Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Electronic Election System for Academic Senate

Bachelor thesis

2018
Adam Štefunko

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Electronic Election System for Academic Senate

Bachelor thesis

Study programme:	Computer Science
Study field:	2508 Computer Science
Department:	Department of Computer Science
Supervisor:	RNDr. Jaroslav Janáček, PhD.

Bratislava, 2018
Adam Štefunko

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta:
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský

Názov:

Cieľ:

Literatúra:

Kľúčové slová:

Vedúci:
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: doc. RNDr. Daniel Olejár, PhD.

Dátum zadania:

Dátum schválenia:

doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

....
študent

....
vedúci práce

Acknowledgement:

Abstract

Abstract in the English language .

Keywords:

Abstrakt

Slovenský abstrakt v rozsahu 100-500 slov, jeden odstavec. Abstrakt stručne sumarizuje výsledky práce. Mal by byť pochopiteľný pre bežného informatika. Nemal by teda využívať skratky, termíny alebo označenie zavedené v práci, okrem tých, ktoré sú všeobecne známe.

Kľúčové slová: jedno, druhé, tretie (prípadne štvrté, piate)

Contents

List of Figures

Introduction

This is the introduction to our bachelor thesis, introducing its important aspects.

Chapter 1

Description of the Electronic Election System

In this chapter, we talk about some possible definitions of the electronic election system, we propose a definition with which we want to work and we present some requirements on such a system. We also discuss two main approaches to the system in terms of technology used and describe our choice of the system we use for our solution.

1.1 Definition of the Electronic Election System

To have the system correctly and precisely designed, it is necessary to have it clearly defined. At first, we need to say something about the electronic election or voting itself. We have borrowed descriptions from two large online encyclopaedias. We think they are amongst the first sources to which a person being interested in this particular topic is introduced. According to Wikipedia, electronic voting "refers to voting using electronic means to either aid or take care of the chores of casting and counting votes” [22]. In comparison, Encyclopaedia Britannica offers this description of electronic voting, "a form of computer-mediated voting, in which voters make their selections with the aid of a computer” [2]. It can be observed that the two descriptions are vague in terms of roles, or how the actual voting and counting is processed. They only say that computers are used to manipulate with the votes and that voters are enabled to vote using computers. For the use of our thesis we define a computer system which performs manipulation with the votes and which is responsible for the electronic election. This is the definition we propose:

Definition 1: *Electronic Election System* is a computer system which

i. authenticates the voter,

ii. enables the voter to cast a vote,

iii. securely transfers and stores the vote,

iv. calculates votes.

This definition simply enumerates all the basic roles of our electronic election system. We want the election process to be as automated as possible; therefore, the definition is directly derived from the tasks performed during regular election for our academic senate. The system authenticates a voter, so it checks whether the voter is authorised to vote; it enables every authorised voter to cast their vote; it transmits and stores every vote in a way that no vote is lost or changed; and after the voting period has ended, it calculates all the votes and outputs the result.

All the authorities and voters involved in electronic elections need to follow a particular electronic voting scheme (or protocol). This scheme prescribes procedures which should be proceeded during the voting process and which should describe how the electronic election system should perform its roles. Such scheme usually consists of three stages [23]:

1. **Initialisation**, during which the elections are announced, questions are being made, and all the private and public keys are being generated.

2. **Voting**, during which voters are casting their votes: ballots are being created and then sent.

3. **Counting**, during which ballots are being opened and counted, and the final results of the elections are being published.

Several existing voting schemes are discussed in chapter 2 and our proposed voting scheme is described in chapter 3.

1.2 Requirements on an Electronic Election System

Every electronic election system must satisfy several requirements to make sure that all its tasks have been performed unmistakably and that the result is demonstrably correct. In this section, we present several requirements which we find essential for the purposes of the elections to the academic senate. We also compare them to the requirements presented in bibliography sources on this topic.

P. P. Bungale and S. Sridhar from Johns Hopkins University, Baltimore have named several *Requirements for an Electronic Voting System* [3]. They have divided them into two categories: "Functional Requirements" and "Security Requirements". We have chosen several of them which we find most important for our electronic election system, and we have added a few that are not included in Bungale's and Sridhar's original paper, yet we find them very important. We have partially followed their division. However, besides adapting the terminology we have added an extra category for the sake of better distinction. Thus, we divided them in these three categories: *usability*, *security*, *accuracy.*

Usability requirements are *easy-to-use user interface*, *mobility*, *cost-effectiveness* and *confirmation.*

- **Easy-to-use user interface.** The system should have a user interface which voter can use easily with (almost) no instructions provided. On top of that, voting options should be displayed in a way that no candidate is disadvantaged.
- **Mobility.** The voters should not be restricted to a certain place where they can vote. In terms of electronic voting, the voter should not be limited to a certain type of technology used for voting. In spite of Bungale's and Sridhar's opinion, who call voting via the Internet "*infeasible* both for security issues as well as social science issues" [3], we advocate Internet voting because we think that in certain conditions, it is preferable and can meet our requirements (particularly this one).
- **Cost-effectiveness.** The technology used for electronic voting should not be expensive and hard to implement, yet it must provide adequate functionality and security, so it can be effectively used as an electronic election system.
- **Confirmation.** Each voter should have the chance to confirm that their vote corresponds their own decision and have the chance to modify their vote before committing it. Voters also should have the chance to verify whether their vote was correctly transferred and stored.

Security requirements are *secrecy*, *anonymity*, *reliability* and *incoercibility.*

- **Secrecy.** There must be no chance to determine how a voter voted.
- **Anonymity.** No vote must be associated with a voter's identity.
- **Reliability.** The system must be robust enough, so that no votes are lost or illegally changed in any case, and it must be ensured there is no malicious code or bugs. Also, the system should be simple enough because such system offers fewer possibilities for the attackers and, thus, is less vulnerable.
- **Incoercibility.** There must be no way in which voter can prove how they have voted. This prevents from anyone else's impact on voter's choice and it also prevents from vote-selling.

Accuracy requirements are *authorisation*, *uniqueness and limitation*, *persistence* and *correct computation.*

- **Authorisation.** It must be secured that only authorised voters can cast their votes.
- **Uniqueness and limitation.** Each voter can participate in an election by no more than one vote and no more candidates than allowed can be chosen.
- **Persistence.** It must be guaranteed that votes remain intact after they have been committed and sent.
- **Correct computation.** The votes must be correctly computed according to the published rules of the election.

For comparison, Rjašková [23] has provided a set of seven requirements, which we introduce to the reader: *eligibility*, *privacy*, *individual verifiability*, *universal verifiability*, *fairness*, *robustness*, *receipt-freeness*, *incoercibility.* We strongly believe that almost each our requirement finds its counterpart in one or more Rjašková's requirements, and vice versa. For instance, counterpart to confirmation is individual verifiability and counterpart to reliability is robustness. Receipt-freeness means that there is no way to prove how a voter voted [5]. This can be substituted for incoercibility and vice versa.

1.3 Types of Electronic Election System

In this section, we present two possible types of electronic election system, which are the most common, and of which we choose one for our purposes. Then, we compare those two types regarding our requirements which we presented in section 1.2. We also present their advantages and their drawbacks. Finally, we give reasons for our choice of used type of electronic election system.

Regarding the technology used during the voting phase (as described in 1.1), there are two main types of electronic election system: *e-voting* and *i-voting* [2].

1.3.1 E-voting

This type uses special-purpose machines designed directly for the purposes of the elections. These machines either directly record the ballots or they optically scan traditional paper ballots, which are then stored in their internal memory.

Reliability relies on testing the machines during the initialisation phase and on confidence that the same software is running during the whole election process. However, thanks to relatively smaller number of individual machines used during the election and thanks to higher possible responsibility from the authority, these machines can be checked for malicious software and controlled without much difficulty. Also, network communication between a device of this type and any other device is generally disabled. Hence, it is possible to implement this type of electronic election system in a way that it does not break security requirement in such a manner which is fatal to the election process.

However, regarding functional requirements, we find this type of system awkward. Firstly, special-purpose hardware is built to hold the tasks during the election process and these machines are placed in special-purpose polling stations where is a commission to control voting process. Voters have to come to the polling station where they can cast their ballot. Depending on the software in these machines, it is possible that the voter has to learn how to use the machine before they cast their vote.

1.3.2 I-voting

The second type, on the other hand, uses Internet to hold the communication between the authorities and the voter that enables the voter to be authorised and to cast their vote.

One of the biggest security issues with this type of the system is the relatively large amount of independent devices, owned mainly by the voters themselves, which can run a malicious software which might not, depending on the security support of the individual device, be spotted. Rubin notes how many violent acts may potential attackers perform: "view every aspect of the voting procedure, intercept any action performed by the legitimate user with the potential of modifying it without the user's knowledge, and further install any other program of the attacker's desire—even those written by the attacker—on the voter's machine" [17]. The second issue is the Internet communication itself. Thus, cryptographic protocols, some of which are to be described particularly in chapters 2 and 3 and which are largely discussed in [23], are used to prevent Internet communication from vulnerability during the election process and to assure to a great extent that security and accuracy requirements are not to be broken.

Despite all the security issues, mobility and cost-effectiveness are two important advantages of this type of electronic voting system. A voter needs nothing, but their computer or mobile device with either a special-purpose application or Internet browser. Moreover, it is relatively cheap to develop a special software that runs on a user's device. There is left to consider these two approaches to the software solution—whether to use Internet browser or a special-purpose application—because we find the first option as preferable from the point of view of mobility, whilst the second is, in our opinion, preferable when considering security. We should still bear in mind that a lot of important details lie on the way how the system is developed, which is to be discussed more deeply in chapters 3 and 4.

1.3.3 Comparison

All these facts and expectations described above have been considered for the purposes of our choice. We state that not all requirements we present in section 1.2 and which are tight to the two types described in this section are equally important for the purposes of the system. Also, some requirements can be equally satisfied using either type. Mobility and cost-effectiveness have perhaps been the most crucial during our decision process because our system is to be made for no more than a thousand of voters involved in one election. Albeit not the most secure, our system should provide reasonable security using accessible solutions. Some other security and functional requirements, such as anonymity, reliability and persistence, are essential, but these can be adequately achieved using either type of the system and appropriate protocols.

E-voting systems are ideal for elections involving a large number of voters, e.g. parliamentary elections, which is not our case. Thus, we decided to use i-voting because it meets our functional requirements perfectly and it also provides adequate security thanks to cryptography. Possible security issues can be treated quite quickly without any difficulty also thanks to the relatively small amount of individuals involved in the elections.

Chapter 2

Summary of Existing Solutions

In this chapter, we introduce and describe some techniques used to design electronic voting schemes as well as some existing solutions to the electronic elections.

2.1 Electronic Voting Scheme Techniques

Electronic voting systems use several cryptographic primitives to assure integrity, authenticity and correct transfer of votes as well as to preserve voter's anonymity. There are at least three specific techniques used to design an electronic voting scheme. These are *blind signatures*, *anonymous channels* and *homomorphic encryption* [11]. They are used besides symmetric and asymmetric cryptographies, certificates, digital signatures, etc. Here, we describe the ideas on which they are based and some of their properties. We also briefly describe *mixing networks*, *onion routing* and *David Chaum's untraceable electronic cash protocol.*

2.1.1 Blind Signatures

In many voting systems, blind signatures are used to detach the vote from the voter's identity. The basic principle of such schemes is that blinded empty ballot is sent to the authority alongside voter's personal information. After verifying voter's identity and correctness of the ballot, it is blind signed by the authority and sent back to the voter. Then, voter unblinds the ballot, fills it and sends to votes collector. Simple example of such a scheme is described in [1].

As written in [11], all protocol requirements, except receipt-freeness, are accomplished in schemes using blind signatures when some other cryptographic primitives, such as encryption, are used together with blind signatures. But, these schemes have restricted usability due to the fact that in order to ensure fairness and verifiability, voters need to be active in at least two phases. However, there are some protocols that overcome this functional drawback.

2.1.2 Verifiable Anonymous Channels

We use anonymous channels to send anonymous messages. This means that there is no way to trace the received message back to the sender. Anonymous channels can be built using *mix networks* or *onion routing.*

- **Mix Network** uses a chain of proxy servers that mix the multiple messages and send them in a random order to the next node, which might be either another mixing server or a different type of device.

- **Onion Routing** encapsulates a message in new layers of encryption. Analogous to this are layers of onion, hence the name. The message is, then, transmitted through a series of subsequent routers, each decrypting the actual top layer of encryption and uncovering the message's next destination. In every moment of the message transfer, only precedent and following location is known to each *onion router*, thus making the sender's identity untraceable [10].

It is necessary to ensure that no messages are dropped or substituted during the transfer. Therefore, *proofs of correct computation* need to be provided. Channels that are able to do so are called *verifiable anonymous channels.* Their main drawbacks are computational complexity and inefficiency [11].

2.1.3 Homomorphic Encryption

Use of homomorphic encryption is based on the desire to decrypt the sum of votes without being able to decrypt the individual ballots. For that reason, voters can openly authenticate to the server.

However, such schemes have several serious drawbacks. Not only do they require computationally intensive zero-knowledge proofs, but they also do not allow voter to choose multiple options.

2.1.4 Untraceable Electronic Cash Protocol

In [4], David Chaum introduces a protocol which can be used to send electronic cash while preserving sender's anonymity and avoid double-spending at the same moment. Moreover, if a sender tries to send several copies of an electronic coin, their identity can be revealed. The protocol also provides a method that can be used to reveal double-spender's identity. With a few modifications, this protocol is useful for electronic voting to prevent double-voting.

The Chaum's protocol can be described in the following way. It can be noticed that we present only the most important steps to the reader.

In this example, the symbol $\cdot$ denotes concatenation. Let n be an RSA modulus and let f and g be two-argument collision-free functions. Let u be sender's account number and v be a counter associated with this number and kept in the bank.

1. **Formation**

   (a) The sender chooses $k \in \mathbb{N}$ values a_i, c_i, d_i and r_i, where $1 \leq i \leq k$. All a_i, c_i, d_i and r_i are modulo n.

   (b) The sender sends to the bank k *blinded candidates* $B_i = r_i^3 f(x_i, y_i) \pmod{n}$, where $x_i = g(a_i, c_i)$ and $y_i = g(a_i \oplus (u \cdot (v + i)), d_i)$.

   (c) The bank randomly picks $k/2$ indexes from $\{i_1, ..., i_k\}$. The sender shows their a_i, c_i, d_i and r_i to the bank for each picked i. From these, the electronic coin C is formed following steps described in [4].

2. **Payment**

   (a) In order to pay, the sender sends their coin C to the shopkeeper.

   (b) The shopkeeper randomly chooses $k/2$ binary values $x_1, ..., x_{k/2}$. For each $x_i \in \{x_1, ..., x_{k/2}\}$:

      - If $x_i = 0$, the sender shows the shopkeeper a_i, c_i and y_i.
      - If $x_i = 1$, the sender shows the shopkeeper $a_i \oplus (u \cdot (v + i)), d_i$ and x_i.

   (c) The shopkeeper sends obtained values to the bank in order to verify the coin C.

When the sender tries to use C twice, it can be observed that with high probability complementary binary values will be sent to the bank for at least one x_i and the sender's identity will be revealed. This is possible because in this situation, all the values i, a_i and $a_i \oplus (u \cdot (v+i))$ are known, from which the sender's identity can be easily extracted.

Although not designed directly for the purposes of electronic election, this protocol can be used, requiring only slight modifications, such as making u a unique personal identification number given to every voter prior to the election. Such a modified scheme can be found in [15].

2.2 Existing Electronic Election Systems

Many governments are interested in replacing traditional election schemes based on paper voting with electronic election systems. At the turn of the millennia, *SERVE* was an ambitious experiment by the United States of America in the field of Internet voting. It was later cancelled, mostly due to large security weaknesses SERVE showed.

Switzerland was amongst the first countries to use electronic elections in some of their cantons. Estonia was the first country to run nationwide electronic parliamentary elections. In this section, we describe pioneering election systems developed for Estonia, Norway and Switzerland.

2.2.1 Estonian Internet Voting

Estonian identification cards (ID cards) have a chip that stores asymmetric cryptography key pair, which their owners can use to digitally sign documents. This has been used to develop an internet voting protocol, which was first used during the election in 2005, while it was still possible to vote traditionally [19].

The protocol used in Estonian internet voting system uses method of *double envelope*. While the outer envelope is used to provide the voter's identity, the inner is used to cryptographically protect the vote. These entities play role in this protocol: *voter*, *voting commission*, *voting application*, *server*, *ballot box*, *counting device* and *certification authority*.

The protocol consists of three stages, which we now describe:

1. **Initialisation.** Election asymmetric encryption key pair is generated. The public key is given to every voter while the private key is divided using a secret-sharing scheme and each member of voting commission is given their piece. The voting application is digitally signed and provided to every voter.

2. **Voting.** Voter, who has validated their ID card, connects to the server and launches the voting application on their device. In order to establish secure connection, the voter needs to provide a PIN code associated with their ID card. The voter, then, chooses their favourite options to form the vote. When the vote is formed, it is encrypted using the election public key and a random value r, and the voter digitally signs the vote using their ID card. The *double-enveloped* vote is, finally, sent to the server.

   The server verifies voter's identity and strips off the upper envelope. It sends the vote in the lower envelope to the ballot box and associates it with a random token t. The voter can check whether their vote was correctly recorded using a verification app on their smartphone. The app reveals r and t, which are used to compare the encrypted vote stored in the ballot box with a simulated vote encrypted using r and corresponding the voter's choices.

3. **Counting.** Valid encrypted votes stored (without the signatures) in the ballot box are burned to a DVD, on which they are transferred to the counting device.

The private key is reconstructed and loaded onto the counting device. It produces the result, which is burned on another DVD and published.

Estonia's Internet Voting Committee claim that, in terms of security and reliability, this voting protocol is equal to traditional elections. However, it seems to be controversial. Tests run by a team from the University of Michigan show that the system can be successfully attacked [19].

2.2.2 Norwegian Internet Voting Protocol

In Norway, trial of Internet elections was first organised in 2011. A protocol that would satisfy the security expectations was defined for the Norwegian elections [6] and a new cryptographic protocol was published two years later [7]. A new instantiation of the cryptosystem underlaying Internet voting in Norway was introduced and is described in [8].

In the original paper, associated with the election in 2011, a simplified voting protocol is described. We believe that this protocol provides enough information about the ideas on which the Norwegian internet voting is based. The players in the protocol are: *voter*, *voter's computer. ballot box*, *receipt generator* and *decryption service.*

The voting scheme is divided into three stages and the following steps are processed:

1. **Key generation.** Asymmetric encryption keys and three secret parameters for the election are generated, first for the decryption service, second for the ballot box and the third for the receipt generator. From these, associated public parameters are computed. Expected return codes for each voter are also generated and sent to the voter.

2. **Voting.** When the voter choses their options, voter's computer sends the encrypted vote to the ballot box. With the cooperation of receipt generator, receipt codes are generated and sent to the voter. Those are checked by the voter, if they correspond with the expected receipt codes.

3. **Counting.** Submitted votes from the ballot box are sent to the decryption device. There, they are decrypted and counted in order to get the final result.

According to the author of the protocol, its security properties are not perfect, yet it is reasonable to be used for an i-voting experiment. In order to assure that the vote remains confidential, the voter's computer should not be corrupted. If the vote is not correctly submitted, although not corrupted before being submitted, the voter can see this through receipt codes and can complain. The author also concludes that "any corrupt infrastructure player may prevent the election from completing." [6].

After the elections in 2011, the protocol used for it was gradually renewed and improved. One of the last introduced improvements on the underlying cryptosystem was that new techniques were used to prove the knowledge of the decryption of the encrypted text. These new techniques have the same effect as former; however, due to their use, the underlying cryptosystem has a better security proof [8].

2.2.3 Swiss Online Voting Protocol

There are multiple public voting events in Switzerland every year. Not only can the citizens choose their representatives in Federal Assembly, but also referenda are organised federally or regionally.

Cantons of Geneva, Zürich and Neuchâtel were the first to introduce electronic voting. In 2013, The Federal Council of Switzerland published a framework which provides security, verifiability and functionality requirements in order to allow all the voters to vote electronically. In a publication by Scytl [21], a protocol ensuring cast-as-intended verification is provided. This protocol has been created on the basis of Norwegian internet voting protocol (described in this thesis in 2.2.2). It is basically the improvement of that protocol "by not needing to rely on the strong assumption that two independent server-side entities do not collude to preserve voter privacy" [21]. The building blocks of the protocol are *ElGamal assymetric encryption scheme*, *bit-length prime numbers* representing voting options, *pseudo-random function family*, *signature scheme* made up of probabilistic algorithms and a *verifiable mix network.*

These are the participants in the Swiss voting scheme:

- **Election Authorities**, responsible for the whole elections;
- **Voters**, who express their opinion by casting a vote;
- **Registration Authorities**, who provide voters with all the information needed;
- **Voting Server**, which receives, proceeds and stores the votes;
- **Voting Device**, responsible for enabling the voter to select their options, forming the vote and correctly send it to the voting server;
- **Code Generator**, which generates return codes from the votes;
- **Auditors**, in charge of verifying the correctness and integrity of election process.

The Swiss voting scheme consists of following processes divided into four phases:

1. **Configuration.** A set of voters' identities ID is defined and published. The protocol uses several key pairs, of which public keys are published. Those keys are *election public key*, *global code generation public key*, *signing public key*. Voting options are also published in this phase. The global code generation private key is given to the code generator and the registration authorities, and the signing private key is given only to the registration authorities.

2. **Voter registration.** In order to register to the election, voter provides their identity $id \in ID$ to a registration authority. Then, voter is given their public and private keys, a set of return codes associated with particular choices and confirmation and finalisation codes. Also, voter's public key is published together with reference values associated with return codes and a validity proof for finalisation code.

3. **Voting.** When the voter is successfully authenticated, voter's *id* and public key are stored in their voting device. The voter votes by choosing from the voting options and entering their private key. From these, the vote is created and send together with voter's *id* to the voting server.

   When the server successfully receives the vote and the voter's identity, return codes are generated and shown to the voter, who, then, is asked to confirm the vote. The voter does so by providing their confirmation code to the voting device. Subsequently, the confirmation message is sent to the server. When the vote is confirmed and finalisation code is sent back to the device and the voter checks whether it matches the finalisation code they have obtained during the registration.

4. **Counting.** The counting algorithm takes the votes stored on the voting server, the election private key and validity proofs for finalisation codes and produces the final result, which is, then, verified.

The described protocol requires the voter to type several private values to the voting device. This means to copy by hand 52 or 410 characters when *Base32* is used [21]. The voter cannot perform such a task. Therefore, the protocol also describes a usability layer, whose role is to reduce the length of values that the voter needs to directly provide for the voting system. Its description can be read in the original paper a we do not include it in this thesis.

Chapter 3

Our Solution

In this chapter, we present our own solution to the electronic election system. We describe players in the final voting scheme we implement, the voting scheme itself and the format in which the vote is stored and read. Then, we bring some description of existing security schemes and services we use in our solution. We also discuss some other proposals of the voting scheme, which we modified or rejected.

3.1 Our Voting Scheme

Our voting scheme is similar to the scheme that has been developed for elections in Estonia. This scheme is described in chapter 2. Both schemes are based on the principle that voter's identity cannot be known to any player besides that particular voter when their vote can be decrypted or read. It means that computer that is responsible for decrypting the votes cannot obtain any information about any voter who casted their vote.

We bear in mind that we require the voting process to run in an Internet browser and that this technology has several limitations. The reader can read more about limitations of the technology used in chapter 4.

3.2 Players in Our Voting Scheme

We shortly describe each player in our voting scheme and their purpose.

- **Voter** V authenticates to S using *authentication authority* T and casts their vote using *voting application* A. They also can cast their vote in paper form in a special-purpose room.

- **Election commission** B are in charge of key generation during the first phase. They hold secret key SK generated for *machine for vote counting* M and share corresponding public key PK. They also securely transfer encrypted data from *server for vote collection* S to M. Data can be, then, decrypted using SK, provided to M by B. In the end, they publish the result of the elections. Last but not least, they are responsible for paper votes casted in a special-purpose room designed for the elections.

- **Database** D stores information about all the candidates and identities of all authorised voters. This information is then used by A and S to display list of possible candidates and authorise *voter* V. This database is physically stored on server S and is accessible only to S.

- **Voting application** A provides user interface for V in which they can cast their vote. It also validates the vote, converts it to a proper format and encrypts it using PK. Then it sends the vote together with the voter's identity.

- **Server for vote collection** S receives encrypted vote and voter's identity and checks whether this identity corresponds to any of those stored in *database* D. If the encrypted vote comes from an authorised voter, then it stores the encrypted vote and the voter's identity in its internal database of votes. This server also provides storage for voting application A and database D.

- **Machine for vote counting** M receives all the votes (without identity of any voter) stored in S's internal database and secret key SK. It, then, decrypts all the received votes using SK, validates them and computes the result of the elections, which is finally published. We point out that this machine is not connected to any network prior or during the elections. This machine is restricted to be in off-line mode until the results of the elections are published and it is proved that the election process has been correctly performed.

- **Authentication authority** T is responsible for authentication of V and provides M with V's identity, which must *1-to-1* correspond to a value in D. For purposes of our voting scheme, we use *Cosign*, which is a service commonly used for authentication of students studying at our faculty. From this point, we mean *Cosign* any time we talk about T. Similarly, we use term *UK login* of voter V instead of *V's identity*. More information about *Cosign* is provided in section 3.5 and chapter 4.

3.3 Phases of Our Voting Scheme

Our voting scheme is designed to have three subsequent phases: *initialisation phase*, *voting phase* and *counting phase*. The basic purpose of these stages is discussed in chapter 1. In this section, we provide a detailed description of what is done during each of these phases.

3.3.1 Initialisation

1. Database D with a table of candidates is created and stored on S. For each candidate, the table contains a unique ID and name of the candidate.

2. A table of authorised voters is created in D. For each voter, this table contains their *UK login* and a boolean variable set to 0. This variable indicates whether the voter has already casted a paper vote.

3. A special *UK login* is created for the election commission B. This login is used when a voter casts their paper vote.

4. Public key PK and secret key SK are generated. They are to be used to encrypt the vote by the voting application A and decrypt it by the machine for vote counting M, respectively.

5. Using *Shamir's Secret-Sharing Scheme*, the key SK is divided into n parts, where n is the number of members of B. The key SK can be reconstructed from any combination of k parts, where k is the smallest number of members of B that have to sign the *Protocol of elections*. The reader can read more about the scheme in section 3.5.

6. Public key PK is shared with all the voters by inserting it into A as a constant value for the whole elections.

3.3.2 Voting

For each voter V:

1. Voter V opens a specific Internet location in their browser and logs in using $Cosign$ in order to authenticate to S.

2. If V is successfully authenticated, voting application is launched in their Internet browser. This application contains a form in which there are three options for each candidate.

3. When V submits the form, it is then validated regarding the rules of the elections. When form valid, vote in the format described in section 3.4 is created and this vote is encrypted using *S/MIME* protocol and PK. Finally, the vote and voter's *UK login* are sent to S via an $HTTPS$ connection.

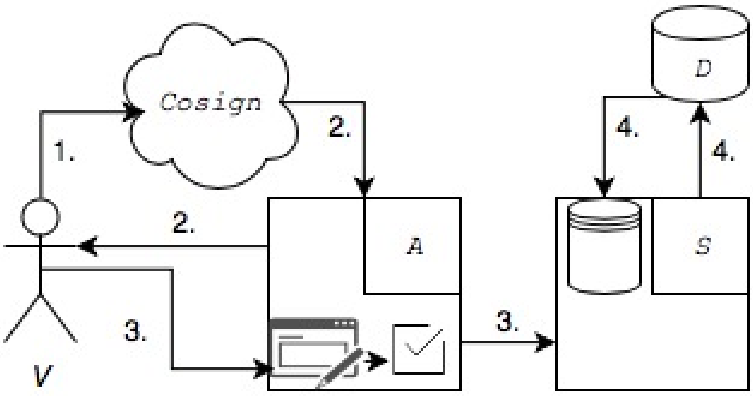

Figure 3.1: Voting phase of our scheme

4. When S receives the encrypted vote, it compares the sender's *UK login* with logins in the table of authorised voters stored in D. If it matches exactly one record, then the login and the encrypted vote are stored in S's internal database of votes. If there already is a record with the same login and vote different form the special `0` character vote in the internal database, the former vote is rewritten with the new one.

5. If V decides to use paper vote and goes to the special-purpose voting room, the election commission B authenticate to S using their special *UK login*. When they give V a ballot paper, they access the table of authorised voters and set the boolean value in the voter's row to 1. This is how they prevent V from casting another electronic vote. When V puts their vote to the ballot box in the room, B sends a special `0` character vote together with V's *UK login*, which rewrites V's electronic vote if they have sent one.

3.3.3 Counting

1. Server S is disconnected from network and off-line mode is enabled.

2. Using a secured USB device, the election commission B transfers encrypted votes from S's internal database of votes to machine for vote counting M. It is crucial that *UK logins* stored in this database be excluded from the transfer.

3. With aid of at least k members of election commission B, secret key SK is reconstructed. Then, it is inserted into M and used to decrypt votes.

4. Decrypted votes are, then, validated by M. All votes that are in the right format and follow the rules of the elections are counted and the final result is published.

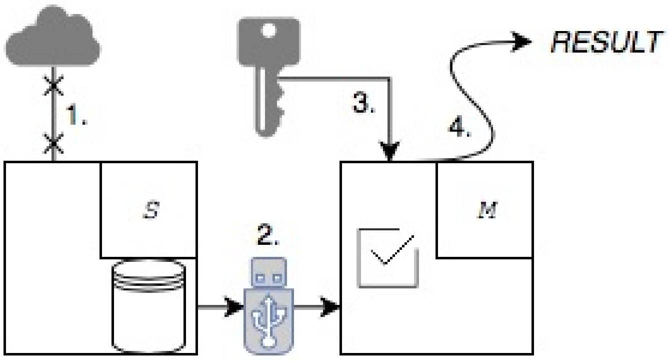

Figure 3.2: Counting phase of our scheme

3.4 Format of the Vote

Definition of format to which each vote is formatted prior to being sent to the server for vote collection S is also part of our voting scheme. We want the vote to be stored in one compact file and the choices to be clearly separated. It is known how many candidates are in the elections. Let M be the number of candidates. Hence, we propose that the vote is of the following format:

```
<candidate_number_1#value_1>...<candidate_number_M#value_M>
```

Here, `M` is the representation of number of candidates M.

This format represents a string of ordered pairs (c_i, v_i), where $c_i \in \{1, ..., M\}$ is `candidate_number_i` and $v_i \in O$ is `value_i`, $i \in 1, ..., M$. Value v_i is numerical representation of one of the voting options and O is the set of all possible numerical representations. For example, 1 represents YES, 0 represents NO and $O = \{0, 1\}$.

Let $(c_1, ..., c_M)$ be an ordered set of all candidate numbers. Then, $(c_1, ..., c_M)$ is a permutation of $\{1, ..., M\}$.

It is obvious that the candidate number 1, for instance, is given YES if and only if the vote contains exactly one ordered pair of value $(1, 1)$.

3.5 Security Tools Used in Our Voting Scheme

In our voting scheme, we use these cryptographic protocols and schemes as well as security services to securely transfer and store the data.

3.5.1 S/MIME

Developed to provide a secure way to send and receive MIME data, S/MIME (Secure/Multipurpose Internet Mail Extensions) consists of several cryptographic services, such as authentication or data encryption. Not only can it be used by traditional email clients but also by message transfer services that use cryptographic solutions to prevent any person from illegal intervention. In order to use any of the services provided by S/MIME, one must obtain a certificate from a certificate authority, which they then must install. S/MIME uses asymmetric cryptography based on *PKCS#7* standard.

Ramsdell and Turner provide a detailed specification of S/MIME Version 3.2 in [16]. More can be also read in [20] and [13].

3.5.2 Shamir's Secret-Sharing Scheme

In [18], Adi Shamir proposes a solution to the problem of dividing data into n pieces in such a way that these data can be reconstructed using k of these n pieces, but attempt to reconstruct the data with fewer than k pieces results in no information about it. Such a scheme is called *(n, k)-threshold scheme.* The scheme he proposed is based on polynomial interpolation.

Let $p(x)$ be a polynomial of degree d. It is obvious that $d + 1$ distinct points $(x_0, p(x_0)), ..., (x_d, p(x_d))$ are sufficient to define $p(x)$ and that $p(x)$ cannot be defined with fewer than $d + 1$ distinct points.

In Shamir's scheme, assume that data D is a number. Then, to divide D into n pieces in such a way that it can be reconstructed by using at least k pieces, we follow these steps:

1. We pick a random $k - 1$ degree polynomial $p(x) = a_0 + a_1 x + ... + a_{k-1} x^{k_1}$, where $a_0 = D$.

2. We, subsequently, evaluate $D_1 = p(1), ..., D_n = p(n)$. Then, each of n distinct pieces of secret consists of an ordered pair $P_i = (i, D_i)$, where $i \in \ 1...n$.

3. In order to reconstruct data, at least k pieces of secret $P_{i_1}, ..., P_{i_k}$ are put together. Polynomial interpolation is then used to find $p(x)$, which is the Lagrange polynomial for the set of used pieces of secret. This makes recovering data D possible because $p(0) = a_0$ and $p(x)$ has been constructed in such a way that $a_0 = D$.

The scheme as described above consists only of the basic idea on which secret sharing is based. In reality, we use modular arithmetic instead of its real counterpart. Let $p > \max(D, n)$ be a prime number. Then, the set of integer coefficients $a_0, ..., a_{k-1}$ is picked randomly from a uniform distribution over integers in $[0, p)$. Values $D_0, ..., D_n$ are also computed modulo p. The set of coefficients forms a field $\mathbb{Z}_p$, which makes polynomial interpolation and data retrieval possible [18].

This scheme has some useful properties. It can be observed that the individual pieces are at most of the same size as the original data. With k fixed, some pieces can be added or deleted without affecting the other, or the pieces can be changes without changing the original data. It is also possible to reflect hierarchy in a particular group, such as election commission, in a way of distributing the pieces. For example, chairperson of the commission is given three pieces, vice-chairperson two and the other members one.

3.5.3 Cosign

Comenius University in Bratislava uses *Cosign*, a secure single sign-on web authentication system originally developed at the University of Michigan [14]. It is based on web cookies.

Authentication data are sent to the central server, which uses an authentication protocol to verify these data. The server, then, sets the user a web cookie, controlled by *Apache* filters during every user's attempt to access addresses secured by the system. During the whole process, *TLS* is used for a secured communication.

In chapter 4 is described how authentication via *Cosign* has been integrated into our system.

3.6 Other Proposed Solutions

In the course of designing the final voting scheme for our election system, we proposed some other solutions. Those were either rejected, or the final scheme was based on them and their modifications. There are two elementary design classes being in consideration during the whole process. Those are based on question whether server gives client any additional information used to authorise the sender of the vote. Regarding this, we consider two meaningful classes, which we call *authorisation by a token* and *authorisation by user information.* In all of the schemes we have looked at, at least three players represented by computers are needed: a *client* C, an *authentication server* A and a *counting machine* B.

3.6.1 Authorisation by a token

In this design, authorisation in provided by a *token.* This token assures B that the vote comes from an authorised voter without associating it with a particular voter. In order to preserve anonymity, token must be generated or handled with in a way that it cannot be associated with any particular voter. In chapter 2, we describe some existing solutions using this design.

Here, we briefly describe voting phase of a scheme that uses server A's signature to prove that the vote comes from an authorised voter.

Let V be voter and $n \in \mathbb{N}$, $n \geq 2$, be constant defined by the scheme. Then, for each voter V:

1. Voter V logs in and voting application is opened on C.
2. Client C sends n encrypted votes and V's identity to A, where 1 of those is the real vote and the other votes are fake.
3. If V is an authorised voter, A signs all n votes and sends them back to C.
4. Client C sends the signed real vote to B.
5. Server B validates the vote and if it recognises the signature with which the vote is signed, then it stores the vote.

The main issue with this scheme is that it may enable the voter to double-vote when their voting application is corrupted. In chapter 2 we introduce the reader to David Chaum's scheme, which was designed to be used with electronic cash to prevent double-spending. When using a modification of this scheme to prevent double-voting, we face a problem. In order to achieve this, additional data about the voter have to be generated and stored on client's side. Of course, these data cannot be stored on voter's personal computer or smartphone, since our scheme is supposed to be mobile and these data have to be accessed from any device of those kinds. This means that a cloud storage in which these data can be stored has to be provided. This storage has to be secured in a way that only voter can freely access the data. It raises new security tasks, which we think the reader can imagine. We do not say that it is impossible to implement such a system, but we think that it is not necessary to implement such a complex system for purposes of our type of elections.

We have also come up with a totally different scheme using mix networks, referred to in chapter 2, to mix the tokens in order to assure anonymity.

3.6.2 Authorisation by user information

Using voting schemes in this class, C assures that the vote comes from an authorised voter by providing a publicly inaccessible piece of information unique for that voter that can be matched with a record in A's memory. In order to retain anonymity, the vote must be in an illegible form during the whole time it can be associated with this piece of information.

Here, we provide a simple scheme that has been modified and partly used in the final voting scheme.

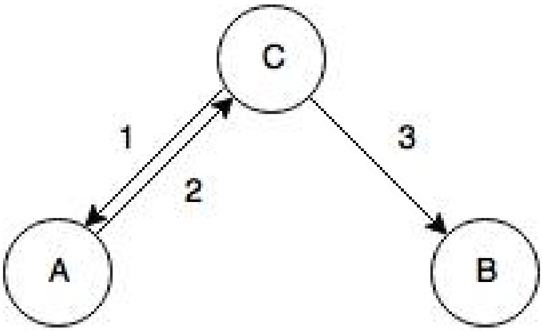

Figure 3.3: Authorisation by a token

Let V be voter. Then, for each voter V:

1. Voter V logs in and the voting application is opened on C.

2. Client C sends encrypted vote and V's identity to A.

3. If V is authorised to vote, then A sends the encrypted vote without V's identity to B.

4. Server B stores the vote.

This scheme has several security issues. Let corrupted A send B encrypted votes and voters' identities. These votes can be, then, decrypted by B, which also knows who casted which vote. Thus, it can be revealed how voters voted.

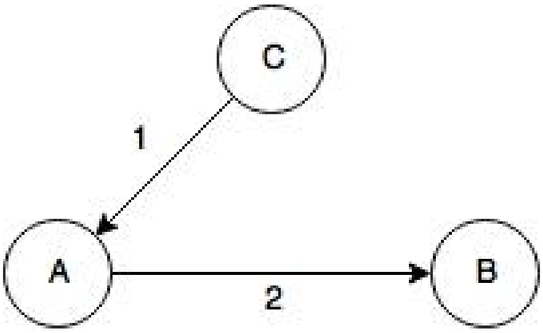

Figure 3.4: Authorisation by user information

Chapter 4

Implementation of Our Solution

In this chapter, we describe some important implementation details and we provide examples of important parts of our code.

4.1 Overview

4.2 Database

4.3 Voting Application

4.4 Server for Vote Collection

4.5 Machine for Vote Counting

4.6 Supplementary Programs

Chapter 5

Analysis of Our Solution

In this chapter, we discuss how our solution meets requirements established in chapter 1.

5.1 Usability

5.2 Security

5.3 Accuracy

Conclusion

This is the conclusion of our bachelor thesis, summarising its important points.

Bibliography

[1] Standford University Applied Gryptography Group. Electronic Voting. Retrieved April 21, 2018, from `https://crypto.stanford.edu/pbc/notes/crypto/voting.html`.

[2] Encyclopaedia Britannica. Electronic voting [online]. Retrieved January 24, 2018, from `https://www.britannica.com/topic/electronic-voting`.

[3] Prashanth P. Bungale and Swaroop Sridhar. Requirements for an Electronic Voting System. Department of Computer Science. Johns Hopkins University, Baltimore. Available at `http://www.cs.jhu.edu/~rubin/courses/sp03/group-reports/group4/group4_requirements.pdf`.

[4] David Chaum, Amos Fiat, and Moni Naor. Untraceable electronic cash. In *Conference on the Theory and Application of Cryptography*, pages 319–327. Springer, 1988.

[5] Stéphanie Delaune, Steve Kremer, and Mark D Ryan. Receipt-freeness: Formal definition and fault attacks. In *Proceedings of the Workshop Frontiers in Electronic Elections (FEE 2005), Milan, Italy*, 2005.

[6] Kristian Gjøsteen. Analysis of an internet voting protocol. *IACR Cryptology ePrint Archive*, 2010:380, 2010.

[7] Kristian Gjøsteen. The norwegian internet voting protocol. *E-Voting and Identity*, pages 1–18, 2012.

[8] Kristian Gjøsteen and Anders Smedstuen Lund. The Norwegian Internet Voting Protocol: A new Instantiation. *IACR Cryptology ePrint Archive*, 2015:503, 2015.

[9] Kristian Gjøsteen and Martin Strand. Fully homomorphic encryption must be fat or ugly? *IACR Cryptology ePrint Archive*, 2016:105, 2016.

[10] David Goldschlag, Michael Reed, and Paul Syverson. Onion routing for anonymous and private internet connections. *Communications of the ACM*, 42(2):39–41, 1999.

[11] Rolf Haenni, Eric Dubuis, and Ulrich Ultes-Nitsche. Research on e-voting technologies. *Bern University of Applied Sciences, Tech. Rep*, 5, 2008.

[12] Epp Maaten. Towards remote e-voting: Estonian case. *Electronic Voting in Europe-Technology, Law, Politics and Society*, 47:83–100, 2004.

[13] Microsoft. Understanding S/MIME [online]. Retrieved April 15, 2018, from `https://technet.microsoft.com/en-us/library/aa995740(v=exchg.65).aspx`.

[14] University of Michigan. cosign: web single sign-on [online]. Retrieved April 16, 2018, from `http://weblogin.org`.

[15] Michael J Radwin and Phil Klein. An untraceable, universally verifiable voting scheme. In *Seminar in Cryptology*, pages 829–834, 1995.

[16] Blake Ramsdell and Sean Turner. Secure/multipurpose internet mail extensions (S/MIME) version 3.2 message specification, 2010. Retrieved April 16, from `https://tools.ietf.org/html/rfc5751`.

[17] Aviel D. Rubin. Security Considerations for Remote Electronic Voting. *Communications of the ACM*, 45(12):39 – 44, December 2002.

[18] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[19] Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine, and J Alex Halderman. Security analysis of the estonian internet voting system. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 703–715. ACM, 2014.

[20] Sean Turner. Secure/multipurpose internet mail extensions. *IEEE Internet Computing*, 14(5):82–86, 2010.

[21] Scytl Secure Electronic Voting. Swiss Online Voting Protocol.

[22] The Free Encyclopedia Wikipedia. Electronic voting [online]. Retrieved January 24, 2018, from `https://en.wikipedia.org/wiki/Electronic_voting`.

[23] Zuzana Rjašková. Electronic Voting Schemes. Master's thesis, Faculty of Mathematics, Physics and Informatics. Comenius University, Bratislava, April 2002. Available at `https://people.ksp.sk/~zuzka/elevote.pdf`.