knows the magic word (which is not true)[5]. By letting Peggy choose her entrance at random, any recording would become unconvincing, since it might well be possible that Peggy and Victor could be collaborating (we naturally do not want to just natively trust people in these contexts). The secret information that Victor does not know what entrance Peggy had chosen (which is a statement Victor can never prove to anyone without Peggy's cooperation!) is used to make the proof convincing only to Victor and not to anyone else.

[5]Here we reach the limits of this simplified analogy, but one should be able to imagine the problem here. A better analogy of this problem is given in [42].

A critical remark is that zero-knowledge proofs cannot be considered mathematical proofs in essence. This is because, as can be seen in the example, there will always be a tiny chance of luck. However, each iteration of the steps will increase confidence.

2.2.2 Examples of zero-knowledge proofs

There exist several mathematical methods for performing zero-knowledge proofs, which can be considered categories of proofs. One of those is the zero-knowledge proofs in the Fiat-Shamir heuristic [22], which are also used in the IBM Idemix attribute-based credential system [10, 45]. These proofs are based on multiplicative integers in a certain modular group of a prime number, and its security relies on the used hash functions under the random oracle assumption [40, 24]. The Fiat-Shamir heuristic can be used to create non-interactive zero-knowledge proofs, which, in contrast to the example given previously, do not rely on a challenge-response-like mechanism. We will not discuss this concept in-depth but rather assume that general systems exist to proof statements in zero-knowledge.

2.3 Attribute-based credential systems

Identity management is the process of identification and authentication, in order to authorize users' access to, e.g., a (digital) system. In classic identity management, this is done by users identifying themselves to an identity provider. Based on their identity, they are authorized a certain way.

Attribute-based credential systems take a different approach to this process. With attribute-based credentials (ABCs), users are not authorized after disclosing their identity, but rather by proving possession of attributes: tiny pieces of information that are not necessarily identifiable. By applying a cryptographic signature scheme, the ABCs are just as secure and trustworthy as an identity in a classic identity management system. A simplified overview of how this could work in bare essence is described in section 2.2.

Attribute-based credential systems (or anonymous credential systems) were proposed some time ago by Chaum in [16, 17] as an alternative to 'regular' credential system, based on fixed, unique identifiers, that offers little privacy. A definition used in [10] is as follows. "An anonymous credential system (...) consists of users and organizations. Organizations know the users only by pseudonyms. Different pseudonyms of the same user cannot be linked. Nevertheless, an organization can issue a credential to a pseudonym, and the corresponding user can prove possession of this credential to another organization (who knows her by a different pseudonym), without revealing anything more than the fact that she owns such a credential." A more formal treatment is given in [9].

The concept of these systems has been elaborated on in the past to create a system that has the desired properties [18, 19, 36]. These properties include, for example, unforgeability (the inability to create fake credentials), non-transferability or consistency (credentials are bound to one single specific user), and anonymity (credentials should not be traceable to a single person) and (multi-show) unlinkability (usage of a credential cannot be traced back to an earlier use of that credential) [16].

All-or-nothing non-transferability The property of non-transferability is rather subtle in practice since the property of anonymity is fundamentally in contrast with this. Because credentials cannot be linked to an individual, it cannot be prevented that users just fully share all their credentials with others. They could just hand over their credential wallet to a different person.

This can only be discouraged by the concept of all-or-nothing non-transferability [10], meaning that users can not partly 'lend' their credentials to another user, but only do it with all their credentials (or nothing). This is the best approach to discourage users from sharing credentials, hence to achieve non-transferability within an anonymous credential system. Another approach to achieve the same goal is to include some valuable secret from outside the system (such as having a secret key that also gives access to one's bank account, which is called PKI-assured non-transferability), which would work very similarly but has the drawback that such a secret is not always available [12].

2.3.1 IBM Idemix

Currently, two attribute-based credential systems, as defined in the previous, have been proposed: IBM Idemix and Microsoft U-Prove, the most successful one being Idemix. The Idemix scheme is described by Camenisch and Lysyanskaya in [10, 13, 12, 45] and its name is an abbreviation for *Identity Mixer*.

In Idemix, two parties are considered: users and organizations. The latter is again split into two types: issuing organizations (issuers) and verifying organizations (verifiers). Users can request credentials at issuers, who can issue those to a user.

Credentials are groups of attributes of a single issuer. During issuance, a signature from the issuer is generated and put on the credential. The attributes within these credentials can then be selectively disclosed to verifiers by the user. Verifiers can cryptographically check whether indeed the attributes are valid (signed by the issuer) because users deliver a proof of that signature

during disclosure. Nevertheless, the required properties for ABCs (described in the previous) are maintained. Disclosure of the attributes takes place in zero-knowledge, e.g., by the usage of a zero-knowledge proof. This way, only the minimal required amount of information is disclosed to the verifier, and the proof can, even if recorded, not be used by another user.

During issuance, this is achieved by using the special Camenisch-Lysyanskaya (CL) signature scheme [12], which uses zero-knowledge proofs and is a multi-party computation to (partially blindly) create the signature on the attributes that forms the credential. This way, users enjoy optimal privacy.

Note that the key difference of these systems with traditional identity management is that the verifying party will only talk to the user and not to the issuer, this way maintaining the privacy-enhancing properties.

Idemix protocols

The Idemix system consists of several protocols that describe the actions that can be performed within the system [13]. Without explaining the details, those are[6]:

1. U_registerNym: registers a user to the ecosystem
2. O_registerNym: registers an organization to the ecosystem
3. U_getCredential: a user requests a credential
4. O_issueCredential: an organization issues a credential to a user
5. U_showCredential: a user discloses a credential to an organization
6. O_verifyCredential: an organization verifies the correctness of the disclosed credential
7. O_checkDoubleSpending: for one-show credentials, a verifying organization checks double spending of the credential
8. DO_deAnonymize: a special de-anonymizing party can de-anonymize certain actions

The last two steps are special steps in the sense that they are optional.

One-show credentials The Idemix specification describes the concept of so-called one-show attributes. Where ABCs can, in general, be used multiple times and usage cannot be linked back to earlier usage, the Idemix system describes a special kind of attribute where multiple usage of the same credential can be traced back. After a credential has been disclosed to a verifying organization, the organization can check whether this credential

[6]The prefix 'U' stands for users, the prefix 'O' for organizations

was disclosed multiple times at the same verifying party. In [10], this is done by changing the disclosure protocols.

De-anonymization The other special protocol is de-anonymization. Idemix specifies an optional special organization called the de-anonymizing organization. This organization receives information upon user-registration in the system, which allows them to de-anonymize actions that took place in the system. They can do this by only receiving a transcript of the performed actions, so no interactive cooperation of the user is required.

An example of such an application could be as follows. A person entering a building can do so by authorizing anonymously using their credential(s) to the building manager. However, in case a crime was committed within that building, the police could, upon order of the court, de-anonymize the transcript to discover who was in the building at that time.

2.3.2 IRMA

The IRMA project is one of the real-world implementations of the Idemix scheme. It started as an implementation of the protocols on smart cards [47], hence with a major focus on efficiency of computations, but has now moved to smartphones [4]. The main objective of IRMA is to implement a rather complex Idemix system in a usable and simple ecosystem. With the application on smart cards, this was not only an ideological matter but also a technical requirement, since the smart card could not perform too complex or inefficient computations.

One of the consequences is that IRMA does not feature a full implementation of the system: several (cryptographic) features are omitted [4], among which the earlier mentioned concept of one-show credentials and de-anonymizing organizations.

The software stack that IRMA provides is rather simple: users can download the IRMA app for their phone and start using it right away. As an issuer or verifier, two components are required: the `irma.js` JavaScript library and the IRMA server. The `irma.js` JavaScript library can be embedded in websites with ease and will connect to the IRMA server that actually performs the IRMA sessions. This way, it is also possible to run the IRMA server at a third-party Service Provider [4], making the system even simpler to setup. Actions (issuance or disclosure/verification) are initiated by presenting a QR code to the user, that scans it using their mobile phone app. This connects the phone to the IRMA server to perform the steps as prescribed in the Idemix specification. For more technical information on the way IRMA works, we refer to `https://irma.app/docs`.

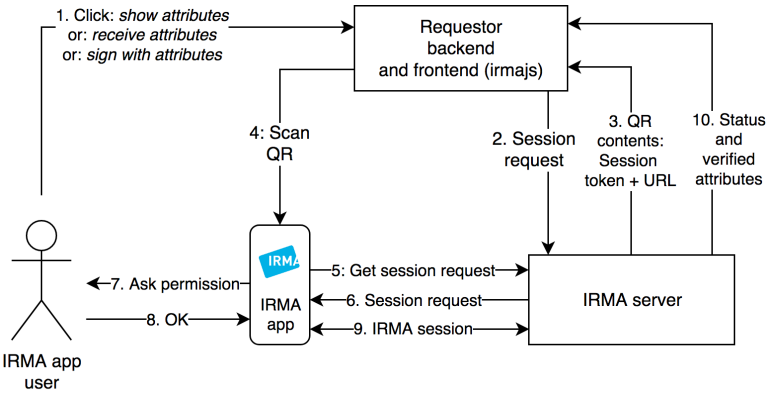


Figure 2.2: The schematical flow of a typical IRMA session [41].

The credentials specified in the Idemix scheme are kept in the IRMA wallet on the user's phone. Apart from the 'real' attributes, each credential also contains some meta-data attributes. These are just treated as regular attributes and just contained in the credentials as well, but are hidden from the user (but can be used in disclosure similarly to regular attributes). This involves, for example, an expiration date of the credential[7].

Another more specific example of a special attribute is the private key of the IRMA app (of the user)[8]. This private key is included in each credential to make sure that the credential is really bound to the IRMA app. Possession of the private key is proven in zero-knowledge in each IRMA session, naturally without revealing the private key itself. Upon issuance of a credential, a blind signature of the issuer is generated on all attributes in the credential (using the CL signature scheme mentioned previously), including the special attributes such as the expiration date and the private key of the user.

Keyshare

In attribute-based credential systems, in general, there is no central party involved with issuance and verification of credentials. Organizations only speak to users, and users only speak to organizations, not via a central party. IRMA, however, makes one exception for this. The foundation maintaining the IRMA ecosystem (the Privacy by Design foundation) maintains a keyshare server [41].

The reason for this lies in an essential observation for the IRMA ecosystem:

[7]Note that in practice to maintain anonymity, no exact dates will be chosen, but rounded to epochs of a week [41].

[8]In fact, it is not the full private key, as explained in section 2.3.2

we cannot consider the user and their app to be the same party. A user's phone might accidentally be hacked, making the app unreliable, or users might lose access to their phone. We must account for this. This is different from models such as Idemix that do not consider those practical problems.

To protect the credentials in the IRMA app on people's phones, therefore a short PIN needs to be entered before the credentials can be used. This way, loss of one's phone does not directly cause a problem. To truly securely realize this, the private key the user uses in all sessions is not stored entirely on the local device, but partly on the central IRMA keyshare server. The complete private key, required to perform all IRMA actions, can only be retrieved after a correct PIN is entered. This way, it is also possible to revoke an IRMA wallet entirely, by just telling the IRMA keyshare server not to allow sessions to take place if the wallet has been revoked[41].

Privacy-wise, this causes some challenges. Although it is inevitable to have the keyshare server know when we are using IRMA (since we must have contact with it), we still do not want to disclose anything else about our IRMA session. Moreover, the full private key that is required in each proof may never be shared with the IRMA app itself (since we cannot trust it), but may only exist in joint cooperation of the keyshare server and the IRMA app. The usage of the keyshare protocol realizes this.

From the IRMA web client (the online portal of this keyshare server), a user can now log in and see the usage of the IRMA key (all the times the user has logged in to the IRMA app) and possibly revoke it. Note that no information on whether or not a real IRMA action has been performed is revealed to the keyshare server, neither to whom.

Attribute based signatures

A new feature implemented in IRMA that is not mentioned in the Idemix scheme is the concept of attribute-based signatures. The way to realize this has been proposed in [26]. Attribute-based signatures are electronic signatures generated by a user that include a set of the user's attributes, instead of an identifier (public key) of the user, as is custom with classic digital signatures. Attribute-based signatures thus offer better privacy with the same integrity and verifiability. Organizations can request a user to electronically sign data, while not the identity of the user is included in the signature, but only certain attributes of that user. In practice, attribute-based signatures are very similar to non-interactive disclosure proofs.

After the signature has been created, the signature can be verified on whether the credential of the disclosed attributes within the signature was valid at the moment of signing it. Currently, the signatures can only be generated on statements (strings), but the attribute-based signatures could also be

generated on (pdf) documents or other types of data [41].

An example of an attribute-based signature could be a doctor's statement, such as a prescription. Whenever a doctor wants to hand out a statement, he or she will need to put a signature on the statement. Usually, such a statement would include an identifier for the doctor. However, this is not always required necessarily. Only the fact that the person writing the statement was a doctor can be sufficient in some cases. An attribute-based signature, using an 'IS DOCTOR'-attribute (issued by a national register of doctors) in the signature, could solve this.

2.4 Revocation and cryptographic accumulators

As described previously in subsection 2.3.2, credentials within IRMA contain several attributes as meta-data, among which an expiration date. These attributes, once issued, only remain valid for a certain amount of time. After that period, they become invalid, and users will need to renew the attributes (which is essentially nothing else than just requesting the same attributes again).

To revoke (the validity of) attributes in case a user loses its phone, IRMA features a keyshare server, that allows users to revoke a wallet. This only enables a whole wallet to be revoked, not certain specific credentials. Note that this form of revocation is performed at a different level in the system than the attributes itself. The accessibility of the private key used by the IRMA app is prevented, but in essence, the validity of the credential itself is not affected.

An issuing organization does not have any influence on the validity of already issued credentials. That is why credentials in IRMA typically have rather short validity periods.

If an issuer wants to have more influence on how long attributes are valid, the expiration dates must simply be set earlier. This is not only inconvenient to users (that need to re-request their credentials more often), but also, there will always remain situations in which even the shortest validity period is not sufficient. As an example, when having elections with IRMA voting passes (in any form whatsoever) upon death of a citizen, we want to revoke the voting pass directly. Waiting for the expiration date to expire is no solution if the fields of application become more sensitive. Hence, there might be cases in which it would be desirable to have credentials be revocable by the issuer, after issuance, but before the expiration date has passed.

Straightforward revocation

The classic approach to revocation is the usage of regular simple blacklists or whitelists. The straightforward approach would be to introduce an identifying number to each credential, where after each proof, the identifier should be compared to a blacklist (or whitelist) to verify whether the credential is still valid. Such an approach has only two major downsides:

1. Adding identifiers to the credential destroys the privacy-protecting features of IRMA
2. The approach is not easily-scalable since a list has to be kept, and for every proof, the whole blacklist or whitelist has to be iterated.

Although there are various techniques to solve the first problem (using zero-knowledge proofs) (among others, considered in [33]), the second problem remains must be solved as well to apply it in IRMA.

Alternative approaches exist, and their applications, for example, in IBM Idemix, have been showed often [6, 11]. This approach makes use of so-called cryptographic accumulators. Here, the effort of showing validity of credentials (showing that they are not revoked) is not put on the side of the verifier but on the side of the user. Instead of the user showing a credential-identifier that the verifier will use to check validity, users will prove that their credential is not revoked themselves.

Various optimizations have been proposed, currently allowing computations either be constant or linear in size of the number of elements to contain.

2.4.1 The basic principle of accumulators

The idea of accumulators was firstly mentioned in [7] as quasi-commutative one-way functions. This property can best be described as having a sort-of hash function (with both an initial value and input value) and a stream of input values we all want to compress onto an initial value using the function. Mathematically speaking, we describe this as $h : X \times Y \to X$ with $x \in X$ as initial value and $y_1, y_2 \in Y$ as our input values, such that

$$h(h(x, y_1), y_2) = h(h(x, y_2), y_1) \tag{2.10}$$

With this, we can create a single *accumulated* value z, which essentially contains a whole set of input values $y_1, y_2, \ldots y_n \in Y$ such that

$$z = h(h(h(\ldots h(h(h(x, y_1), y_2), y_3), \ldots, y_{n-2}), y_{n-1}), y_n) \tag{2.11}$$

where the order of each y_i can be changed without changing the value of z. As described in [7], this property can be use for "space-efficient cryptographic protocols for (...) membership testing" which is exactly our requirement for white listing.

Quasi-commutative one-way functions Several functions that meet the above state requirements exist. One of those examples is one we have seen in section 2.1 as well. The function of modular exponentiation in group n, $e_n(x, y) = x^y \pmod{n}$, is clearly quasi-commutative and can, with several restrictions, also be used as one-way function based on the RSA assumption [7]. Therefore this function can be used as primitive for building an accumulator (called RSA accumulators).

Apart from this, a wide variety of functions have been proposed, often with certain additional useful properties, of which an overview is provided in [20]. Those include, among others, dynamic accumulators (that support both addition and deletion of elements to the accumulated set) [11] and universal accumulators [34] that can be used for both membership and non-membership proofs [6]. This way, accumulators can be used for both efficient whitelisting and blacklisting. For simplicity, we will, for now, consider a whitelisting approach.

2.4.2 Accumulators for membership testing

To use the accumulator property for membership testing, we could associate all values y_i with identifiers [7]. Whenever a user y_j wants to prove membership of Y, he can do so by keeping a value z_i being the accumulated value of all $y_i \in Y$ with $i \neq j$, and then using that information prove that indeed $z = h(z_i, y_j)$. Whenever a user with identifier y' would try to forge membership, he would need to construct a x' such that $z = h(x', y')$, which should be hard (infeasible) for one-way functions [7].

Performing the proofs described in zero-knowledge would allow users also to hide their identity, therefore allowing for a zero-knowledge proof of membership. Such constructions have, among others, been proposed in [23, 6].

Note that one of the big benefits of this method is that each party does not need to maintain the identifiers y_i of all participants within the system, but only the accumulated value of the set. This is a big win on storage-complexity [7].

Scalability

As explained, accumulators solve the problem of storage-complexity for membership proofs. However, with classic accumulators, users would still need to compute accumulated values themselves, costing linear time. This poses problems with computational complexity when the number of members of a set becomes large. To solve this, improved accumulators have been proposed that allow for more efficient computations. One of those is called Braavos and is specifically designed for application in attribute-based credential systems with zero-knowledge proofs and anonymous revocation

[6, 11]. This optimized accumulator allows for efficient updates for both users and issuers, where updates only need to be performed at all upon revocation.

2.4.3 Applications in attribute-based credential systems

Revocation schemes for attribute-based credential systems, such as Idemix, have been proposed several times (both based on accumulators and other techniques). The IRMA ecosystem will implement revocation as well in the very near future, very similar to the proposals of [6, 11], using the CL-RSA-B accumulator (by Camenisch and Lysyanskaya)[9].

To realize revocation, each credential will contain an extra meta-data attribute, similar to the expiry date: the revocation value (or often called Anonymous Revocation Component, ARC, or witness) [9]. This revocation value will be an identifier for the attribute and forms the associated value for our accumulated set value described in the previous. The accumulated value will be publicly available to any IRMA user and verifier, published and maintained by the issuer as a non-revocation public key.

Upon disclosure, the user will proof in zero-knowledge that their credential is not revoked (we will use the accumulated set as blacklist and will, therefore, prove non-membership [6] of the revocation value). When a credential is revoked by the issuer, the ARC is added to the accumulated set, and a non-membership proof is not possible anymore. Hence we will not accept the credential. The exact technical construction of the accumulator guarantees these features [6].

This revocation scheme does require users to be online when disclosing an attribute since they will need to know the most recent accumulated value to generate a proof. For IRMA on smartphones, this is, however, a very reasonable restriction. Moreover, the computations to be performed are efficient and only need to be performed at all when the accumulated set value changes, which only happens upon revocation (so not continuously). This setup also makes revocation instantaneous.

[9]Note that different revocation schemes have been proposed, most notably the scheme in [35], which was designed especially for IRMA. However, since IRMA is currently developed for mobile phones and not for smart cards anymore, the benefits of this scheme do not outweigh the disadvantages compared to accumulators. This is why there is chosen for the approach of [6].

Chapter 3

Legal requirements for (public) elections

Before discussing a voting scheme that can be used to realize digital elections, we will explore the required properties such a scheme should feature. In this chapter, we will use a regulatory framework to draw up a set of requirements to satisfy. Additionally, we will directly identify certain recommendations for practical development of an election system.

3.1 Background on digital elections

As can be read in section 1.3 as well, digital voting has had a tumultuous history within the Netherlands. Other countries have also tried to introduce digital elections. As an example, Estonia is one of the leading countries that widely adopted electronic elections [27]. Although the country itself calls it a success [46], analysis of the schemes revealed that nevertheless, several security problems exist [5], those with the biggest impact regarding the guaranteed availability of the systems. This problem is fundamental (and not specific to only Estonia). As a conclusion, the recommendation is that electronic voting should never be the only way of voting, but rather an additional way, complementary to classic offline voting [5, 25].

3.1.1 Electronic voting in Estonia

As mentioned, Estonia is an example of a country that widely adopted electronic voting. The country was the world's first country to organize legally binding public elections digitally.

People can choose to vote both over the internet (online and remote) or at a polling station. The system, 'I-voting' (internet voting), makes use

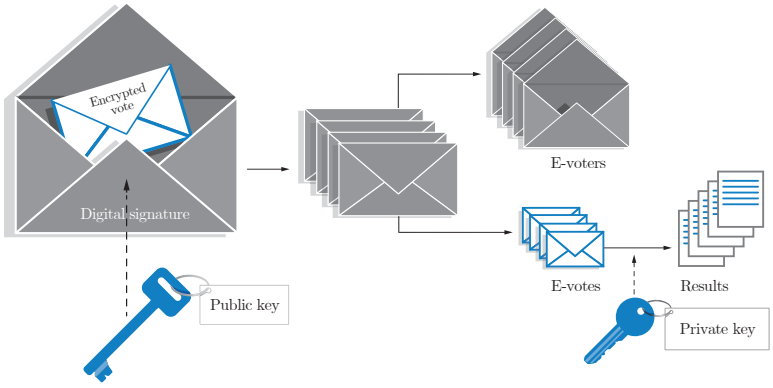


Figure 3.1: The graphical representation of the envelope scheme used in Estonian I-voting from [46].

of the Estonian national electronic identity system. From a regular computer (smartphones are not supported), users install special software to vote. Then, a form of electronic identity is used to verify the identity of the voter (for example, using a smartphone application that contains the Estonian 'Mobile-ID', or using a smart card reader and an Estonian identity card). When the vote is cast, people can also verify whether their vote was cast correctly. They can vote again to change the vote, or vote on paper (up to some days before the end of the digital election) invalidating their online votes.

The system in use in Estonia is a more or less straightforward envelope voting scheme, where votes are encrypted using a public key of the election and signed by the private key of each user (from their Estonian eID). Officials then verify that the identifying information in the signatures is removed from the content of the votes, before the private key (that is physically split under several individuals) is used to decrypt all votes. An overview is provided in Figure 3.1.

The system, however, does not offer privacy by design but relies on those organizational measures. Although users can verify whether their vote was cast correctly (they can request their vote from the server), the system is not end-to-end verifiable. Users cannot verify whether their vote is indeed counted correctly in the end. Estonian election officials are responsible for supervising that. Nonetheless, despite several lawsuits, the system is considered to sufficiently meet the criteria to make it legally binding. Note that in paper elections, end-to-end verifiability is also not trivial to achieve (but not impossible, we will discuss this further in subsection 3.2.2).

Additionally, it is denoted various times that, among others, the system is vulnerable to DDoS attacks [5], which is quite common to online elections

in general.[1].

The Estonian example is interesting to keep in mind when investigating how to practically implement a voting system, even when the fundamental underlying technology is different (as is the case with attribute-based credential systems). The system implements good-practices we will also discuss later, such as the ability to change a vote, maintain the ability to vote on paper, and provide understandable documentation for citizens.

3.1.2 Electronic voting in the Netherlands

In the Netherlands, in 2007, an advisory committee on the establishment of the election process published a report in which recommendations are made on whether and how to adopt electronic ways of voting [2] for public elections. The committee was established as a response to the problems with voting machines, as described in section 1.3. The report is considered one of the most important guidelines for electronic elections in the Netherlands.

3.2 Principles and requirements

When researching proposed (electronic) voting systems, it occurs that almost every proposal uses their own set of requirements and definitions their scheme satisfies or should satisfy. At first sight, this makes it seem that there is no general consensus on the required properties of electronic voting systems.

This, however, is not true. In fact, in-depth research has been conducted on precise principles and requirements of voting systems. We will discuss two of those frameworks. As we will see, although the exact way in which they are described differs, the underlying principles generally remain equal.

In 2002, a comprehensive framework was published [25, 37] that describes the principles and requirements for electronic voting systems. The principles directly find their origin in a regulatory model of elections, that is prescribed by law[2]. The requirements that can be found in Table 3.1, were stipulated by various panelists with expertise in the field [38]. The framework identifies six so-called constitutional requirements, from which a set of more specific requirements are derived.

[1]For more information on the i-voting system, we refer to `https://www.valimised.ee/en/internet-voting/internet-voting-estonia`

[2]Note that per country the exact legal requirements for elections varies a lot [28], but for the purpose of this topic, the fundamental principles the election laws of most (democratic) countries are based on, can be considered equivalent.

Constitutional requirements	**Voting systems design principles**
Generality	1.1 Isomorphic to the traditional
	1.2 Eligibility
Freedom	2.1 Uncoercibility
	2.2 No propaganda in the e-voting site
	2.3 Non-valid voting capability
Equality	3.1 Equality of candidates
	3.2 Equality of voters
	3.3 One voter - one vote
Secrecy	4.1 Secrecy
	4.2 Balance security vs. transparency
Directness	5.1 Unmonitored ballot recording and counting
Democracy	6.1 Trust and transparency
	6.2 Verifiability and accountability
	6.3 Reliability and security
	6.4 Simplicity

Table 3.1: Constitutional requirements and design principles [25]

Principle	Corresponding principles from Table 3.1
Transparency	*6.1, 6.4*
Verifiability	*6.2*
Integrity	*4.2, 6.3, 5.1*
Eligibility	*1.2, 3.2*
Liberty	*2.1, 2.2, 2.3, 3.1*
Secrecy	*2.1, 4.1, 4.2*
Unicity	*3.2, 3.3*
Accessibility	*1.1, 3.2, 6.4*

Table 3.2: Election principles from [2] and their correspondence with the design principles from [25, 37] (as given in Table 3.1).

Apart from this framework [25], the report by the Dutch advisory committee [2] also includes a list of principles, that mainly follow from principles of the Council of Europe. This list is very much in accordance with the other framework, but we consider it more general and usable. The principles of [2] are given in Table 3.2, together with information on how they relate to the framework of [25].

Since we consider the principles defined in [2] more general and usable, we will follow mostly these requirements in this research. In the following, we will briefly discuss those principles and focus on what their consequences are for application in attributed based solutions for electronic elections.

3.2.1 Transparency

To protect democratic values, the election process should be transparent so that any individual can understand the process and that everyone can be convinced the process is indeed democratic. No unanswerable questions may exist.

A problem that is often experienced regarding electronic voting is that the processes related are incomprehensible to many citizens. While in paper elections, the process is clear (anyone understands how paper ballots are recorded, counted, and processed), for complicated electronic solutions, there is the risk of losing this transparency. This also reduces trust in the election, which we will later (in section 3.4) see to be very important.

This property of transparency is somewhat fluid. For a large part, it relies on the property of verifiability, which we will discuss next, but it also involves more organizational aspects of the election that are not inherent to the underlying voting scheme.

General recommendations to improve transparency

Open-source software is a fundamental starting point to make an electronic process more transparent. However, still, this does not provide transparency to all users, since not every citizen can understand the software. In practical elections, understandable documentation should be available that precisely explains how the election is performed (and how this can be verified).

3.2.2 Verifiability

In an election, the outcome (and therefore, all intermediate steps) must be verifiable. As an example, for the outcome to be verifiable, it must be possible to recount all votes. Furthermore, it must be verifiable that all votes are correctly recorded (as intended by the voter) and that only eligible people have voted.

This principle can be conflicting to the property of secrecy, which should make it impossible for anyone to verify what some specific person voted. Nonetheless, the voter itself should be able to verify whether their vote has been recorded correctly, and anyone should be able to recount all votes.

Note that offline elections are also not fully end-to-end verifiable: citizens cannot check their own vote after it is cast anymore. However, this is not required, under the assumption that no paper ballots are lost or miscounted (which is something anyone can verify since all these steps are open to the public: anyone can observe the counting of votes if they want).

Paper trail These requirements have been a topic of debate in the past since, for fully digital elections, there is no guaranteed reliable way to realize them. We will always rely on specific software, and in practical large scale situations, it turns out to be very hard to verify the correctness of all software in use. It is mainly due to this, that the recommendation in [2] was always to have a paper-trail of cast votes, essentially eliminating the use of electronic voting machines and only allowing counting machines.

For online remote elections, however, a paper trail just cannot be implemented, which is why we need to rely on cryptography to realize this. Still, this requires us to trust the devices people use. This problem is fundamental to having an online remote election. We will discuss the consequences further in section 5.5.

3.2.3 Integrity

Integrity is very much in line with verifiability and prescribes that it should be impossible to change the outcome of the election in any way different than by casting legitimate votes [2]. Once a vote has been cast, it must not be able to be changed or revoked by anyone else than the user (which we will discuss further in subsection 5.1.1). On paper, this is achieved by using specific paper ballots. Online, again, this can be implemented with standard cryptography (using signatures). In contrast to the previous properties, this property is quite binary in its nature: either integrity holds or not.

3.2.4 Eligibility

In elections, only eligible people should be able to vote. This is the reason why in the Netherlands, the municipality is involved in elections, since this party knows which citizens are eligible[3].

[3]Historically, in the Netherlands, municipalities kept citizens records. Nowadays, this has largely been centralized to the central government, but municipalities remain responsible for the data, even though they do not necessarily own the infrastructure. For simplicity, we will just refer to municipalities everywhere.

In digital elections, this requires an identification step where one's identity is revealed at some stage in the process. This poses problems with the secrecy of the election. We will rely on cryptography to solve this problem.

Again, the nature of this principle is binary: either ineligible people are unable to vote or not.

3.2.5 Liberty

Liberty of an election entails that anyone eligible should be able to vote in all liberty [2]. This is very much related to the principle of secrecy that we will discuss next.

A property that is not explicitly mentioned in [2] but is in [25], is the possibility to vote non-valid. This means that, even if the content of votes is secret, it may not be possible by any means to enforce people to vote at all. Any system should allow for an option for a blanc or invalid vote that is not linkable to the individual.

For online digital elections, this principle mainly relies on the secrecy offered by the system.

3.2.6 Secrecy

Secrecy is probably the most prominent principle of elections. Although multiple subtly different definitions are used, we will consider the property as two-sided.

Not only should votes be anonymous, so that the content of a vote can by no means be linked to the individual, but also should the voter not be able to reveal his or her own vote [2].

Note that this is subtly very much stronger than just regular anonymity, which is just the first part. Especially following from the last part of the definition, users may not be able to prove how they voted, which is very difficult to realize. Among others in [25], this property is called uncoercibility.

Uncoercibility

We will give some more insight into the importance of the property of uncoercibility and why it is called this way. Uncoercibility means that, regardless of context, voters should not be able to be coerced to vote a specific way (or vote at all).

It is obvious that this problem can be very general and practical in its nature. This is because coercion could take place in various forms. For example, in remote (online) elections, regardless of applied cryptographic schemes, the very fact that people vote from home means they could be

coerced by, for example, their relatives. In this sense, the property is very much related to the previous property of liberty. Here, we also identify one of the fundamental problems of online remote elections, in which this kind of coercion is practically inevitable to prevent entirely. In subsection 5.1.1, however, we will discuss what (practical) measures we can take to make some improvements.

However, coercion could also take place in different forms. Think, for example, of people selling their vote or being (either positively or negatively) discriminated based on what they voted. This kind of coercion has a more fundamental background and comes to play in large scale applications. This problem could, however, be solved by a new property that is fundamental to a voting scheme: receipt-freeness.

Receipt-freeness

Receipt-freeness generally means that it must be impossible to determine what someone voted, even with cooperation of that same person. More specifically, voters must not be able to deliver a proof of what they voted [21, 8].

In offline elections, polling station staff are required to supervise that people will only enter the voting booth alone. Because only a single copy of a ballot is provided[4], citizens do not receive a so-called 'receipt' of what they voted. This way, even with cooperation of the voter, there is no guaranteed way to prove what somebody voted, therefore guaranteeing the freedom of the election.

Ballot selfies This is also the reason why no cameras are allowed inside the voting booths, and why in the Netherlands recently there was discussion about *'stemfies'* (people taking a photo of themselves voting inside a voting booth)[5]. Such a photo could be considered a proof of a vote, making the person coercible. Note that to really be problematic, such a 'receipt' must be absolutely convincing. In case, despite the receipt, there could be a way for a voter to have voted a different way than indicated by the receipt (maybe because the voter could forge a fake receipt), the property is still intact. The receipt is, in that case, reduced to the same level of trustworthiness as an oral statement of what someone voted.

[4]In the Netherlands, citizens can request a new ballot once if they made a mistake on the sheet, which is only provided after the first ballot is destroyed.

[5]The *'stemfie'* (from the Dutch word for 'voting', *stemmen*, and 'selfie') was introduced on social media by people to motivate people to take part in elections but was also considered a violation of the election law because people were proving what they voted, therefore affecting uncoercibility. The electoral council finally decided that *stemfies* were allowed, among others, because people can request a new ballot. Therefore, a mere *stemfie* could not be considered a full receipt that could be used to prove what someone voted.

Scalable attacks on online elections In online remote voting systems, the problem of receipt-freeness is much more prominent than in offline elections. In offline elections, it is just impossible for voters to make a copy of their final ballot. In online voting, however, every data package sent can be recorded, and every computation can be logged. Moreover, due to the increased exposure of the election environment (the world wide web), attacks can easily be scaled up. For attackers, it becomes easy to target large groups directly.

Think of this example: with online digital ads, people might be coerced to download a certain tool that monitors them voting a specific way to receive an amount of money as compensation. In offline elections, this 'business model' would be infeasible for attackers because of practical reasons, but in online remote elections those boundaries disappear.

We have seen that the principle of secrecy has many sides. For our scheme, we will generally consider only anonymity, which again is a more or less binary property. Receipt-freeness of our scheme will be considered more in-depth in section 5.1 and 6.1. Other forms of coercion, that are more related to the principle of liberty, will also be discussed, but they are more fundamental in their nature.

3.2.7 Unicity

Unicity of elections means that any eligible person can only vote at most once. However, still, votes should remain anonymous, so we cannot just allow linking one's identity to the vote to verify this. Again the nature of this property is binary: we cannot allow for any way to have people vote multiple times.

To realize this, a system with voting passes must be in place to ensure people can only vote once. In the previous chapters, it has been explained why this is easily achievable in classic identity management, but is rather difficult in attribute-based systems with unlinkable credentials, especially in practical situations where people can lose their voting credentials.

3.2.8 Accessibility

The principle of accessibility is largely similar to what [25] describes as generality. Anyone should be able to take part in the election.

This principle can put requirements on, for example, the user interface for citizens, that should be easy to understand. Think of using simple language, but also the use of color-schemes friendly for people suffering from color blindness. Moreover, the system should be accessible to people using differ-

ent operating systems on various devices, regardless of their internet speed. The Estonian I-voting example discussed previously (subsection 3.1.1), is an example to consider here.

As can be noted, there are numerous cases to account for. In fact, it is impossible to have such a system that every individual experiences the same threshold to vote (in traditional paper elections, this is not the case either). Nonetheless, is it essential to make sure the system does not exclude specific significant groups of people from the election because that could significantly impact the outcome of the election.

An example of a group to consider for online elections is maybe (but not solely) the group of elderly people that perhaps do not have a (reliable) internet connection or device to use. Nevertheless, other groups are also important to consider. As an example, the municipality of Rotterdam in 2016 aimed to organize a digital referendum, somewhat similar to the Amsterdam OpenStad project (section 1.2). However, the plan was canceled because too many people were assumed not to have an activated DigiD (national governmental electronic identity), which was required to vote.

Maintaining a paper alternative To guarantee the accessibility (or generality) of the elections, a strong conclusion presented in [25] is that there should always remain an alternative way for people to vote on paper with relative ease. Moreover, [25] concludes that the paper alternative should not just be offered, but should also be *fully equal* to the online method, in the sense that the period in which people should be able to vote, should not differ between the methods since this could also influence the election outcome. In practice, however, this last requirement is rarely implemented.

Additionally, measures must be in place to ensure unicity when people try to vote both on paper or online. Depending on the exact implementation of the scheme, this can, however, be rather easy to realize.

Simplicity Apart from these direct forms of accessibility, there is also a more indirect form of accessibility, related to simplicity. In order for an election to be accessible, the process must be simple enough for people to understand. For example, when a system involves difficult application of cryptography that a user has to be actively involved with, the accessibility of the system fails. This is also closely related to the principle of transparency.

Voter registration Another thing to keep in mind is the concept of voter registration, which also differs per country. In most countries, citizens do not actively need to register themselves to vote, but this happens automatically: in the Netherlands, every citizen receives a paper voting pass per mail and hence are registered automatically. This is not the case in all

countries, however. Certain countries do require citizens to actively register to vote anyway. It is important to remark that voter registration in online elections might drastically decrease accessibility for countries that have no active voter registration phase classically since it requires users to perform an extra step. As we will see, this voter registration is also essential to the scheme we will present.

As can be seen, the consequences of accessibility in election processes are significant, even without thinking about digital security itself (think of (D)DoS attacks on the systems used in the election).

Online voting for increased accessibility It might seem that we should consider digital remote elections just as a rather inaccessible concept since there are so many things to account for. It is, however, important to remark that for many people, online voting should actually be considered a way to make elections *more* accessible. The fact that we do not need to go to a polling station physically could be considered a big win. This conclusion is also supported by, for example, [49], that states that widespread criticism on electronic voting has "unfairly concentrated on the associated risks and neglected the benefits", such as the increased accessibility.

We see that for elections people find not relevant enough (in the Netherlands, the provincial states and water board elections are a good example of so-called *second order elections*), people do not want to take the effort of going to the polling station, while it is likely that they would cast their vote digitally if it could be done with ease [1]. This is precisely why, for small scale digital elections, it could be desirable to use digital elections.

3.3 Key-principles

In the previous section, we have seen a general framework of properties election systems should satisfy. As seen, some principles put requirements on the fundamental schemes that are used while other principles are more circumstantial. The fundamental principles we have seen are eligibility, unicity, integrity, and secrecy (and especially anonymity). These are properties that generally are considered binary: they either hold or do not hold, there is little in-between. Principles such as accessibility, transparency, liberty, and perhaps also verifiability, on the other hand, can be considered dimensions on a more gradual scale. They rely on the context and practical implementation of a scheme in context to the election.

For the scheme we present, we will mainly deal with the fundamental properties, since the other properties are more dependent on the specific use-case.

3.4 Trust

In the previous sections, a regulatory framework has been presented about the way (electronic) elections should be organized. To have legally valid elections, these are the fundamental requirements to be satisfied. It is, however, important to note that not every election is necessarily bound to the Dutch election law. Therefore, all those properties are not always enforced. Especially in the kind of elections as mentioned in section 1.2, we could allow violations of the above framework.

As an example, one could argue that it is desirable to maybe make sacrifices on secrecy to increase accessibility. This could be an argument to allow ourselves to have a system partially vulnerable to certain forms of coercion in small scale digital elections. Legally we could be allowed to do this, and depending on the context, this might be a fair decision to make.

Nevertheless, it remains important to follow the principles as close as possible to increase trust in the election process. History teaches that the success of digital elections is not merely a case of legal compliance, but also of trust in technology. Examples are given in section 1.3 and among others in [2, 30].

Whenever citizens might discover that, for example, it is easy to vote twice in some online system, because the principle of unicity has been sacrificed to increase accessibility, the trust in the outcome and the underlying system can easily evaporate, causing a system not to be adopted. Again, criticism on electronic voting systems could (maybe unfairly) influence the adoption, regardless of the offered benefits [49]. We have seen a similar thing in with the earlier mentioned plan of the municipality of Rotterdam, aiming to organize a digital municipality-wide referendum.

As a conclusion, we must remind that systems should perform correctly to gain trust.

Chapter 4

Anonymous voting schemes using ABCs

Digital election systems can be used as a more accessible and possibly cheaper alternative to paper elections, especially for small scale elections. Specifically, we consider the type of digital elections in which people can vote themselves online and remotely, for example, from their smartphone. The IRMA framework provides a privacy-friendly yet simple and user-friendly ecosystem that can be used in (among others) this context. As we will see, however, regular IRMA sessions cannot natively be used for this purpose. Therefore, in this chapter, we present a scheme that consists of several IRMA sessions that realizes a scheme that satisfies most desired requirements for elections, as discussed in section 3.2.

Our scheme in other attribute-based credential systems In a more general sense, ideas used in the scheme we present might be partially applicable in other attribute-based credential systems as well. Note, though, that in certain systems, features that are provided with this scheme might exist by default already, in which case our scheme has less added value in the functionality it provides. As an example, in systems that natively implement the concept of one-show credentials, it could be more elegant to build upon those, than to strictly follow our scheme (that is based on IRMA that does not natively support one-show credentials)[1].

Looking into the future The goal of the scheme is to present a way in which digital elections can be organized using the IRMA framework. It is not the intention to make fundamental changes to the IRMA system.

[1] A critical remark is that usage of attribute-based signatures is not supported by many systems, although it forms the basis of the solution we present.

Nonetheless, specific steps in the schemes we will present, rely on features that at the moment of writing are not yet implemented in IRMA. The feature of revocation, which is required to provide users that lost their voting pass with a new one, is a feature that is currently under development and will be released within the foreseeable future at the time of writing. The concept of blind double signing of attributes or blind generation of a vote identifier is also not yet supported. However, as we will see, the changes required to support this are rather small since the concept is already applied to a special part of the signature of each credential. The implementation could be performed with relative ease in the foreseeable future.

Phases in the scheme Our scheme will consist of two distinct phases: voter registration and vote casting. As we propose it, we will require the phases to be performed entirely separately: after the first vote has been cast, nobody can register to vote anymore. In section 5.2, however, we will see that this requirement can actually be made less restrictive.

Both phases depend on an IRMA session: one for issuance and one for disclosure (more precisely: signature generation).

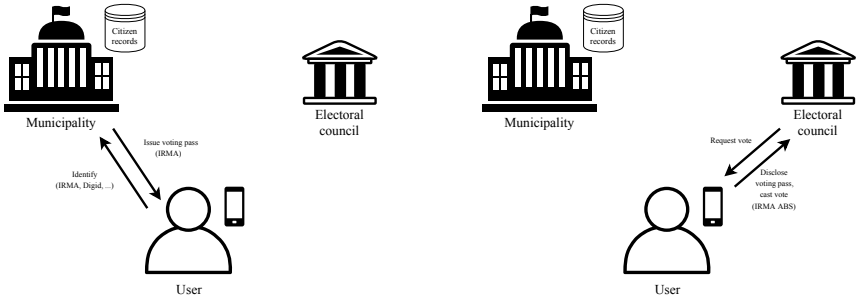
First, we discuss the parties involved in this scheme. Then, we consider both phases in more depth separately. We first focus on the vote casting phase, which forms the core of the election. Next, we will discuss the requirements for the registration phase (that will actually be performed prior to the vote casting phase). Here we will first present some obvious partial solutions that will turn out to not be entirely correct, and use these examples to present more specific requirements for a correct 'full' solution. Finally, we schematically present an overview of the scheme as a whole.

Additionally, in chapter 5, we discuss the security and limitations of the scheme and propose additions to the scheme that further improve it.

4.1 Involved parties

Our scheme considers several parties that are involved with the election process, similar to traditional paper elections. Apart from regular users, who are the citizens that cast their votes, we consider the electoral council and the municipality. The municipality, traditionally, is responsible for the distribution of voting passes to citizens. The electoral council is involved with the election itself: it verifies eligibility on information provided by the municipality and makes sure the votes are recorded and counted correctly.

In our scheme, those parties will have similar roles. The municipality will still decide on eligibility and hence be an issuer of attributes of eligibility. For this, users will need to disclose their identity, either by DigiD or accepted



(a) Municipality issues a voting pass to user upon identification

(b) Electoral council verifies voting pass and user casts vote

Figure 4.1: Communication between user, municipality and electoral council in IRMA voting scheme.

identifying IRMA attributes[2] (Figure 4.1a). The electoral council, on the other hand, will maintain the infrastructure that provides eligible users to vote. Therefore the electoral council will be the verifier of the eligibility attributes (Figure 4.1b) that authorizes users' access to the voting system.

4.2 Casting a vote as attribute-based signature

The central concept of our scheme is the way we cast votes using attribute-based signatures (ABSs). ABSs, as explained in section 2.3.2, are a feature of IRMA that allows for the creation of digital signatures on statements that, instead of containing the public key of the signer, contain their attributes. Here we build upon [26].

ABSs are the perfect concept for application in the context of elections. This is because of several reasons:

- Integrity of votes is guaranteed: the digital signature guarantees that the vote cannot be changed.
- Verifiability and transparency of votes: the digital signatures are created by the IRMA app on a user's phone. The user does not need to trust the verifier's software to know whether the vote is created correctly: the vote statement is unambiguously shown to the user by the already trusted IRMA app.
- Verifiability of outcomes: since the votes do not include an identifier of the signee, but only their attributes, when applied correctly, all votes can be published publicly for anyone to see, and be (re)counted by

[2]The choice to make here depends on the strength of the authentication. For the reliability level 'high', there are more requirements than for the reliability level 'substantial'.

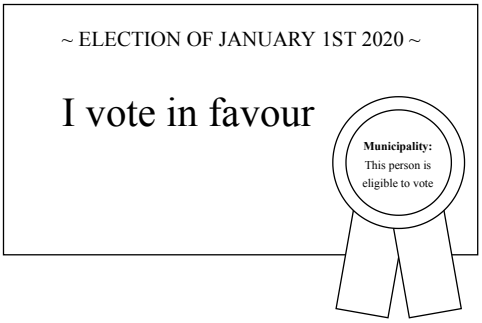


Figure 4.2: A graphical representation of a first approximation of an attribute based vote.

anyone to verify the election outcome.

- Secrecy: when the attributes disclosed in the ABS are non-identifying, the signature cannot be traced back to the user.
- Simplicity and accessibility: the implementation in IRMA makes sure that the user does not need to perform difficult election-specific key-management, but rather the IRMA private key, which already exists, is used. By using the IRMA ecosystem, we hook into a general system that is not election specific but has other use cases as well.

Vote statement A cast vote in our scheme consists of an ABS on a vote statement. This can be a statement as simple as "I VOTE 'YES' IN THE REFERENDUM, ORGANIZED BY THE MUNICIPALITY OF AMSTERDAM, HELD ON JANUARY 1ST, 2020, ON THE STATEMENT WHETHER THE TOWN SQUARE SHOULD ADOPT MORE TREES."[3]. This statement, together with disclosed attributes (the 'voting pass'), contains all required information for a vote. Which attributes are exactly required to form this voting pass, will be discussed next in section 4.3.

Requesting an attribute-based vote When voting, users will access the software, presumably a web page, of the electoral council and select the option they wish to vote for. Upon doing so, they will push a button to initiate an IRMA session to generate an attribute-based signature of their vote, disclosing the attributes in their 'voting pass' that authorize them to vote indeed. Next, the user verifies and confirms their vote on their IRMA app. The electoral council then also verifies the correctness of the

[3]Although for optimization of implementation, the form of the statement can be standardized and simplified, it is recommendable to keep the exact statement unambiguous and clear, such that it is easy to understand for any voter. This to improve transparency and simplicity of the election and possibly also prevent man-in-the-middle attacks that could trick the user into signing a different statement in the IRMA app as intended.

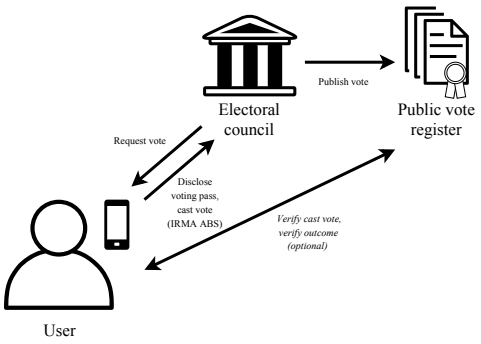


Figure 4.3: The electoral council publishes votes in a public register, allowing users to verify their vote.

ABS (checking eligibility and unicity) and publishes the vote in a public register[4].

Optionally the user now, or later when the election is over, also verifies whether their vote has indeed been correctly added to the public register (Figure 4.3) before removing their 'voting pass' attributes from their app.

Correctness The approach of recording votes as attribute-based signatures offers several benefits. Most notably, it allows for perfect verifiability. Since votes are stored in a public register, anyone can verify the election outcome themselves. Users themselves also can verify whether their own vote has been recorded correctly in the public register. Nevertheless, as long as the attributes are not identifying, the votes remain anonymous. Furthermore, because attributes and signatures are unforgeable, the integrity of the election is also maintained: a vote can only be created by the IRMA app of a user.

4.3 Voting attributes

By using ABSs for our votes, we cover a large part of the verifiability and integrity of votes. The challenge remains to guarantee secrecy, eligibility, and unicity. As seen previously, this will be achieved by the attributes we disclose in the ABS. We must determine which attributes to use to form this 'voting pass'.

Secrecy [5], or in IRMA more specifically called multi-show unlinkability, is a core feature of IRMA and is therefore easily achievable: it will be achieved as long as we do not include identifying attributes in our voting pass. The

[4]In this research, we do not consider the way such a public register could be maintained.
[5]We do not necessarily follow the full definition from subsection 3.2.6.

challenge is thus reduced to finding attributes that guarantee eligibility and unicity while *maintaining* secrecy. In other words, as long as the voting attributes in our ABS do not identify the user, we do not have to worry about loss of secrecy in any other way.

4.3.1 Partial solutions

To gain a better understanding of the problem, we will first present several approaches for attributes to be included that do not satisfy all desired properties.

Just using attributes of eligibility

A first idea to build upon would be to just include attributes for eligibility in the 'voting pass'. For example, if we would allow only adults living in the municipality of Amsterdam to vote, we might want users to disclose the attributes Over 18 and Hometown. These attributes not even need to be issued specific to an election and could, therefore, fully eliminate the need for a voter registration phase.

Because the attributes are non-identifying, this approach would satisfy secrecy (and naturally eligibility). However, unicity is not satisfied. Because IRMA credentials are (multi-show) unlinkable, each eligible voter could vote multiple times without this getting noticed.

Requiring a voting number

It becomes evident that we need to give each eligible voter a unique voting number to be included in their vote, which allows us to check whether they have already voted. Therefore, we could consider election specific voting numbers being issued by the municipality, additional to the other attributes for eligibility. We would require disclosure of this voting number in the attribute-based vote, so if anyone would vote twice, we could notice this in the public register (since one voting number would occur twice).

Now we would satisfy unicity. However, this makes it an obvious violation of secrecy: the municipality issues a voting number that identifies the user. Because the municipality knows the identity of the voter, they could link the vote to the person.

A voting number from a different party

To solve the problem of identification based on the voting number, we cannot accept the voting number being issued by the same party that decides on eligibility (or more specifically, not issued in the same session), because that party is also required to know the identity of the user.

Therefore we could think of issuing the voting number as a separate credential. This, however, does not solve the problem of unicity. Fraudulent users could easily request multiple voting numbers and still vote multiple times. Because we cannot allow for the user's identity to be disclosed while requesting voting numbers, this would go unnoticed.

4.3.2 Observations

By now, we have identified several requirements for the voting number:

- The voting number may never be linkable to the user's identity.
- No two users may possess the same voting number.
- No user may possess more than one voting number (in essence).

Additionally, the voting number should be specific to each election for obvious reasons (which is why we need an election-specific voter registration phase).

We have seen that it is not straightforward to meet all requirements simultaneously, especially the first and last one. Either we link the voting number to the user, violating secrecy, or we allow users to vote multiple times.

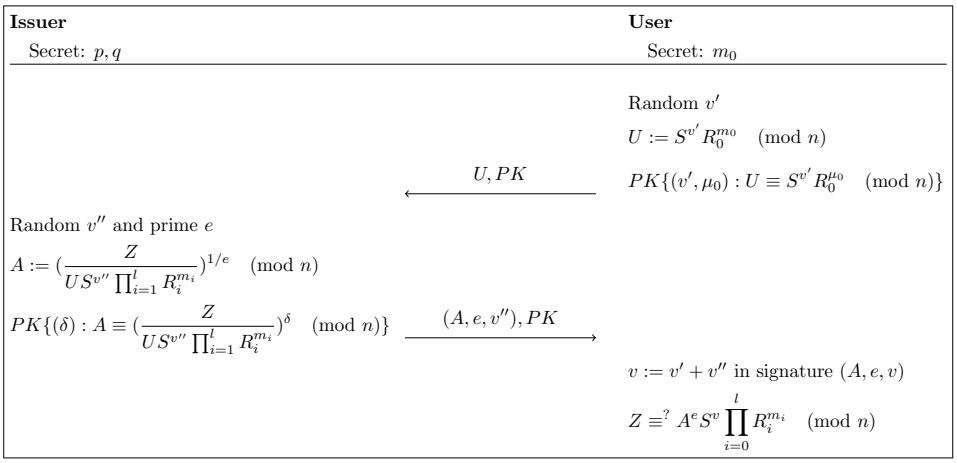
To solve this violation of unicity, we must not consider attributes for unicity and eligibility separately, but we will require the attribute of eligibility to be (cryptographically) bound to a voting number. Only together can they form an accepted 'voting pass'. This way, even if users would acquire multiple voting numbers, they can only use one to vote, because only one is contained in the credential. This must, however, be done blindly in zero-knowledge, so that we cannot link the voting number back to the user's identity.

There are several subtly different ways in which we can realize this, that we will now discuss.

4.3.3 Blind double signing the voting number

One approach to realize a direct binding with a voting number would be by having the municipality create a digital signature on a voting-number attribute. Nonetheless, to protect secrecy of the votes, the municipality may never learn the voting-number (also not in hindsight). To achieve this, we can use blind signatures.

Within IRMA, this is in fact already implemented in some form. In regular IRMA sessions, we have seen that during issuance, the credential (which is basically a signature) is blindly bound to the private key of the IRMA app of the user. This is because we do not want to allow a credential issued to one app to be usable by another app. The exact value of the private key, however,



Scheme 4.1: Simplified description of credential issuance in Idemix [45], as given in [3]. m_0 is the private key of the user's app. m_1 to m_l are other attributes to be issued in the credential. S, Z and R are system parameters derived from the issuer's public key $n = pq$ ($S, Z, R_0 \dots R_l \in QR_n$ with QR_n being the multiplicative subgroup of quadratic residues $QR_n \subset \mathbb{Z}_n^*$).

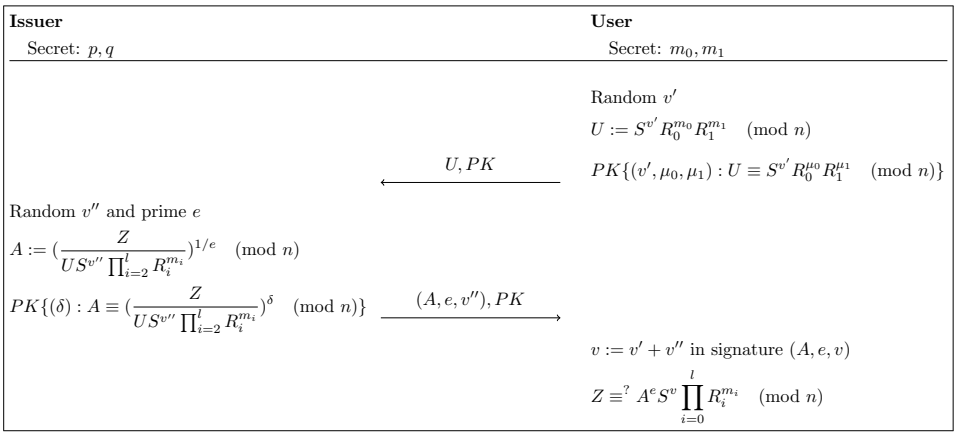
is naturally never disclosed, so this is done in zero-knowledge and with blind signatures (using the CL-signature scheme). In basis, however, the private key just remains a (special) attribute of the credential. A simplified overview of this scheme [45] is provided in Scheme 4.1 [3].

We can use this very same concept to include a second special attribute in the signature as well: the voting number. Similarly, this would disallow usage of the eligibility attributes together with a different voting number.

To do this, we assume extended IRMA protocols for issuance and disclosure. We will not only have m_0, the private key of the user, be a user secret in the credential issuing protocol, but also have m_1 (or possibly multiple m_i) be secret (or similarly, have m_0 consist of multiple values). Hence, during issuance, not only the private key is blindly contained as an attribute in the credential, but also other attributes (from other issuers) can be blindly contained in the credential. A way to implement this change is given in Scheme 4.2.

This could be considered as blind double signing of an attribute (since the attribute will be signed by a second issuer, the municipality, as well, but blindly).

During disclosure, not only a zero-knowledge proof is performed on the credential belonging to the user's private key, but also of the possession of the other attributes (in this case, the voting number). During disclosure, how-



Scheme 4.2: Extended credential issuance similar to Scheme 4.1 where, apart from m_0, also one other attribute m_1 (possibly issued by another issuer to make a blind double signature) is blindly included in the signature.

ever, it is not required to hide m_1. Instead, we will require disclosure of the attribute twice, from both credentials it is contained in: both the original credential and the additional credential that is signed by the municipality. To have a valid voting pass, both values should be equal.

For our voting scheme, such a system would divide the voter registration phase into two steps: retrieval of the voting number and retrieval of the eligibility attribute, that is bound to the voting number (think of this as an activated voting pass). This is because users need to identify themselves while requesting eligibility attributes, but we cannot allow users to be identified when retrieving a voting number.

Note that it is undesirable to have both phases take place simultaneously, since the timing of the two actions, can be used to make the voting number (that is not hidden in the first step) linkable to the user's identity (disclosed in the second step).

Retrieving a voting number

Retrieving a random voting number can be done in various ways, that are all straightforward. The only requirement for this process is that no number is issued twice.

For this, we assume voting numbers being issued by a central party. This could be, for example, the electoral council. They will issue unique voting numbers as IRMA attributes to anyone without requiring any form of

authentication. Their only purpose is to make sure the numbers are not colliding [6].

Retrieving eligibility attribute

When a user has a voting number retrieved, they can go to the municipality to get an 'activated voting pass': a credential containing attributes of eligibility (in any form whatsoever), cryptographically bound to the voting number. They disclose their identity (using either DigiD or accepted identifying IRMA attributes). Based on that, the municipality decides on eligibility.

In case the person is indeed eligible, the municipality issues the eligibility attribute using this special issuance protocol, where the voting number attribute is blindly included in the eligibility attribute (just as the IRMA private key). This way, the municipality actually blindly puts a second signature on the already issued attribute.

Now, the user has obtained a credential that contains a unique voting number that is not known by anyone (except for the initial issuer, but they do not know the identity of the user), which we can think of as an active voting pass. They can use this to vote.

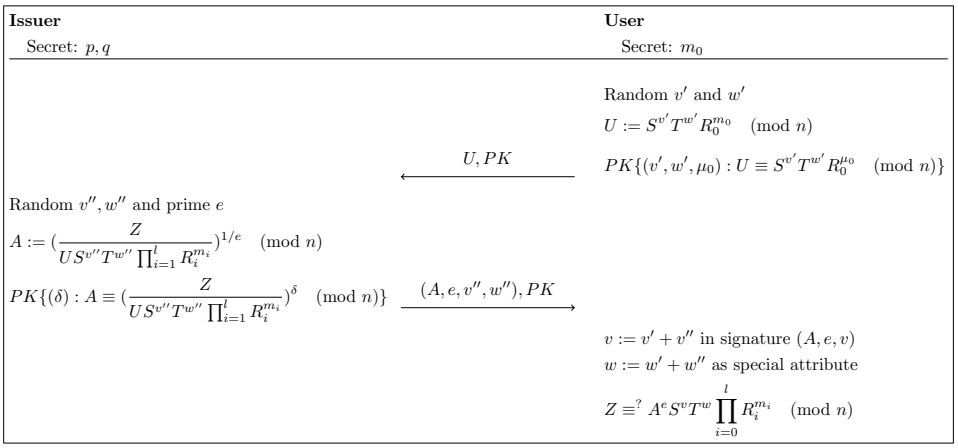
4.3.4 Blind voting number generation during issuance

We have described how the municipality can create a signature on a voting number attribute from a different issuer. It is, however, not necessarily required to have the voting number issued as an attribute at first: it only should be included in the final attributes we will disclose in the voting ABS.

An alternative scheme that accomplishes this as well generates a voting number in a different way and relies more specifically on the signature scheme used during the issuance of attributes. Moreover, it does not require separate issuance of the voting number.

Similar to the previous approach we presented (where we proposed to have multiple messages m_i to be signed blindly), we could also use a similar approach as the random value v that is generated during issuance, as presented in [3] and Scheme 4.1 (in Idemix, this is called a validating tag [10]). This value is constructed from two numbers: v', randomly chosen by the user, and v'', chosen by the issuer. The result is a random value included in the

[6] In fact, the voting number need not even to be issued as an IRMA attribute, but could also be generated at random by the user itself, as long as the number is large enough not to cause collisions. However, organizational measures should then be implemented to handle situations in which we do find collisions. Note, however, that this way, fraudulent users might *deliberately* try to create collisions, which could harm the system.



Scheme 4.3: Extended credential issuance similar to Scheme 4.1 where, apart from validating tag v also one credential identifier w is blindly generated. We additionally use $T \in QR_n$ as system parameter.

credential (and with large enough numbers also unique to the credential), but unknown to the issuer[7].

To achieve this, we will perform the steps that are used to generate the random value v in duplicate. Instead of the user randomly choosing v', they will both randomly choose v' and w'. The issuer also randomly chooses v'' and w'', calculates the blinded signature according to the CL signature scheme (but with w'' included as well), and the user unblinds it while constructing both $v = v' + v''$ and $w = w' + w''$. w can now be used as a voting number. Moreover, in contrast to v, w should not be hidden but should be disclosed when creating the voting ABS. This way, the credential will become one-show (per verifier): similar to the scheme of subsection 4.3.3. each verifier can record the disclosed values w and can verify whether it has been disclosed multiple times. Nevertheless, the issuer is not able to link w to the user's identity since it did not learn w, but only w''. This is presented in Scheme 4.3.

4.3.5 Blind signatures in the attribute payload

Both solutions of subsection 4.3.3 and 4.3.4 are fully functional, but do rely on features not yet supported in IRMA (although the required changes are minimal). As an alternative, we present a detour to achieve the same goal. This approach could be used to create a very bare proof-of-concept of the

[7]Note that the value is not fully decided on by the user since it contains a random component of the issuer as well, so deliberately creating colliding voter numbers is not possible here.

system.

Instead of cryptographically binding the credential containing the eligibility attribute to a voting number, we could also include this binding in the payload-value of the otherwise empty eligibility attribute itself. Using regular blind signatures as described in subsection 2.1.4, the municipality could create a blind signature on the voting number, outside any IRMA session and only then issue it using regular IRMA issuance.

This way, we basically elevate the blind signature on the voting number from the credential (generated in an IRMA session) to the attribute payload (generated outside IRMA sessions), which achieves the same goal: the eligibility of a user is bound to a voting number, which we can verify by using the content of the attribute.

For our scheme, this would mean the following: we assume the user already retrieved a voting number in any possible way. When retrieving an eligibility attribute from the municipality, they again identify to the municipality. When identified, they interact (in a not-IRMA session) so that the municipality generates a blind signature on the voting number. That signature is then issued as the payload of an IRMA attribute for eligibility to the user. Thus, the value of the eligibility attribute will now be the encoding of a signature on the voting number. When voting, in the ABS, the voting number and the signature are again disclosed.

Using this protocol, it is possible to perform the same actions as in the previous proposal. As can be seen, however, this protocol is quite complex, and it requires a whole number of tricks to correctly and securely perform all steps in practice. As an example, we cannot allow the municipality to see the unblinded signature as it would violate secrecy. Therefore, the unblinding step of the signature must be performed somewhere after issuance, but before the creation of the vote to be cast. This poses a new challenge, which in the end could turn out to be way more complex than the changes required for the original proposals. To even make this user-friendly, using the existing IRMA ecosystem, is very likely impossible.

We, therefore, do not consider it recommendable to implement this solution, but merely as a means for experimentation during development. Moreover, when implementing a proof of concept, it would perhaps be better to just allow for a violation of secrecy in the issuance of voting numbers under the assumption that that process can later be performed blindly.

4.4 Overview of the chosen voting scheme

In the previous, we have considered several ways that can be used to acquire relevant attributes in an IRMA 'voting pass' required to realize a system that

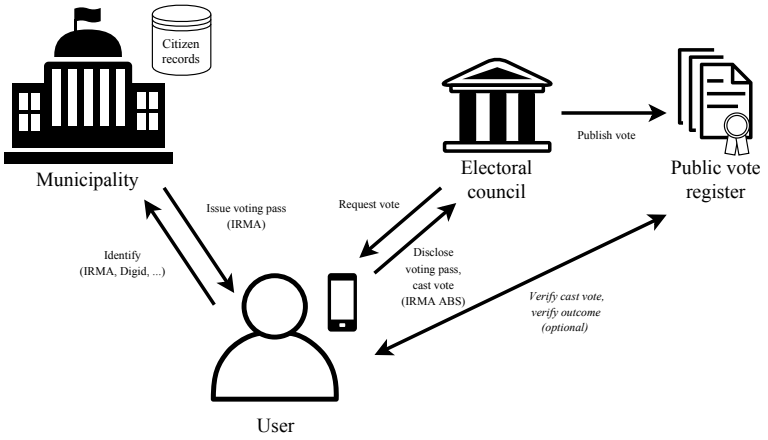


Figure 4.4: A simplified overview of all parties communicating in the voting scheme.

satisfies the requirements. Next, we will consider how those steps can be combined in a voting scheme to be used in elections. For the issuance of the voting number, we will choose the proposal of subsection 4.3.4, since it is simpler for the user.

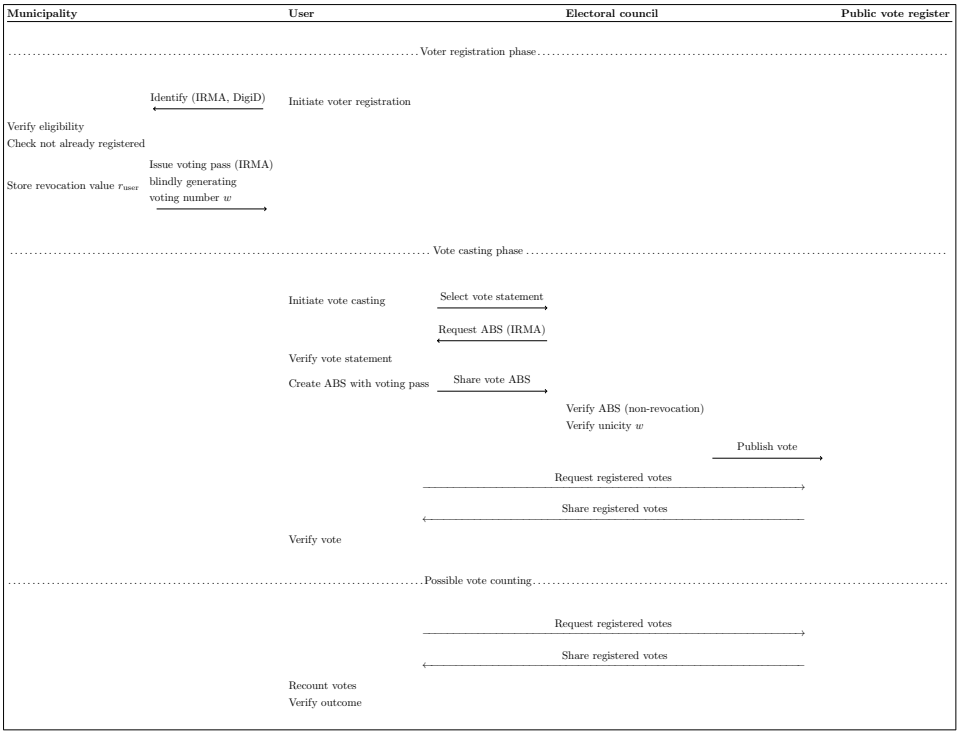
A schematic overview of the scheme is presented in Figure 4.4 and Scheme 4.4.

4.4.1 Prerequisites

The scheme we propose requires citizens that want to participate in the digital election to have installed and set up the IRMA app on their phone. Moreover, we assume two important organizations in our scheme: the municipality and the electoral council.

Electoral council The electoral council shall be the organization responsible for the organization of the election. They provide the infrastructure that citizens can use to vote, and they will report on the election outcome. Everything they do will be completely transparent and verifiable. Hence we do not need to consider them as a semi-trusted party: even a malicious electoral council cannot directly influence the outcome of the election, assuming votes are published in a write-only register and verified by all users. Their role is similar to the electoral council for offline elections: they guarantee availability and accessibility of the election and facilitate verifiability.

More specifically, the electoral council must provide a publicly available register with votes cast. This register must be write-only, so published votes cannot be removed anymore. There are multiple ways to realize such a system, varying from very basic solutions to advanced blockchains. We will not cover the ways to create such a register in this research.



Scheme 4.4: Simplified overview of the voting scheme, based on blind voting number generation as proposed in subsection 4.3.4.

.

Municipality The municipality has a rather specific role in the election: they decide on eligibility. Traditionally, the municipality is responsible for keeping citizens records and will also do so in our elections (even though these days, the national government plays a much larger role in this). The municipality, therefore, is issuer of attributes. Apart from them correctly handing out, re-issuing, and revoking eligibility credentials, based on citizen records, we do not require to trust them.

We can allow the municipality to issue different attributes describing eligibility. For example, some can be based on the regular citizen records the municipality might also issue for other purposes. For our election, however, we require at least one credential that is issued election-specific and includes the voting number for a specific election. Only eligible people should receive this, so we will call it the eligibility attribute or voting pass. At least this credential must be revocable within IRMA (if we want to enable users to re-request a voting pass).

Note that in practice, especially for the OpenStad project described in section 1.2, it is very likely that the electoral council will be closely related to the municipality. This is no problem for our scheme since we do not need to trust either one of them: all properties remain when the municipality and the electoral council would (secretly) cooperate. Nonetheless, as we will see in chapter 5, is it important that in the practical implementation of the protocol, their roles are not merged into a single application that both issues and verifies in a single session, because of timing-based attacks.

4.4.2 Voter registration

To register as a voter, users will interact with the municipality to disclose their identity (either with DigiD or accepted identifying IRMA attributes). The municipality decides on the eligibility of the user and issues an election-specific eligibility attribute, or voting pass. This happens in such a way that a random and unique voting number is included in the credential (described in subsection 4.3.4). The municipality records the IRMA revocation value of that citizen, memorizing that that citizen already has a voting pass issued.

Re-issuing voting passes In case a user might re-request an eligibility attribute during the voter registration phase, the municipality will again verify eligibility and see that that person has already received voting credentials. The municipality should notify the user about this. When the user confirms to proceed, the municipality will first revoke the credential using the recorded revocation value, according to the IRMA protocols. This way, the user cannot use that one anymore to vote. Next, the municipality can proceed in issuing a new voting pass.

Revoking eligibility Between voter registration and election day, it might be possible that eligibility of a user expires because of external reasons (such as death). In such cases, the municipality must directly revoke the eligibility attributes. In theory, this can also be done after the registration phase has ended, but revocation will not have any influence on votes already cast.

To guarantee unicity, we consider two entirely separate phases in our election: voter registration and vote casting. This means that first, all voters should register themselves before election day[8], and only then all registered voters can vote. It may not be possible to register as a voter after the first vote has been cast. Otherwise, re-issued voting passes could violate unicity. We will, however, discuss this requirement further in section 5.2.

4.4.3 Vote casting

In the vote casting phase, registered users will interact with the electoral council. The electoral council will have the infrastructure available to allow users to create and publish an attribute-based vote. This will presumably be in the form of some web page (but other ways could also be used). Here, the voter will choose the option to vote for.

On selecting the option, the electoral council will request an IRMA attribute-based signature for the vote statement chosen, disclosing the eligibility attribute and voting number acquired earlier[9].

The electoral council will verify the vote: all signatures should be correct, the disclosed voting number may not be used previously, and the attributes may not be revoked (they will verify the proof for this, which is delivered by the user). If all is correct, the electoral council will publish the vote in the public register.

Optionally, the user can now verify whether this has indeed been done correctly (and organizational policies must be in place to solve this in case the

[8]The election does not necessarily need to take place on one single day, but can be any time interval.

[9]In practice, we could do two separate things that are equivalent. We could require disclosure of attributes *describing* eligibility, such as Over 18 and Nationality or Hometown, and additionally the Voting number in our vote ABS, that is blindly bound to the same credential. However, since the voting number is also issued by the municipality, and is election-specific, just the voting number itself would suffice as well. It would namely not make sense for the municipality to (blindly) sign voting numbers that belong to ineligible people after all. Only if we would like to organize elections where eligibility is based on attributes from multiple issuers (maybe, we want to have an election in which people that are employed at a company are eligible as well, for which we want to use an attribute issued by that company), we would really require disclosure of multiple attributes in the vote. Also, disclosure of more attributes in the vote ABS could be considered more transparent. The exact choices to make here, depend on the exact implementation of the issuance process, which we will not discuss here.

vote was recorded incorrectly). Because the register is write-only, this check has to be done only once.

4.4.4 Counting votes

After the election, the electoral council will count the votes in the public register and publish the results. Any interested party can verify the results by just recounting all votes and verifying all signatures (that safeguard the integrity of the votes); all steps the electoral council should perform to verify the legitimacy of a vote can be performed offline and transparently. Moreover,

- Secrecy is still intact: no party but the user ever learned the voting number belonging to the user (assuming that users do not publish or disclose their own voting number).
- Eligibility is intact: The municipality issues the required attributes only to eligible users after they disclosed their identity by some trusted means. No ineligible user can retrieve the correct attributes without the municipality's cooperation, because attributes are unforgeable.
- Unicity is intact:
  - When a user would try to vote twice with the same voting number, this can be detected, since the voting number is disclosed in each vote. The electoral council can check whether the voting number is already in the public register and not accept the vote if the voting number is already used (or probably better, not count the previous vote). This can be done offline, and since the register is public, anyone can verify this.
  - When a user would try to vote a second time by somehow presenting a new voting number, the credential issued by the municipality becomes invalid, since it is bound to the presented voting number, and the user will not be able to create a valid ABS for it.
  - When the user would request a new voting number and a new eligibility attribute at the municipality, the municipality will revoke the old credential. With a fully separate voter registration phases, people cannot vote twice.

Naturally, organizational policies must be in place that prescribe how to act in case any party accuses another of any form of fraud, to solve situations in which either the electoral council is not correctly maintaining the public register, the municipality is not correctly issuing eligibility attributes, or a user suspects identity theft.

Chapter 5

Limitations

In the previous chapter, we proposed a voting scheme that can be used with the IRMA framework to realize digital elections. As stated earlier, the scheme is designed to guarantee integrity, unicity, and eligibility while maintaining secrecy and being verifiable, transparent, and accessible. The extent to which all these properties hold, however, depends on the exact context in which the scheme is applied.

In this chapter, we (informally) evaluate under which circumstances those properties are maintained. We identify problems with receipt-freeness and (un)coercibility of the scheme and give suggestions on how this can be improved. Apart from that, we discuss the requirement of entirely separate voting phases and how we can make that requirement less restrictive. We give several extensions or adaptations to the original scheme, that might be recommendable, depending on the exact implementation and context. Finally, we discuss required assumptions and considerations to make when applying the scheme in practical elections and about online remote elections in general.

5.1 Problems with (un)coercibility

In chapter 3, we have discussed the principle of secrecy in elections. The property is two-sided: not only may the electoral council, or any other party (collaborating or not), never be able to learn a citizen's vote, also the user itself may not ever be able to prove what they voted, because this could make them coercible in any form and allow them to possibly sell their vote. In section 3.2.6, we discussed two aspects of this: (practical) (un)coercibility of the scheme in general, with a specific focus on more fundamental receipt-freeness.

The proposed scheme from chapter 4 does satisfy the property that nobody

can find out what a user voted, but it does not disable the user from disclosing this themselves. Because the voting number of the user is just contained in their credential, the user can easily disclose the number multiple times, hence proving what they voted, or even just use the logs of their phone to show the signatures. This makes the scheme vulnerable to coercion.

As also discussed in chapter 3, the problem of (un)coercibility is multidimensional and has a very practical nature as well. For remote election systems, it is custom to just 'give up' on the property. This is because we will never have any guarantees on the physical person that is casting the vote and under what circumstances he is doing that. However, there are some good practices to implement, that make practical small-scale coercion harder.

5.1.1 Changing or retracting your vote

A way to practically solve the problem for a large part could be to allow people to change their vote and possibly even retract their vote at all. This way, if a user would by any means be coerced to vote a specific way, the (coerced) vote could still be retracted later. Naturally, this should only be possible during the election itself.

Since we require the votes to be published in a write-only register (we will discuss more details in section 5.3), we cannot allow the electoral council to just let users delete their votes from the public register.

Instead, we could allow a user to create a vote-retraction ABS using their same voting number. This can just be added to the public register as well, *additional* to the already cast vote. When counting, we should take this into account. Also, when users might try to vote a second time, we could decide that while counting the votes, only the last vote is counted and the others from the same voting number are ignored.

A problem with uncoercibility might still be that this would not go unnoticed since the public register remains available to anyone. By just searching for the voting number, we can easily see whether a vote was retracted or changed. If we consider this a problem, we might still want to be able to violate the write-only constraint. Moreover, as we will also discuss later, we must prevent timing attacks and maybe do not even want to allow the public register to be available during the election (but only after the election). The choices to make here depend mainly on the technology used for the implementation of this public register.

For practical and subtle coercion from private context, the concept of retraction of votes might be a solution (and perhaps the best we can get), but it does not solve the problem fundamentally. Therefore, we should make the

scheme receipt-free as well. In the proposed scheme, this is not straightforward, however. A proposal to do this is presented in section 6.1.

5.2 Requirement of fully separate voting phases

In chapter 4, we discussed the need for fully separate voting phases: first, all users should register to vote, before any vote can be cast. This requirement is one that has large consequences for the accessibility of the election. It requires users to think twice about voting. Though these problems can be solved partially by, for example, sending notifications to registered users reminding them to collect their voting credentials and to cast their vote, we will discuss the implications of not having a fully separate registration phase.

The specific reason for this limitation is that we want to disable users from voting multiple times by first registering, voting, registering again and voting again (which is possible because issuer-revocation has no effect after a vote is cast and it is impossible to verify whether a user has voted without their collaboration).

We will present subtle changes that allow for a subtly less restrictive separation of voting phases.

5.2.1 Election-unspecific voting numbers

One of the reasons we require a voting phase at all is that we need election-specific voting numbers. If we let go of this requirement, we can organize the elections subtly differently. Instead of generating a voting number per election and including it in the municipality's issued credential, we could just generate a single voting number per issuance and use that in multiple elections. This way, users do not need to register for the election, but just need to acquire the municipality-issued credential containing attributes for eligibility every once in a while (as they should also do for other services that use IRMA). This eliminates an election-specific registration phase at all, which is a big win on accessibility.

This approach would make votes linkable to each other, but this does not necessarily have to violate anonymity directly.

To correctly implement this, we must, however, require the municipality to not (re-)issue any credentials on election day. This is because otherwise, people could just vote and acquire a new voting number to vote again. Additionally, the municipality may (still) only issue the credentials to one single IRMA app of a user at a time (so the old one must be revoked as soon

as a new one is issued)[1]. These are new restrictions that have a negative impact on the accessibility of the scheme.

The question is whether these additional requirements and implications outweigh the decreased requirement of election-specific registration from a usability standpoint.

5.2.2 Last-minute retrieval of voting number

A better solution might be just to allow users to register as a voter, right before they want to vote, and not bother about a fully separate registration phase at all. In fact, we can just allow this to happen during election day, as long as the municipality will only issue attributes to *unregistered* voters[2] on election day. Whenever an already registered voter would like to get a voting pass re-issued while the first vote has already been cast, we should not allow that because they might already have voted. This check can be implemented rather easily because the municipality does have this knowledge already.

Sacrificing revocation Users that have not yet registered could just register as a voter even if the election has started, because we know for sure they cannot have voted already. For them, we only introduce the restriction that they are not able to re-request a voting pass. Since they register on election day itself, the problem of people losing their phone is not that much of a problem, so we also do not really need revocation in that case either.

Timing attacks

We have seen we can just allow last-minute voter registration. This, however, does create a problem with the secrecy of the election. Since the user will in practice probably vote directly after registration, the timing of both actions would make them linkable with very high probability. We should, therefore, add some required delay between both steps (or use some other technology to do this).

The exact way to realize this will largely depend on the size (number of voters) of the election. In an election of 1.000 votes, we should probably require more delay than in an election of 100.000 votes. We could consider this the responsibility of the voter: the longer they are willing to wait, the better their anonymity. Moreover, we can still maintain the early voter registration phase before election day, in which citizens can register themselves as a voter and even re-request a new voting pass (with issuer-revocation), as long as the first vote has not yet been cast. However, it is a matter of

[1] In more general sense, for specific attributes that are highly identifying and sensitive to identity fraud, this could anyhow be an interesting policy to implement!

[2] Citizens that have not yet requested voting credentials.

(political) debate how much we can indeed give users this freedom. We will see this very same problem later (in section 5.5) as well when discussing more general problems of digital elections.

5.3 Publishing votes

Closely related to the problem described before is the problem of the timing of votes being published in the public register. In case we can observe a user to 'go vote' (without being able to monitor the content of the session, only when they do it)[3], based on the timing of the update in the public register, we could reveal the content of the vote.

To disable this, we could require the electoral council to only make the register publicly available after the election is finished[4]. In some cases, we might anyhow require this, because, under specific election laws, it might be forbidden to publish intermediate standings during the election at all, because they might influence the outcome.

Just trusting the electoral council to publish the votes after the election is over, could be a solution here. However, we then also trust the electoral council to really do this. Since it is unlikely that users, after the election has ended, will actively check whether their vote is indeed included in the register, this could be considered a problem. The municipality could now potentially drop some votes.

Luckily, also technological measures might exist here, that would allow the electoral council to, for example, publish a hash of each cast vote first and only publish the corresponding original vote later. This way, users could still verify whether their vote has correctly been published, while the content is not yet public.

The exact implementation is mostly dependent on the choices made in the implementation of the write-only public register of votes (as also discussed in subsection 5.1.1), which we will not discuss.

[3]The extent of how far we can observe this depends on the exact implementation of the system and is a fundamental problem to online remote elections, as we will discuss in section 5.5.

[4]Note that still, the timestamp in each generated signature could potentially leak information, so we must require users to wait a specific amount of time, depending on the size of the election, during the casting of their vote before generating the ABS, to make sure this timestamp does not leak too much information but could refer to a sufficiently large number of potential users.

5.4 Correctness limited to application level

We have shown the proposed voting scheme from chapter 4 to satisfy most key properties of an election: integrity, verifiability, secrecy, unicity, and eligibility. An essential remark to make here, however, is that those properties are only realized on the application level of IRMA and the election.

Network-layer identifiers

As often mentioned (for example, in [4, 41]), all properties that IRMA offers and that we rely on, assume that for example, the network layer the application runs on does not reveal additional information. In practice, however, this is an unfair assumption to make. IP addresses are considered highly identifying and fully destroy secrecy of our election when applied in the real world.

A proposal to solve this is to have the IRMA sessions be performed over the Tor network [4]. This way, we do not learn the IP address of a user.

However, this is insufficient. Additionally, the regular network traffic between a user and, for example, the (web) server they contact in order to initiate the IRMA session to cast a vote, must run over the Tor network as well to really keep the vote anonymous. This cannot be achieved by merely changing the IRMA app, but it requires users to actively use the Tor network themselves, instead of their regular browser. Enforcing this would break all accessibility of the system and is practically impossible. To solve this, we could require specific election software, as done in Estonia (see subsection 3.1.1). However, this makes the elections again less accessible.

Secure devices

In addition to network identifiers that can violate secrecy of the system, we must also assume devices of users to be fully secure and unmonitored. Any infected device of the user or intrusion in the user's network might violate the integrity of the vote since the vote could have been altered before it is received by the electoral council. Alternatively, specific software might be monitoring all network traffic or user interaction with the IRMA app, violating secrecy. The scheme we present is correct under the assumption that all actions are performed by a legitimate app controlled by a legitimate user, but in practice, this does not need to be the case. Again we touch upon the more fundamental problems of online remote elections.

5.5 Fundamental problems of remote voting

We have seen various problems with the presented scheme, that are fundamental to online remote elections. People become practically coercible to some extent and we can never rely on users to use fully secure devices and software. Also, by using software, the elections inherently become less transparent to many regular citizens. These are problems we just cannot merely solve in our scheme.

Even if the full technological stack has been fully made to comply with all principles we require, we can never achieve full certainty that the person casting the vote is indeed doing this in all liberty on their own initiative. During remote elections, we just cannot know for sure that the 'entity' casting the vote is indeed controlled by the natural person associated. During offline elections, we do have that certainty because people need to physically identify themselves, using their passport or ID card, at the polling station. Here we can indeed verify (to some extent) that they are not voting under duress.

For some part, we could think that it is the user's own responsibility: users could just use the Tor network themselves if they want better privacy anyway, and if users vote from an insecure device, they should just accept that these are risks associated with them using that device. And for practical forms of coercion, we could think that just offering a way to retract or change a vote should be a sufficient countermeasure. There are sufficient technological measures that solve the problems either partially or fully. It is, however, a matter of political debate to what degree we can indeed give users this own responsibility. We must ask ourselves how far we should only enable security or also enforce it.

These are numerous problems fundamental to having online remote elections. Coercion or intervention of third parties can hardly be detected. Also, the availability of systems could be at risk (think of (D)DoS attacks), which is something we have not discussed at all. In offline elections, we do not have these problems.

That is why for elections with large societal influence and hence are more interesting to attack, one can certainly argue we never should want to have remote online elections at all. For small-scale elections with a rather insignificant impact, however, the ease that comes with digital voting can be a big benefit. Where to place the exact border is a topic for political debate. However, we think the IRMA ecosystem is one of the best attempts to limit any problems.

Chapter 6

Future work

In the previous chapters, we have proposed a voting scheme with several adaptations and considerations to realize digital elections using the IRMA framework. However, we have also seen the proposed scheme to have certain problems that require further research. Additionally, the proposed scheme could also (partially) form a basis for wider application in other contexts than elections. We will discuss the problematic areas that our scheme touches upon that require further research, as well as interesting other applications that can benefit from the proposed scheme.

6.1 Voting number as multi-party computation

As seen in the section 5.1, the scheme we proposed is fundamentally vulnerable to coercion: people can reveal what they voted. This is mainly because users possess their own voting number. The only way to fundamentally counter this problem would be by disallowing the user to know their own voting number at all. Or similarly, at least the 'original' voting number may not be published in the vote. Only then, users will not be able to disclose that number to reveal their identity and vote. There are possibilities to achieve this that require further research.

To achieve this, the voting number could become a multi-party computation, where the final number to be included in the vote ABS only comes to existence in collaboration with some central semi-trusted organization. This organization could again be the electoral council. In the following, we will use the term *voting number* for the original number that is blindly issued to citizens, and the term *final voting number* as the number derived from the original voting number and included in the vote ABS.

This idea is somewhat similar to what already happens in IRMA using a keyshare server for private keys of users (see section 2.3.2). For elections,

however, such server should, among others have the additional requirement of only collaborating one single time to disclose the final voting number, namely when casting a vote (creating the vote ABS)[1].

While voting, we could require the electoral council to include some random value, unknown to the user, in the original voting number to create the final voting number that is included in the vote ABS and published. If users would now reveal their original voting number and identity, they still cannot prove their vote, since the electoral council must collaborate to retrieve the final voting number.

This would require a change in the scheme for IRMA disclosure and ABS generation. Moreover, the implementation must make sure that unicity remains in place in a verifiable manner (so the randomization step may not destroy trustworthiness of the signed credentials by the municipality or create the problem of colliding voting numbers) and the voter may not in any new way be able to reveal that they are the author of the vote. Basically, we must find a way to create partially blind attribute-based signatures. The exact way to make the required changes to the schemes, set up the required infrastructure, and investigate what exact properties should be maintained in this randomization step, is a topic for further research.

Maintaining verifiability

A problem with this extension is that the scheme might become less verifiable. Because users do not know their final voting number, they cannot natively later verify whether their vote is indeed included in the register.

This can, however, be solved relatively easily. Assuming the electoral council would participate in randomizing the (final) voting number, during the vote casting session, the electoral council should in some way, share with the user some identifier of the vote (either a new identifier unknown to the user or the final voting number itself). We will call this a receipt of the vote. This way, the user can later verify the vote.

This receipt must, however, be forgeable: it may not contain any form of digital signature by the electoral council that could prove it linkable to the user. Any user must be able to forge a receipt and use it to show to others they voted some way, even if they did not. Only the user may know which receipt to trust. Only then, the identifier becomes only convincing to the user and not to any other third-party observer (like in section 2.2), hence disabling the user from proving what they voted. The receipt-freeness problem from section 3.2.6 and section 5.1 is then solved by enabling the creation of fake receipts.

[1] Or multiple times, if we allow people to change or retract their vote, as proposed in subsection 5.1.1.

Trusting the electoral council

With this proposed extension, we can introduce receipt-freeness into our scheme, making it less vulnerable to large scale coercion. However, it also requires us to trade some verifiability: it would require us to trust the electoral council to some extent.

Depending on the implementation of the randomization step discussed previously, we must require the electoral council not to maintain records of which final voting number belongs to which original voting number. Otherwise, in collaboration with the electoral council, receipt-freeness can still be violated.

Moreover, in this extended scheme, the electoral council could decide to forge the election outcome. They could do this by registering the same vote of two users only once and share the same identifier to both users. This would go unnoticed: both users would confirm that 'their' vote was published correctly. Nevertheless, there might be cryptographic ways to improve this further, as long as we do not use the voting number (since that would again violate receipt-freeness). This is an interesting topic for further research.

Note that for all statements and suggestions, we still only create receipt-freeness on the application level (as discussed in section 5.4).

6.2 Removing network identifiers and preventing timing attacks

In chapter 5, we have seen that secrecy of online elections in practice very much collapses by so-called network identifiers (most notably, IP address). This data can de-anonymize any vote. There exist well-known techniques to improve this, for example, the usage of the Tor network. Running IRMA over the Tor network would remove the possibility of identifying a user. Research is required to realize this practically. However, it is also required to anonymize the user before initiation of the IRMA sessions, as discussed in section 5.4.

Additionally, we have seen that timing attacks might violate privacy linking a voting number or the vote itself to a user based on time stamps. This is, for example, the case when we allow for last-minute voter registration as proposed in subsection 5.2.2 or on timing of publishing votes as seen in section 5.3. Since in practice, users will most likely not think about this themselves, the most obvious way to solve this is to enforce all steps to be performed separately. However, this makes the election highly inaccessible. Introduction of a semi-trusted party in the network that delays data traffic in order to disable such attacks could potentially solve these kinds of attacks.

Such things are not straightforward, however.

Concludingly, further research is required to make sure the software our scheme builds upon, does not violate the secrecy of votes.

6.3 Unicity using accumulators

A not extensively discussed downside of the system we propose is that the work the verifier needs to perform to verify unicity of each vote is at least logarithmic in the size of cast votes. This is because, for every vote, we need to verify whether not already a vote with that voting number has been cast. Even though this is quite efficient, it is not constant-time as with most other actions in the scheme. Since the tests can essentially also be performed offline after the votes have been cast, this is not a big problem necessarily, but it might certainly be a problem to consider.

This problem is something we have seen in the revocation of attributes as well (section 2.4). Natively, we would require the verifier to verify whether credentials were not on a revocation blacklist. However, the usage of accumulators made this process more efficient, where we put the burden of proving (non-)membership of the black- or whitelist on the side of the user.

A similar approach could be taken to guarantee unicity. Instead of having the electoral council search through the public register to check whether someone voted already, we could instead require users to deliver a zero-knowledge proof of unicity of their vote.

When constructing an accumulated value of all voting numbers in recorded votes, we could let users prove non-membership of this set during signature generation. This would increase the performance of the scheme heavily.

Additionally, since this proof of non-membership is in zero-knowledge, it might also allow for simpler and less restrictive voter registration phases (because we could now let users proof they have not voted yet). It also makes the scheme more difficult, however, since we will have to broadcast updates very often.

Note that to keep the election verifiable, we still require the vote attribute-based signatures containing the voting numbers to be published, so this application would only be required to improve the speed in which verifiers can online check unicity (and not to improve storage complexity of the public register, as was an argument for revocation).

6.4 Blindly issued credentials towards domain-specific identifiers

When we look at the presented system from a broader perspective, we see that it creates the opportunity to create identifiers unique to a person, but without any party knowing who is behind that identifier. Also outside the election context, this can be a very interesting feature.

As an example, when issuing event tickets, we might require this system. Here we can give out tickets (and also re-issue them) with the guarantee that each ticket can be used only once while maintaining the anonymity of the user when presenting the ticket.

Other examples could be medical research. Here it is custom to introduce pseudonyms to each patient participating in the research. For privacy reasons, however, during the processing of the data of the research, these pseudonyms may not be linkable to the patient. The scheme we present delivers precisely that.

For many systems, we might require to have anonymous identities, but we need the guarantee that people cannot have multiple identities. With further research, our presented scheme could be used to realize such domain-specific identifiers.

6.5 Application in non-remote elections

As seen, many vulnerabilities of our proposed system are general to online remote elections, which might be a reason to prefer offline paper elections after all. The benefit of electronic elections, however, is not only that they are more accessible (people can vote from home), but also that counting can be done more efficiently and (assuming integrity of digital votes) without (human) errors.

As seen, one of the benefits of our scheme is that by using attribute-based signatures, the outcome of the election is publicly verifiable. Therefore, it might be interesting to research whether this scheme could be applied in a more hybrid form of traditional elections and online remote elections, where we would still require people to physically visit a polling station (to prevent coercion while voting and have stronger, more reliable authentication while registering), but still cast their vote digitally using their IRMA app. This system could be an alternative to electronic voting machines used in the past. We should, however, not forget that many of the problems of electronic voting in general are not solved (trusting the software, network-layer identifiers). Still, it might be an interesting new alternative that is interesting for further research.

Chapter 7

Conclusions

In our research, we have considered to what extent it is possible to implement digital elections using the IRMA ecosystem, which is based on the Idemix attribute-based credential system. We have seen that there is a societal interest in applications like this, such as the Amsterdam OpenStad project, which desires small scale elections and is planning on using IRMA as a municipality-wide digital identity framework. We have seen that the IRMA framework features several properties that are desirable in election systems, such as unlinkability. The usage of attribute-based signatures, moreover, allows for verifiable recording of votes. However, we have also seen that there are shortcomings, such as the lack of one-show credentials that are featured in Idemix. This makes usage of IRMA in digital elections not straightforward.

To realize elections, we have proposed a voting scheme that delivers the final required feature of unicity while maintaining secrecy, which we have seen to require blindly issued attributes. For this, we proposed the usage of a voting number attribute that is included in an attribute of eligibility issued by the municipality and included in a vote attribute-based signature. We have presented two ways we can make the required changes to IRMA issuance protocols, either based on blind double signing of attributes or blind generation of the voting number during the issuance. We also presented a detour that makes use of regular blind signatures in attribute payload, which could be used for experimentation purposes while not relying on the proposed changes to IRMA.

We have seen the scheme allows for a transparent and verifiable election outcome by using attribute-based signatures for casting a vote.

The proposed scheme does rely on an entirely separate voter registration phase, but we discussed various ways that make this requirement less re-

strictive, hence further increasing the accessibility of the election, while not violating the secrecy. Additionally, we have proposed ways in which the scheme can be extended to allow for change or retraction of cast votes, preventing some forms of coercion. The implementation of this largely depends on the choices made in the implementation of a public register of votes, which is an important topic not covered in this research.

By this, the proposed scheme suffices to almost all requirements of election systems prescribed by law or practically desirable. Nonetheless, we have seen that the scheme does have shortcomings when it comes to receipt-freeness, enabling certain forms of coercion. We have, however, discussed a possibly promising yet complex extension of the scheme that turns a voting number into a multi-party computation, which remains a topic for further research.

Although the proposed scheme with its extensions can be used to realize proper digital elections, we have also seen that the properties only hold on the application level of the system and rely on practically unrealistic assumptions on the underlying layers. Similar to this, but more universally, we have seen the existence of fundamental problems regarding online remote elections in general. There are adequate measures users can take to mitigate the threats, but it remains questionable whether it is desirable to give users that responsibility themselves.

Finally, we have seen several smaller recommendations that should be taken into account during the development of an online election system, among which the preservation of a paper alternative or the use of open-source software with sufficient understandable documentation to keep the system transparent.

In conclusion, we have shown that IRMA is a promising framework to realize digital elections. The proposed scheme is a promising approach to build a proof of concept of such elections. However, in order to meet all requirements, further research is required, and for practical application, there remain several hurdles to take.

For large-scale elections that have larger societal influence and hence, are more vulnerable for attacks, it remains very much questionable whether digital elections are desirable after all. For small scale elections, however, we consider the risks related to online elections not to outweigh the benefits increased accessibility and possibly decreased costs. In this context, we consider having made a good first step into the development of a verifiable and *privacy by design* voting scheme based on the IRMA framework, in which we have delivered recommendations on how to create a proof of concept and have adequately identified problems that require attention and further research.

Bibliography

[1] C. Aarts. *Opkomst bij verkiezingen: Onderzoeksrapportage in opdracht van het Ministerie van Binnenlandse Zaken en Koninkrijksrelaties, Directie Constitutionele Zaken en Wetgeving. (Bijlage bij: Tweede Kamer, 1998-1999, 26200, vii, nr.61).* Ministerie van Binnenlandse Zaken/Justitie, 1999.

[2] Adviescommissie inrichting verkiezingsproces. *Stemmen met vertrouwen.* Ministerie van Binnenlandse Zaken en Koninkrijksrelaties, 2007.

[3] G. Alpár. Cryptography fact sheet about idemix's basic proof techniques. Privacy by Design Foundation, `https://privacybydesign.foundation/pdf/Idemix_overview.pdf`, 2014.

[4] G. Alpár, F. van den Broek, and B. Hampiholi. IRMA: practical , decentralized and privacy-friendly identity management using smartphones. In *10th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs 2017)*, 2017.

[5] V. Augoye and A. Tomlinson. Analysis of electronic voting schemes in the real world. In *UK Academy for Information Systems Conference Proceedings 2018*, 2018.

[6] F. Baldimtsi, J. Camenisch, M. Dubovitskaya, A. Lysyanskaya, L. Reyzin, K. Samelin, and S. Yakoubov. Accumulators with applications to anonymity-preserving revocation. In *2017 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 301–315, 2017.

[7] J. Benaloh and M. de Mare. One-way accumulators: A decentralized alternative to digital signatures. In T. Helleseth, editor, *Advances in Cryptology — EUROCRYPT '93*, pages 274–285. Springer Berlin Heidelberg, 1994.

[8] J. Benaloh and D. Tuinstra. Receipt-free secret-ballot elections (extended abstract). In *Proceedings of the Twenty-Sixth Annual ACM*

Symposium on Theory of Computing, page 544–553. Association for Computing Machinery, 1994.

[9] J. Camenisch, S. Krenn, A. Lehmann, G. L. Mikkelsen, G. Neven, and M. Ø. Pedersen. Formal treatment of privacy-enhancing credential systems. In O. Dunkelman and L. Keliher, editors, *Selected Areas in Cryptography – SAC 2015*, pages 3–24. Springer International Publishing, 2016.

[10] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In B. Pfitzmann, editor, *Advances in Cryptology — EUROCRYPT 2001*, pages 93–118. Springer Berlin Heidelberg, 2001.

[11] J. Camenisch and A. Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In M. Yung, editor, *Advances in Cryptology — CRYPTO 2002*, pages 61–76. Springer Berlin Heidelberg, 2002.

[12] J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In S. Cimato, G. Persiano, and C. Galdi, editors, *Security in Communication Networks*, pages 268–289. Springer Berlin Heidelberg, 2003.

[13] J. Camenisch and E. van Herreweghen. Design and implementation of the idemix anonymous credential system. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, page 21–30. Association for Computing Machinery, 2002.

[14] R. D. Carmichael. *The Theory of Numbers*. The Scientific Press, 1914.

[15] D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. L. Rivest, and A. T. Sherman, editors, *Advances in Cryptology*, pages 199–203. Springer US, 1983.

[16] D. Chaum. Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM*, pages 1030–1044, 1985.

[17] D. Chaum and J.-H. Evertse. A secure and privacy-protecting protocol for transmitting personal information between organizations. In A. M. Odlyzko, editor, *Advances in Cryptology — CRYPTO' 86*, pages 118–167. Springer Berlin Heidelberg, 1987.

[18] L. Chen. Access with pseudonyms. In E. Dawson and J. Golić, editors, *Cryptography: Policy and Algorithms*, pages 232–243. Springer Berlin Heidelberg, 1996.

[19] I. B. Damgård. Payment systems and credential mechanisms with provable security against abuse by individuals. In S. Goldwasser, editor, *Advances in Cryptology — CRYPTO' 88*, pages 328–335. Springer New York, 1990.

[20] D. Derler, C. Hanser, and D. Slamanig. Revisiting cryptographic accumulators, additional properties and relations to other primitives. In K. Nyberg, editor, *Topics in Cryptology — CT-RSA 2015*, pages 127–144. Springer International Publishing, 2015.

[21] J. Dreier, P. Lafourcade, and Y. Lakhnech. A formal taxonomy of privacy in voting protocols. In *2012 IEEE International Conference on Communications (ICC)*, pages 6710–6715, 2012.

[22] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A. M. Odlyzko, editor, *Advances in Cryptology — CRYPTO' 86*, pages 186–194. Springer Berlin Heidelberg, 1987.

[23] E. Ghosh, O. Ohrimenko, D. Papadopoulos, R. Tamassia, and N. Triandopoulos. Zero-knowledge accumulators and set algebra. In J. H. Cheon and T. Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016*, pages 67–100. Springer Berlin Heidelberg, 2016.

[24] S. Goldwasser and Y. T. Kalai. On the (in)security of the fiat-shamir paradigm. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 102–113. IEEE Computer Society, 2003.

[25] D. A. Gritzalis. Principles and requirements for a secure e-voting system. *Computers & Security*, pages 539–556, 2002.

[26] B. Hampiholi, G. Alpár, F. van den Broek, and B. Jacobs. Towards practical attribute-based signatures. In *Proceedings of the 5th International Conference on Security, Privacy, and Applied Cryptography Engineering - Volume 9354*, page 310–328. Springer-Verlag, 2015.

[27] S. Heiberg and J. Willemson. Verifiable internet voting in estonia. In *2014 6th International Conference on Electronic Voting: Verifying the Vote (EVOTE)*, pages 1–8. Institute of Electrical and Electronics Engineers Inc., 2014.

[28] A. W. Heringa and P. Kiiver. *Constitutions compared: an introduction to comparative constitutional law.* Intersentia, 2012.

[29] E. Hubbers, B. Jacobs, B. Schoenmakers, H. van Tilborg, and B. Weger, de. *Description and analysis of the RIES Internet voting system.* EIPSI Eindhoven Institute for the Protection of Systems and Information, 2008.

[30] B. Jacobs and W. Pieters. Electronic voting in the netherlands: From early adoption to early abolishment. In A. Aldini, G. Barthe, and R. Gorrieri, editors, *Foundations of Security Analysis and Design V: FOSAD 2007/2008/2009 Tutorial Lectures*, pages 121–144. Springer Berlin Heidelberg, 2009.

[31] R. Krimmer, M. Volkamer, and D. Duenas-Cid. E-voting – an overview of the development in the past 15 years and current discussions. In *Electronic Voting*, page 1–13. Springer International Publishing, 2019.

[32] K. Krips and J. Willemson. On practical aspects of coercion-resistant remote voting systems. In R. Krimmer, M. Volkamer, V. Cortier, B. Beckert, R. Küsters, U. Serdült, and D. Duenas-Cid, editors, *Electronic Voting*, pages 216–232. Springer International Publishing, 2019.

[33] J. Lapon, M. Kohlweiss, B. de Decker, and V. Naessens. Analysis of revocation strategies for anonymous idemix credentials. In B. De Decker, J. Lapon, V. Naessens, and A. Uhl, editors, *Communications and Multimedia Security*, pages 3–17. Springer Berlin Heidelberg, 2011.

[34] J. Li, N. Li, and R. Xue. Universal accumulators with efficient nonmembership proofs. In J. Katz and M. Yung, editors, *Applied Cryptography and Network Security*, pages 253–269. Springer Berlin Heidelberg, 2007.

[35] W. Lueks, G. Alpár, J. H. Hoepman, and P. Vullers. Fast revocation of attribute-based credentials for both users and verifiers. *Computers & Security*, pages 308–323, 2017.

[36] A. Lysyanskaya, R. L. Rivest, A. Sahai, and S. Wolf. Pseudonym systems. In *Selected Areas in Cryptography*, pages 184–199. Springer Berlin Heidelberg, 2000.

[37] L. Mitrou, D. A. Gritzalis, and S. Katsikas. Revisiting legal and regulatory requirements for secure e-voting. In M. A. Ghonaimy, M. T. El-Hadidi, and H. K. Aslan, editors, *Security in the Information Society: Visions and Perspectives*, pages 469–480. Springer US, 2002.

[38] C. D. Mote. Report of the national workshop on internet voting: Issues and research agenda. In *Proceedings of the 2000 Annual National Conference on Digital Government Research*, page 1–59. Digital Government Society of North America, 2001.

[39] R. Ostrovsky. Foundations of cryptography. *Lecture Notes. Los Angeles, CA, USA: UCLA*, 2010.

[40] D. Pointcheval and J. Stern. Security proofs for signature schemes. In U. Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, pages 387–398. Springer Berlin Heidelberg, 1996.

[41] Privacy By Design Foundation. IRMA technical documentation. `https://irma.app/docs/`.

[42] J.-J. Quisquater, M. Quisquater, M. Quisquater, M. Quisquater, L. Guillou, M. A. Guillou, G. Guillou, A. Guillou, G. Guillou, and S. Guillou. How to explain zero-knowledge protocols to your children. In G. Brassard, editor, *Advances in Cryptology — CRYPTO' 89 Proceedings*, pages 628–631. Springer Berlin Heidelberg, 1990.

[43] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, page 120–126, 1978.

[44] A. Sabouri, J. Camenisch, and K. Rannenberg. Attribute-based credentials for trust (ABC4Trust). In *Attribute-based Credentials for Trust: Identity in the Information Society*, pages 218–219. Springer International Publishing, 2012.

[45] Security Team Computer Science Dept. IBM Research – Zurich. Specification of the identity mixer cryptographic library version 2.3.0∗. Technical report, IBM Research – Zurich, 2010.

[46] M. Solvak and K. Vassil. *E-voting in Estonia: Technological Diffusion and Other Developments Over Ten Years (2005-2015)*. University of Tartu, 2016.

[47] P. Vullers and G. Alpár. Efficient selective disclosure on smart cards using idemix. In S. Fischer-Hübner, E. de Leeuw, and C. Mitchell, editors, *Policies and Research in Identity Management*, pages 53–67. Springer Berlin Heidelberg, 2013.

[48] K.-H. Wang, S. K. Mondal, K. Chan, and X. Xie. A review of contemporary e-voting: Requirements, technology, systems and usability. *Data Science and Pattern Recognition*, pages 31–47, 2017.

[49] J. Willemson. Bits or paper: Which should get to carry your vote? *Journal of Information Security and Applications*, pages 124–131, 2018.