NTNU

Norwegian University of
Science and Technology

A Survey of Modern Electronic Voting Technologies

Martine Stenbro

Master of Science in Communication Technology
Submission date: June 2010
Supervisor: Danilo Gligoroski, ITEM

Norwegian University of Science and Technology
Department of Telematics

Problem Description

Electronic voting is a term encompassing several different types of voting, embracing both electronic means of casting a vote and electronic means of counting votes. Electronic voting technology can include punch cards, optical scan voting systems and specialized voting kiosks (including self-contained Direct-recording electronic (DRE) voting systems). It can also involve transmission of ballots and votes via telephones, private computer network or the Internet. Electronic voting technology can speed up the counting of ballots and provide improved accessibility for disabled voters. However, electronic voting is could also facilitate electoral fraud.

In this thesis the student will perform a study of modern solutions and technologies for electronic voting.

Assignment given: 21. January 2010
Supervisor: Danilo Gligoroski, ITEM

Abstract

The last decade, electronic voting has evolved from being a mean of counting votes to also offer the possibility of electronically casting votes. From recording votes using punch cards and optical scan systems, electronic voting has evolved into the use of direct-recording-electronic machines. Voting over the Internet has also become a hot research topic, and some implementation and testing have been done. Internet voting systems are significantly more vulnerable to threats from external attackers, than systems to cast ballots in controlled environments. Mechanisms to provide security, accuracy and verification are critical, and issues with coercion and usability also arise.

In the first part of this thesis we give a theoretical study about existing electronic voting techniques, as well as requirements and security issues of modern electronic voting systems. We also give a brief background theory of some cryptographic mechanisms and systems.

Secondly, we present two modern voting solutions in development. We have included security functionalities provided by the system, the cryptographic techniques used and some threats and attacks to the systems. These systems can be exposed to compromised computers, ballot stuffing, and corrupt infrastructure players, but are using cryptographic proofs to ensure accuracy and counter attacks.

In the third part, we create a procedure and perform a usability test on one of these modern voting solutions. Our findings emphasize the fact that there is a tension between verifiable elections and usability. The voters have trust in the privacy and accuracy of such a voting systems if more guidance to utilize the means of verification is included, and a trusted third party verifies the system security. The advantages of electronic voting outweigh the risks.

Internet voting is a term of further discussion and testing, but considering coercion and the insecure aspects of the medium, Internet voting will never be 100% safe. It is a question of trade off between the advantages and threats.

Preface

This master thesis was given as a specialization project in the last semester of the authors studies at The Norwegian University of Science and Technology (NTNU). The assignment was given by Professor Danilo Gligoroski at the Department of Telematics, NTNU.

This project assignment was a great learning experience and gave me an insight into the challenging area of e-voting. The research and study of the field of electronic voting was very interesting, especially how Internet voting is developed as cryptographic verifiable elections. Giving the many different research niches in this field, the research process of extracting information was comprehensive and challenging. It was also informative to get some hands on experience with one of these voting systems, and to create a procedure for evaluating the system, giving some useful feedback.

I would like to thank my professor Danilo Gligoroski for his valuable input and quick response to questions. He supported me with good ideas for the experiment approach and the thesis in general.

I would also like to thank all the test participants for taking part in the test election and giving me some interesting feedback.

Trondheim Juni, 2010

Martine Stenbro

Abbreviations

CAI Common Authentication Infrastructure

DOS Denial-of-service

DDOS Distributed denial-of-service

DRE Direct-recording electronic

E2E End-to-end

GSM Global System for Mobile Communication

HMAC Hash-based Message Authentication Code

HTML HyperText Markup Language

HTTPS Hypertext Transfer Protocol Secure

IACR International Association of Cryptologic Research

MAC Message Authentication Code

OCR Optical Character Recognition

PIN Personal Identification Number

PKI Public Key Infrastructure

SAML Security Assertation Markup Language

SHA Secret Hashing Algorithm

SMS Short Message Service

SSL Secure Sockets Layer

URL Uniform Resource Locator

Definitions

Air-gap To ensure a secure network/computer is completely physically, electrically, and electromagnetically isolated from any other computer/network

Audit-trail A chronological log of audit records

Authentication The provision of assurance of the claimed identity of a person or data

Ballot The means by which the voter can express his choice of voting option

Ballot box Where the votes are stored after cast before counting. The ballot box can be physical or electronic

Coercion Using force to compel a person into doing something or abstain from doing an action, depriving a person of his free will

Confidentiality The property that sensitive information is not disclosed to unauthorized individuals, entities, or processes

Cryptography Study of hiding information, through means of mathematics and computer science

Decision Diffie-Hellman Problem A computational hardness assumption about a certain problem involving discrete logarithms in cyclic groups. It is used as the basis to prove the security of many cryptographic protocols (ElGamal)

Denial-of-service attack Attacks preventing normal users to access a service or computer resource. Preventing any part of an information system from normal functioning

Electoral roll A registry of all eligible voters in a country

Hashing Algorithm An algorithm which takes a variable length data message and creates a fixed size message digest

Integrity Assurance that information has remained unaltered from the point it was produced by the source, during transmission, storage, until receipt by the destination. (That the data is unchanged from creation to reception)

N-version architecture A system architecture where the same inputs are given to N different sub-systems and results are

Secure Sockets Layer A secure handshake protocol for authentication that supports X.509 certificates at the transport layer

Tuple The notion of an ordered list of elements, a set of values

Validation (Verification)Finding or testing the truth of something, checking if it satisfies a certain criterion. The process of establishing a valid proof of something

Zero Knowledge Proof A cryptographic interactive method for one party to prove to another that a statement (or operation outcome) is true, without revealing anything else than the trustworthiness of the statement

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As technology has moved forward in several aspects of our lives, the increase in use of mechanics and electronics has also emerged. The use of mechanics in the area of voting was introduced as early as the 1890s with the invention of the Herman Hollerith punch card machinery for the US census [2], and later developed into electronic voting. Electronic voting, or e-voting, is a term encompassing several different types of voting, embracing both electronic means of casting a vote and electronic means of counting votes.

As the area of voting has evolved from public voting, in the early US, to the use of paper ballots with mark choices, the area of electronic voting has also evolved. Electronic voting has changed from the use of punch cards to the use of optical scan systems and specialized electronic voting kiosks (including direct recording electronic voting systems). Electronic voting can also involve remote transmission of ballots via telephones or private computer networks, and the latest development is voting over the Internet.

With the growing usage of computers there is also an increasing amount of both private and governmental services available via the Internet. The Internet can be used for economic transactions, tax forms, university admission services, etc., and as a consequence a development into being able to provide voting via the Internet is also considered.

Electronic voting technology at polling stations can speed up the counting of ballots and provide improved accessibility for disabled voters. A voting system providing remote voting via the Internet could improve the accessibility and provide an even more convenient voting process. It might also lead to greater voter turnout in elections, as well as phase out existing cumbersome (and insecure) processes of absentee voting, which in US still is done

by mail [3].

The promises of accuracy, security and precision have driven electronic voting systems forward. However the worries of corrupt or manipulative software have held back widespread adoption of several systems. Especially Internet voting schemes are discussed and criticized. There are many threats and issues with attacks and fraudulent behavior, including network attacks compromised computers and corrupt system components, to mention some.

The field of cryptographic technology has also evolved. Remote Internet voting systems is now developing into being categorized as cryptographic voting systems, not only to provide the sufficient secrecy but also to provide possibilities of verification. Josef Stalin once said:"Those who cast the votes decide nothing. Those who count the votes decide everything." The purpose of cryptographically verifiable voting systems is to prevent incorrect recording and tallying, by making the processes verifiable for everyone. As paper-based voting solutions should include the possibility of re-tallying an election when errors arose, the paper audit trail is in cryptographic systems being replaced by electronic receipts and the mean of proofs produced in cryptographic mechanisms.

"Trust me" and "The hardware and software have been thoroughly tested!" are no longer acceptable statements for a voting system, or justifications of an election outcome [4]. There is no longer sufficient to control the equipment and let "trusted" officials operate it, it is necessary to verify each election outcome. From 2005 and up to date a number of researchers has proposed ideas for achieving higher assurance of election integrity, without having to trust hardware, software, election officials, and without violating voter privacy [5] [6], and the field of creating trustworthy elections is in continuous development [4]. With the Helios system, we will describe in chapter 4, being the system *most* researched and implemented [7].

1.1 Motivation

In recent years electronic voting has become a very popular and hot research topic. The electronic mean of counting paper ballots has existed for a while now, but the focus now is more on how to cast electronic ballots. Electronic voting technology can speed up the counting of ballots and provide improved accessibility for disabled voters. Voting via Direct recording machines or the Internet could also decrease the use of paper ballots and the manual work

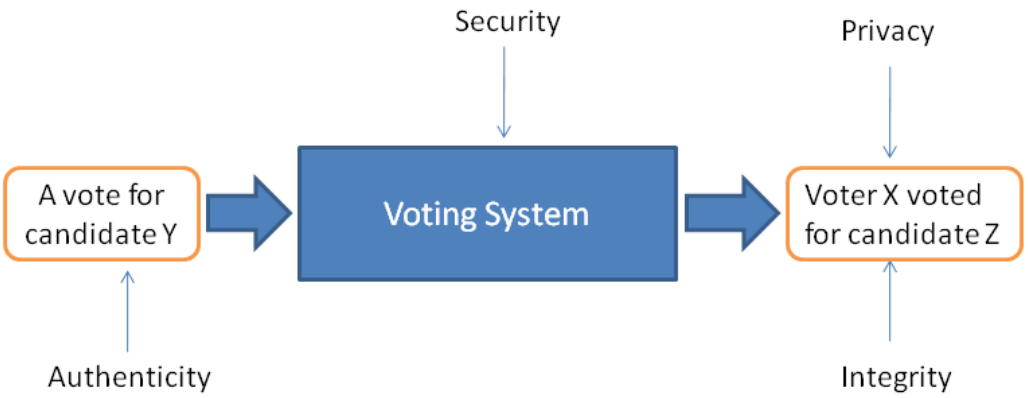

Figure 1.1: Generic set of requirements for a voting system [8]

of preparation (no need of printing ballots in different languages, etc.) The correctness of these systems is also critical, as there is no room for errors in the electronic recording or counting of ballots.

Electronic voting can also facilitate electoral fraud, and especially the concomitant risks of remote voting are significant. The general requirements of a voting system are shown in figure 1.1, but with an Internet voting scheme it gets harder to fulfill these requirements. An Internet system are vulnerable to more and new threats; fraud, compromised computers, attackers compromising system components or other attacks launched over the network. The need for better mechanisms of authentication, anonymity, integrity, confidentiality, secrecy, etc. is crucial. With the development of Internet voting systems, the need for cryptographic means both for providing security against these threats and for verification purposes is increasing. The possibility of verifying several steps of the processes is also an important mean to detect and counter several attacks. The public awareness of such elections called open-audit elections and the verification possibilities it provides should be improved.

This fall The National Institute of Standards and Technology (NIST) had a workshop regarding so-called end-to-end verifiable voting systems, which is an improvement of verification in voting systems. These are voting systems that can provide the voter with verifiability through the entire process of casting votes. These systems are widely discussed, and are a great improvement of electronic voting, but such systems can also lead to drawbacks not only with coercion problems, but also with the usability for the voter that are actually requested to use the system.

The International Association for Cryptologic research (IACR) has the last two years been considering systems they can use for internal elections in the organization and discussed various systems in the field of e-voting [9]. They have considered several systems and are now testing an open-audit system called Helios [10], created by Ben Adida and Harvard University. This system, which will be described later, has also been used for elections at another university in the US, and maybe a similar system could be used for governmental elections in nearby future.

Some countries have developed, and are testing or already deploying remote voting via the Internet in elections. In Estonia a system has been deployed [11], research and discussion is going on in the US, and also in Norway electronic voting is a hot topic these days. The Norwegian Government is working on a project called "E-vote 2011", planning to try out advance voting via the Internet in some selected municipals for the 2011 municipal election [12]. If everything works out according to the plan, they aim to make it possible for everyone in Norway to vote in advance via the Internet for the 2017 general/parliamentary election.

Since electronic voting is an area of growing interest, this thesis will include a study of technologies used for electronic voting. It will include a description of modern solutions and techniques, new voting systems evolving, and also requirements and security issues of these modern systems.

As two similar, but still significant different modern cryptographic voting systems the online system of Helios and the system being developed for E-voting in Norway these days, will be presented. To gain a greater understanding and get some hands on experience with the Helios voting system, the thesis will also include a test election using this system.

There is also discussion going on (for instance in the mentioned NIST workshop) about the combination of auditability and usability in the development of these end-to-end verifiable voting systems. There is a tension between these two properties; since the voter has to go through more tasks to perform verification, it is less efficient and convenient. The voter has to understand the means of auditability and therefore it can not be presented to the voter in cryptographic terms, etc. Therefore we also wanted to perform and include a usability test of the Helios voting system, it being an open-audit voting system.

1.2 Objectives

This thesis consists of three main parts. First, to give an introduction to the term, a theoretical background study about existing electronic voting systems is given; systems and techniques to record or count votes, requirements of a modern electronic voting system, security threats, etc. In this part some theory related to cryptography used in modern voting system solutions is included.

Secondly, we have chosen two modern voting solutions in development to look further into and present; looking at voting solutions, the cryptographic techniques used, and threats and attacks to the systems.

In the third part we created a procedure and performed a usability test on one of these modern voting solutions. We created a fictional test election using the system, recruited 40 test participants to vote and then present an evaluation of the system seen from a potential user. With this test we hoped to retrieve some interesting feedback to the challenges of creating an auditable and usable modern voting system.

1.3 Method

Our research can be divided into three phases:

- Theoretical study of modern solutions and technologies for electronic voting
- Presentation of two modern voting systems; the Helios voting system and the Norwegian "E-Vote 2011" system
- Test election with the Helios voting system

For the two first points of our research, our main source of information included books, papers, standardization documents and accredited web pages. The process was to find good sources of useful information using the library and the Internet, to get required knowledge in the area of electronic voting and cryptographic techniques.

Before the test election was created, a procedure for analyzing the usability of Helios was planned and an online survey was created. The results from the survey was gathered compared with result of a similar test. The test methodology used during the usability test is described in 5.

1.4 Limitations

The area of electronic voting is extensive and complex. This thesis describes several modern technologies for electronic voting, but mostly the field of remote Internet voting is further considered. Only two Internet voting systems and their security solutions are presented. Security or accuracy of other existing techniques or solutions are not further explored.

Regarding the "E-Vote 2011" system in development, the information is based partially on available information and updates from the project's web site and the proposed solution from the chosen contractor. The solution and documentation might change during the development phase approaching the test in 2011. All documentation will be available for analysis before deployment, but now changes are still being made.

1.5 Document Structure

The remainder of the document is structured as follows:

Chapter 2: Background Theory presents existing technologies and solutions for electronic voting systems, some requirements and security issues of such systems, and used cryptographic techniques and schemes.

Chapter 3: "E-Vote 2011" describes the planned Internet voting solution to be implemented for an test election in Norway 2011, including the internet cryptographic protocol and security threats.

Chapter 4: The Helios Voting System contains a presentation of the Helios Open-Audit voting system. The chapter includes security mechanisms and some potential threats to the system.

Chapter 5: Testing Helios includes a description of using the Helios system to create a test election, and presents the procedure and results of the performed usability test.

Chapter 6: Discussion and Conclusion compares the two modern solutions, and discuss issues with such E2E verifiable voting systems. Conclusion is included in this chapter and summarized the research.

Additionally, the following appendices are included:

Appendix A: "E-vote 2011" contains a figure over the entire "E-vote 2011" system.

Chapter 2

Background Theory

In this chapter theory related to electronic voting is presented. Some existing e-voting techniques, with advantages and disadvantages are presented. Requirements of a modern e-voting system and the security issues of these systems are also included, as well as some scenarios were such systems have been used. In the end some cryptographic techniques and protocols used in voting systems are explained.

2.1 Electronic Voting Systems

In this section, theory related to existing voting systems and technologies used are presented. This includes technologies for paper-based voting systems where electronic means are used to count the paper ballots, and technologies for electronically casting votes. Some of the technologies mentioned are older technologies used in less voting systems these days, while some are still in phases of development and testing.

2.1.1 Paper-Based Electronic Voting Systems

Paper-based electronic voting systems are voting systems where the electronic mean of tabulation is used to count paper ballots. The votes are cast, or marked, on paper ballots by hand or using a marking device, and then an electronic tabulation device is used for counting. This can speed up the process of counting, and not only give less manual work but also less error regarded to human failure. To use the electronic mean of counting ballots,

different types of marked paper ballots can be used to cast the vote, depending of what kind of electronic device that is used for tabulation. These systems include punch card voting, and the *marksense* and optical scanning systems.

2.1.1.1 Punch Card Systems

The use of punch cards for data handling is a technology that has been used since the 19th century [13]. The punch card sorting and tabulation equipment was invented by a statistician named Herman Hollerith. It was first built to process the large amount of information from the 1890 US census, but later developed for commercial and scientific purposes, and also for voting.

The machinery of a punch card system consists of several components to carry out the processes. A punching device as showed in figure 2.1 punches the information and data on the cards. When having several decks of punched cards a card collator machine can sort and merge cards from different stacks according to certain factors, for instance adress or county. To sort/tabulate the cards with specific loads a sorting/tabulation machine can be used. An example of a tabulation machine is showed in figure 2.2.

The punch cards are stiff cards or paper strips that are punched according to a certain value or designation. There exist a lot of different types of punch cards, of different sizes, with a different amount of punching locations, for different purposes. Punch cards were used as input for data processing in electromechanical devices like tabulation machines and unit record equipments, which was used before the invention of today's electronic computers. These machines had a function of sensing punched holes with either electrical or optical sensors, and could have high-speed mechanical feeders that made it possible to process up to 2000 cards per minute [13]. To generate a summary (report) of a deck of punch cards, they could be fed into the tabulation machine and selected fields from each card were added to the value of one or several counters in the machine.

Early computers were using punch cards for program entry and storage. When using punched cards, the computer could register the presence of a punched hole with "1" and the absence of a hole with "0" and by this save the information in the binary number system. As late as up to 1970, the punch card was the most popular storage medium used, but is now an almost obsolete technology [13]. Figure 2.3 shows an example of a punch card from a Fortran program. The punch card is no longer used for programming

Figure 2.1: A punching device to punch holes in a punch card [14].

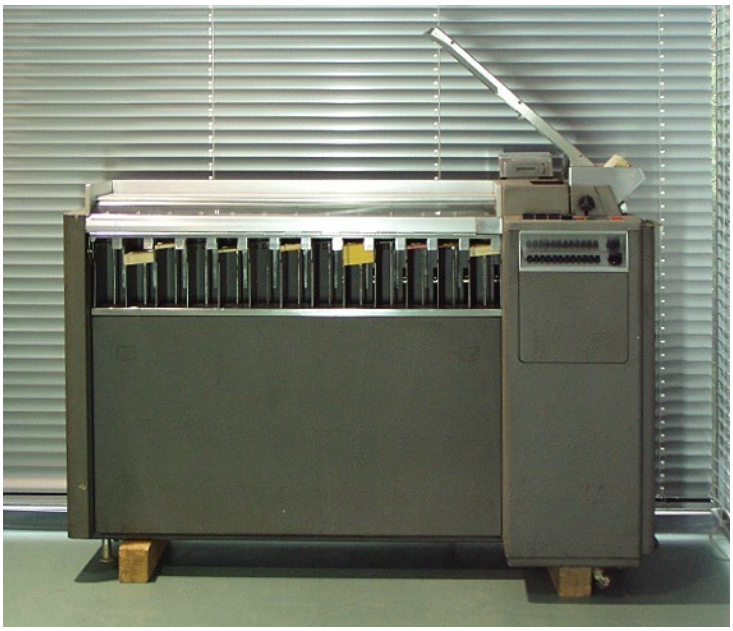

Figure 2.2: A sorting machine for punch cards [14].

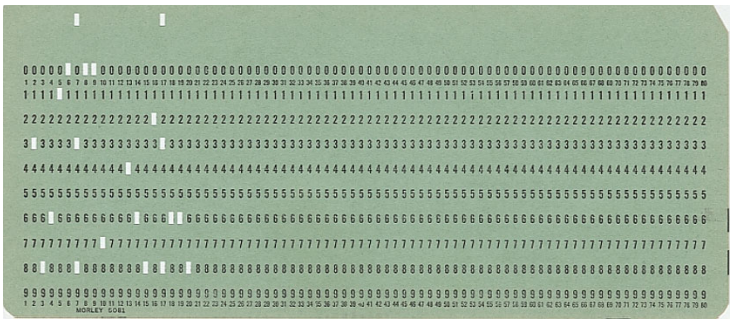

Figure 2.3: A punch card with a Fortran programming statement. [15]

purposes, but has been used in voting systems in some states in the US recently.

An advantage to mention with the punch card system used for the 1890 census, except for the efficiency, was the ability for verification. The card with its printed image was kept by a board and enabled all the cards to be read back for recounting or for verification by others [13]. Verification and paper-audit trail are also issues that arise in the discussion about electronic voting scenarios.

Punch Cards for Voting

Electronic voting systems based on punch cards have been used since 1964 [2]. For voting purposes the punch cards were used to record the vote and then they were run through a tabulation machine to count the votes. When using punch cards as ballots for voting purposes the perforations represents the polling choice made by the voter. The voters punch holes in the cards (with a supplied punch device) opposite their candidate or ballot issue choice and place it in a ballot box. To generate a result, the punch cards can be fed into a tabulation machines at the precinct, or all punch cards can be gathered and transported to a central location for tabulation.

There are two common types of punch cards used for voting purposes; the "votomatic" card and the "datavote" card [2]. The "votomatic" card is a direct descendant of the original punch cards (by Joseph P. Harris [REF]) and only has numbers indicating the holes to be punched. The ballot issue choices or list of candidates are printed on a separate booklet, and you punch the card according to which number that corresponds to your choice of candidate. On the "datavote" card, the names of the candidates are printed on the ballot next to the punch hole. The use of punch cards systems for voting purposes has both advantages and concerns. When using punch cards and a sorting machine there is easy to detect misplaced cards, to prevent errors in the counting process. Another advantage is that there exists an audit trail of punch cards making it possible to perform a recount if necessary. One of the concerns associated associated with the use of punch cards is related to the material of the card. When using punch cards for tabulation, errors related to the holes on the card can occur. When punch cards are used for elections in the US the voters cast their vote by using a pin to punch through the card by hand [2]. This leftover paper that was punched out is referred to as the "chad". If the holes are not aligned or fully punched, called the "hanging chad", the ballot can be counted incorrectly when tabulated.

Many jurisdictions switched from punchcard systems to more advanced systems like marksense or DRE systems, when these could be deployed. In the county of Los Angeles, the largest election jurisdiction in the US with 3.8 million registered voters, they continued to rely on their punchcard voting system in the late 1990s and 2000. For the 1996 US Presidential election, a variation of a punchcard system was used by 37.3% of registered voters in the United States [16]. In the 2000 US presidental election the votomatic punched card systems got a considerable bad reputation. In three of Florida's counties, after counting and recounting, it was claimed the punch card system's uneven use had even affected the outcome of the election [17]. With this, the US presidential election of 2000 brought the punch card systems into infamy and the US was critized of using such an old-fashion technology.

2.1.1.2 Optical Scan Systems

Optical scan is another technology that can be used for the counting of paper ballots, and was first applied to voting in the 1980s [18].

An optical scan system used for voting is made up of the following 4 components [18]:

- Computer-readable ballots
- Marking devices
- Privacy booths
- Computerized tabulation device

The votes can be cast using a marksense system, but also using a computer or direct-recording electronic voting system (described in section 2.1.2).

Marksense systems are using a paper ballot where the candidates and issue choices are printed next to an empty marking space (could be an empty square, rectangle, circle or oval). The voters cast their vote and record their choices by filling in the space and place the ballot in a sealed ballot box or feed the ballot into a computer tabulating device at the precinct. In figure 2.4 i have showed an example of a ballot were the voters choice has been marked. The optical-mark-recognition equipment is a device for reading printed or handwritten symbols or bar codes from paper and translate the information to bits. The device reads and counts the vote using "dark mark logic" were the computer selects and count the colored/dark spot [18]. The technology these optical scanners can be using is arrays consisting of tightly

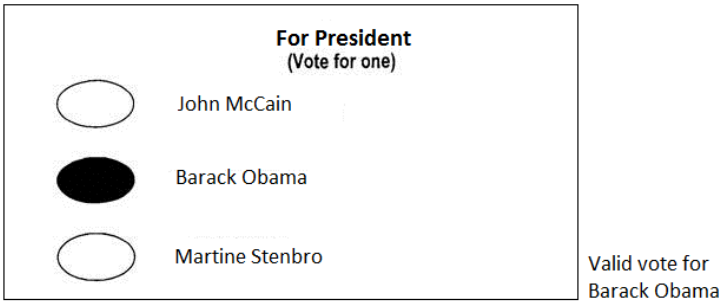

Figure 2.4: A ballot example were a choice is marked by filling the oval empty space.

packet rows of light receptors that can detect variations in light intensity, and then the software instructs the equipment to assign valid marks on the ballot to the proper candidate or issue. The software can be configured for instance to only allow one vote per person, and it should be programmed to detect over-voting, rejecting the ballot. An optical scan ballot should also support a choice for a write-in of a candidate's name (an empty line next to a marking space for write-in).

When using a computer, or a direct-recording electronic voting system on the precinct (section 2.1.2), the vote is cast electronically and a paper ballot with the vote is printed for counting purposes. The paper ballots can be printed with bar codes representing the vote which then is counted using optic scanners. The scanning and counting can be performed directly at the precinct or the paper ballots can be transported to a centralized location for the scanning. When performing the scan and counting at the precinct a mistake of over-voting can be detected and corrected by voter, while over-voting is detected at the central location no correction can be done. Figure 2.5 shows an optical scan tabulator at the precinct, with a sealed ballot box to keep a paper trail.

"Marksense systems were used by 24.6% of registered voters in the United States for the 1996 Presidential election, and their use were on the rise" [2]. In the 2004 election, about 32% of American voters used an optical scan paper ballot system [20]. After the issues with the votomatic punch card system in Florida for the 2000 election, for the 2008 presidental election all Florida counties were using paper ballots combined with optical scan machines [17].

Some of the advantages with using optical scan systems is that a marked ballot clearly shows the voter intent, there is no problem like "the hanging chads", and there is a paper audit trail for possible recounting by hand, for

Figure 2.5: An optical scan tabulator, with a sealed ballot box, at the precinct. [19]

instance if there is a close election.

Concerns about optical scans systems include errors related to the optical scan cards (ballots). Stray marks or incomplete markings could lead to misinterpretation and if using DRE machines to cast, printers running low on toners would be a problem and has to be controlled and maintained. Some other asked question is about reliability, the paper ballot can increase if high humidity and this can cause problems for the different sensors, and concerns about whether the scanners or software can be manipulated [21]. One issue that was found in the Sequoia Optical Scan system [22], was not directly associated with their optical scan equipment, but with the communication between the optical scanning equipment and the "master" system. The communication was not encrypted or authenticated so it could be intercepted and altered in transit by anyone who had gained access to the local network used in the system [23]. So even if the scanners at the precinct were correct and reliable, other factors of the system gave concerns.

2.1.2 Systems to Electronically Cast Ballots

There exists a wide variety of e-voting setups for electronically casting a vote, ranging from the use of an electronic device inside a polling station to casting the vote from a remote location using a telephone or a PC transmitting the vote via the Internet. In this section I describe some of the systems and schemes used for casting electronic ballots.

Figure 2.6: A photo of a DRE machine used at a polling station. [24]

2.1.2.1 Direct-Recording Electronic Voting Systems

A more recent invention used for elections is called direct-recording electronic (DRE) voting systems. This is an electronic implementation of the old mechanical lever systems, where the voters cast ballots by pulling down levers that correspond to each candidate or issue choice and each lever had a mechanical counter that recorded the number of votes for that position. The DRE machine was first introduced in the 1970s and is still used in elections [18]. Figure 2.6 shows a picture of an example DRE machine at a polling station.

The DRE voting system is a computer on the precinct with a screen to display the ballot and an input device in the form of push buttons or a touch screen [25]. The machine processes data with computer software and records voting data and ballot images in memory components. The DRE machines can also be used for just the casting of the vote, and then print a ballot for tallying if other electronic tabulation techniques like optical scanning is used.

The ballot information is prior to the election programmed into electronic memory storage and uploaded to the machine. The screen displays the electronic ballot and the voter registers his choice of vote directly into the system, as shown in figure 2.7. The vote is saved on an electronic storage medium together with the other votes cast on the machine. The tally can be performed at the precinct and the result transmitted, or all the votes can be saved on a removable storage medium and transported to a centralized location for tallying.

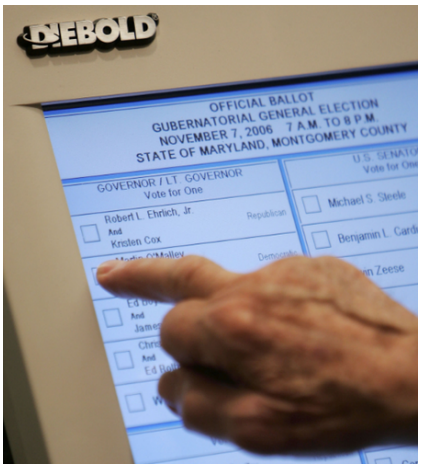

Figure 2.7: A display on a DRE machine showing the electronic ballot, were the voter can cast his vote using the touch screen. [20]

In a DRE system the ballot styles can be programmed for each precinct regarding the layout, the contests, candidates, pictures of candidates, bilingual options etc. Special options can be configured to support disabilities, for instance larger text or an audio option for visually impaired voters.

For more security features on a DRE machine, smart card technology can also be used [18]. The DRE machines can include a card cartridge system, for activation before access, and can have integrated circuit chips to process and store data - used to open poll and authorize voter access to ballots. When the voter inserts his card, the correct ballot is displayed on screen.

One advantage with the use of a DRE voting machine is that the vote is stored directly into the machines memory and saved for electronic tallying, and the DRE system can later also print a record of ballots cast to produce a paper audit trail if necessary (Voter Verified Paper Audit Trail (VVPAT)). Another advantage is that the voter can print a "receipt" after casting the vote to verify to himself that the correct vote was registered. One concern, as mentioned before, when not printing an audit trail, is that it is easier to lose an electronic ballot than a physical paper ballot if an error occurs.

In 1996, 7.7% of the registered voters in the United States used some type of DRE voting system, and in 2004 the number had grown to almost 30% [26]. Regarding the earlier Florida problems with using punch card system, in 2004 a number of Florida counties changed to DRE units [17]. Again they were criticized, this time of not providing any paper copy accountability of the DRE machines' reliability, and for the 2008 election they went back to

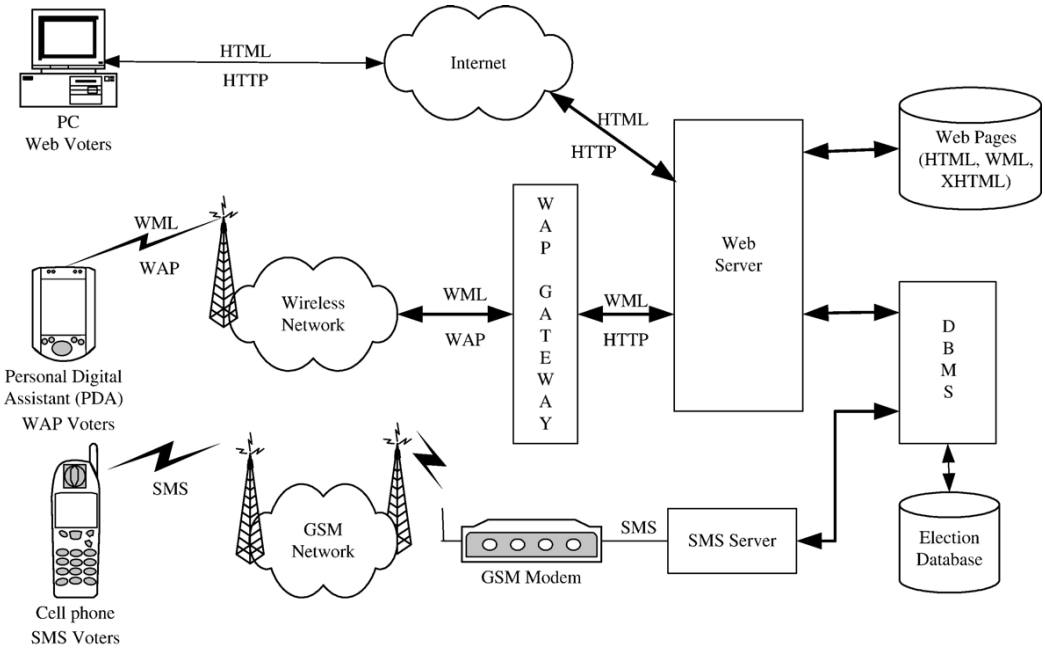

Figure 2.8: An example remote e-voting system [8].

using paper ballots read by optical scan machines.

2.1.2.2 Public Network DRE Voting Systems

As explained, a DRE voting system is an election system that uses DRE machines to display electronic ballots, and the voter cast his vote directly into the storage memory using for instance a touch screen. A public network DRE voting system can use a DRE machine (computer) at the polling station for the casting of the vote, but after the vote is cast the system can transmit the vote date from the polling place to another location over a public network [27]. By this the public network DRE voting system can utilize either precinct count or a central count method. The vote data may be transmitted as individual ballots as they are cast or as one batch when the polls close. The vote data may also be transmitted periodically as batches of ballots throughout the Election Day to support an updated result at all times. At the central location vote data from several precincts are added up. Public network DRE voting system not only includes casting the vote from a computer at the polling station, it also includes remote voting as Internet voting and telephone voting.

Remote Voting

Voting over the Internet can be done from remote locations using a computer connected to the Internet. The term Internet voting could also imply the use of traditional polling locations with voting booths consisting of voting systems connected to the Internet. But when referring to Internet voting in this thesis we mean votes cast from a remote location, for instance through the web browser of your home computer, via the Internet. These Internet voting systems are also called cryptographic voting systems.

Internet voting is a type of absentee voting which means the voter can use any personal computer with Internet connection to cast the vote, and it is sent to be stored in the election system [28]. This is regular Internet users with personal computers installed with standard operating systems and software. To vote over the Internet the voter needs a digital signature to log into the system, for instance identify himself with a PIN code and/or a smart card and then the particular ballot for an election he can participate in is showed. The voter submits his choice and the encrypted ballot is transmitted over the Internet to a remote server (an electronic ballot box) of the election system. An overview of am example remote e-voting architecture is showed in figure 2.8.

As shown in figure 2.8, remote voting schemes can also be designed so a phone can be used to cast votes via the GSM network [8]. A telephone voting scheme can be based on the voter calling to cast a vote or voting with SMS [29].

An example of a telephone voting scheme could be a voter calling the number of a vote collector, where an automatic answering message informs about the voting procedure. The voter authenticates keying in a PIN code, and then makes his choice by keying in a code corresponding to for instance the candidate he is voting for. After the voter having made his choices the system can recite the voter's options, so the voter can choose to change or confirm his choices.

SMS is another mean of casting votes over the GSM network [29]. This is a non-interactive method compared to the "calling method", where voting is performed in one text message. A "SMS vote", in this example scheme, had to include the voter's code, a district code and the code to the candidate voting for. After transmitting a valid vote, the system sent back a SMS confirming the vote was recorded. The receipt did not say anything about which candidate the voter voted for.

Cell phones could also be used as a supplementary channel, in addition to the Internet, for instance to receive codes for authentication or for verification. In the Norwegian election system described in chapter 3 the cell phone is used for exactly these operations.

The area of remote voting is growing in popularity and the last decade several countries have developed and are testing the use of Internet voting. Systems for Government elections has been developed in the UK, Estonia and Switzerland and for party primary elections in the US and France [30]. In Switzerland Internet voting is already an established part of local referendums and voters get their passwords to access the online electronic ballot through the postal service.

In Estonia most voters can use Internet voting, if they want to, both in local and parliamentary elections. They had a pilot project of Internet voting for the municipal elections in 2005, and now almost everyone on the electoral roll has access to the e-voting system.

Internet Voting in Estonia

The user authentication for the Estonian voting system builds on the country's national ID card. The national identity card used in Estonia is a smart card equipped with a computer-readable microchip, and by this not only function as a paper document but also as an electronic identity [11]. The eID cards enable secure online authentication and is used to get access to the online ballot in an election. All a voter needs is his identity card, a computer, a card reader and PIN codes associated with his identity card to be able to vote from a computer anywhere in the world. The voter inserts the card into their computer, with a suitable adapter, and it allows them to authenticate to a government web site over an SSL-encrypted channel. PIN1 is used to authenticate the voting person, while after the voter has made his choice a second pin PIN2 is used to confirm the choice and sign. Electronic votes over the Internet can only be cast during the days of advance voting and on Election Day the voters, that has not voted electronic, have to go to polling stations and fill in a regular paper ballot.

To protect privacy and a free expression (against coercion), a voter can replace his vote as many times as he wants, with only the last vote actually being tallied [11]. By being able to cast mulitple votes during a period, this can provide some resistance against bribery and coercion attacks. A resistance against network attacks like a man-in-the middle attack, can be

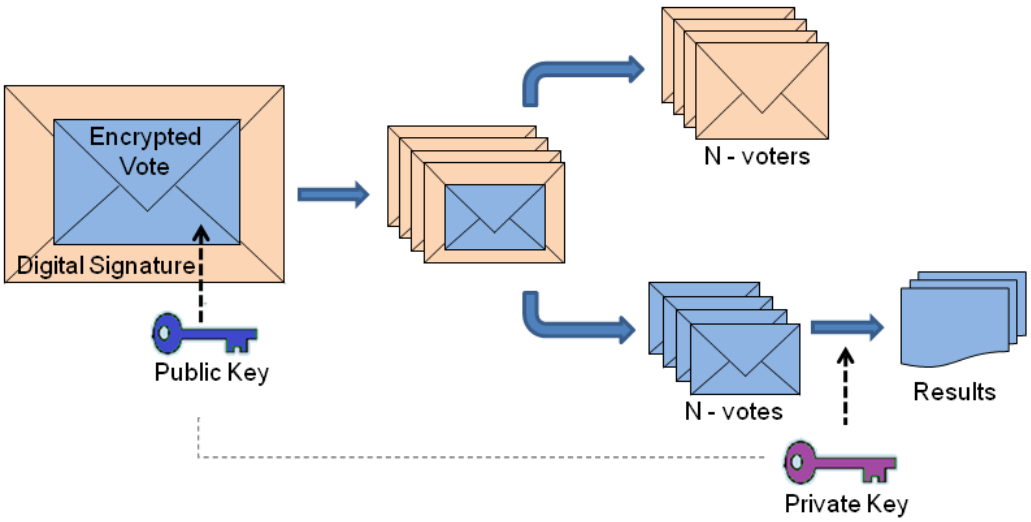

Figure 2.9: The "double envelope scheme" used in the Estonian voting system [11].

provided by the use of Secure Sockets Layer (SSL).

Secrecy of the Estonian voting system is based on a "double envelope" scheme, shown in figure 2.9, where the voter's choice is encrypted by the voting application and then signed digitally. Works like the voter seals his choice into an inner blank envelope and then puts the inner envelope into another envelope which he writes his name and address on. All the signed and encrypted votes are collected at at central site to check and make sure only one vote per voter is counted. After this control the digital signatures with personal data (the signed envelope) are removed and the anonymous encrypted votes (the last ciphertexts of each voter) are mixed together in a simulated ballot box. After mixing, these ciphertexts are sent to a tallying component for decryption and counting.

The scheme uses public key cryptography and the National Electoral Committee holding the private key, opens the encrypted votes collegially on the Election Day.

The Estonian voting system shows no protection against compromised client platforms, which I will describe later is one of the significant security problems with remote Internet voting. The Estonian system does not really have countermeasures against any corrupt components. If the voter's computer is compromised the ballot information is visible to the attacker, and the attacker has the possibility of changing the votes [31]. If the ballot box that

collect/mixes the electronic votes is compromised, ballots can be deleted or maybe fraudulent ballots can be inserted. In worst case, if the tallying device is corrupt it can decide the entire election result.

The VOI Pilot Project in the US

Another example of an Internet voting system was a pilot program called Voting over the Internet (VOI) that was carried out in 2000 by the Federal Voting Assistance Program (FVAP) [3]. Their project was a test to see if votes could be cast in a reliable and secure manner using the Internet, and only included 84 volunteers in 21 states and 11 countries [30]. To make sure, and assure the volunteers, that their votes would be counted in the case of a failed experiment, each volunteer was also allowed to cast a traditional paper-based absentee ballot. The system was not designed to tabulate votes, only imitate established absentee ballot and cast these electronically over the Internet.

To carry out the pilot each of the volunteers received a CD with a browser plug-in so they could display and transmit ballots to the FVAP servers. The Department of Defense (DoD) controlled a digital certification program to authenticate the voter identity, which they deactivated once a voter transmitted a ballot to prevent multiple casting of votes. The entire ballot, except the destination, were encrypted and then transmitted over the Internet to the FVAP server. The FVAP server was protected by keeping it in a secure location with very limited access and uninterrupted power supply. In addition, two intrusion detection systems (IDS) were installed to monitor and detect any malicious activity.

At the local election sites the election officials used terminals to access a local election official (LEO) server that was connected to the FVAP server. The FVAP server transmitted the encrypted ballots addressed to that specific LEO site over the Internet, and once the ballots arrived, a computer at the LEO site could decrypt the ballots and print paper copies.

The VOI project was cancelled in 2004 due to concerns about security issues, and was never implemented. The issues were regarding the voter anonymity being compromised or hackers intercepting and manipulating the ballots sent over Internet, and the Department of Defense would not approve of implementation.

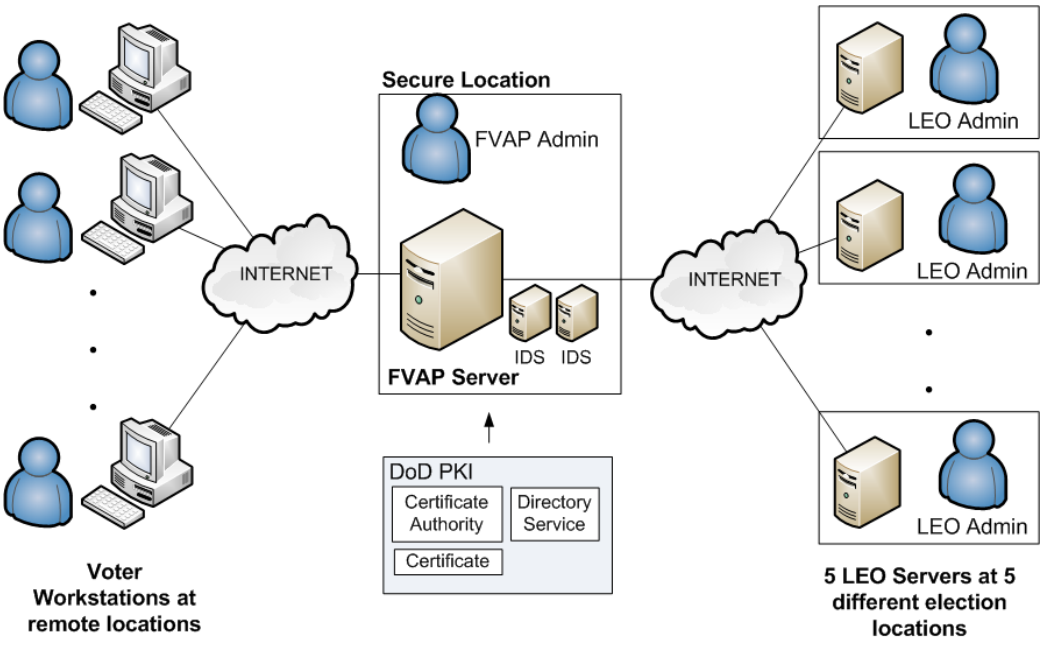

Figure 2.10: Overview of the VOI system by FVAP [3]

2.1.3 End-To-End Verifiable Voting Systems

The newest development in the area of voting systems is end-to-end (E2E) verifiable systems, also called open-audit voting systems. The last few years have witnessed the emergence of end-to-end voting systems, which enable voter-verification of election outcome [32]. The purpose of E2E systems is primarily to improve election integrity through E2E verifiability.

E2E verifiable system are often a combination of paper and electronic methods to reach their solution of providing verification. Some examples of E2E systems using paper ballots are Pret a Voter [33], Punchscan [34] and Scantegrity [35]. Each of these systems provides its own strengths and weaknesses based on its implementation and approach to the problem of verification. They aim to provide a method of verification and still be usable in any election scenario. A remote Internet voting system would most likely be a cryptographic E2E system without a paper trail, and the type of E2E system I will describe further.

The cryptographic voting protocols offer the promise of verifiable voting without needing to trust the integrity of any software in the voting system. E2E systems are voting systems with strict integrity properties and strong tamper-resistance [4]. The idea of a E2E verification is showed in figure 2.11.

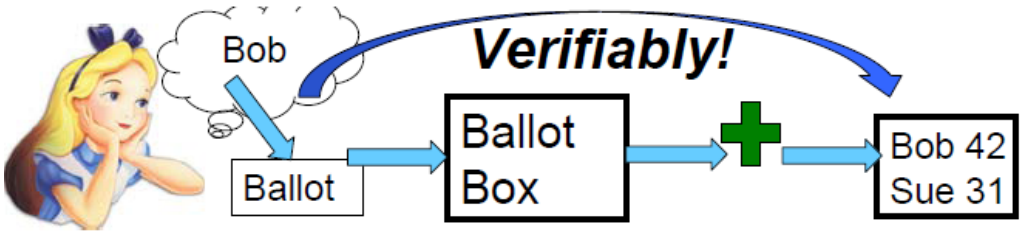

Figure 2.11: E2E verification: Alice being able to verify her vote for Bob is in the final election result [4].

These E2E systems use existing cryptographic methods to generate receipt and proofs, giving the voter a possibility of verifying their votes. The voters by this mean of receipt codes and proofs also verify their votes have not been modified by corrupt actors and is accurately recorded in the election system. By the use of different cryptographic methods this verification is possible without revealing for the system or outsiders which candidates were voted for. In open-audit voting systems there is also possible for anyone to verify that the ballots are counted as cast [4]. Since these systems not using paper receipt they have to include satisfying measures as the following.

Before submission to the election, a ballot is encrypted. An E2E system gives the voter a possibility of verifying the encryption, to verify the vote is recorded as intended. Based on the encryption process the system can also provide the voter with a receipt of his recorded vote. To also verify a vote is collected as intended, the collection of cast ballots has to be public. Most E2E proposals (few has been implemented) displays a collection of cast ballots on a website that the voters can access and check that their ballot is in the list and correctly in the list. To be able to use the measure of a bulletin board of cast votes the ballots somehow has to be identifiable to the voter. An additional feature that an open-audit voting system has to include is the possibility of verifying the collected votes are counted/tallied correctly. And this should be possible to do, not only by a participating voter, but by anyone (any external person with interests in the correctness of the election). An open-audit system can use a mixnet (described in section 2.5.5) to decrypt the collection of ballots and then tally the decrypted ballots. A mixnet ensures that a voter can not be connected to a certain vote when decrypting, and in addition a mixnet allows anyone to check the correctness of the decryption. By providing these measures described, an E2E verifiable voting system can provide a "chain of custody" without offering any paper audit trail [4]. A describing figure of a such a system with universal verifiability is showed in

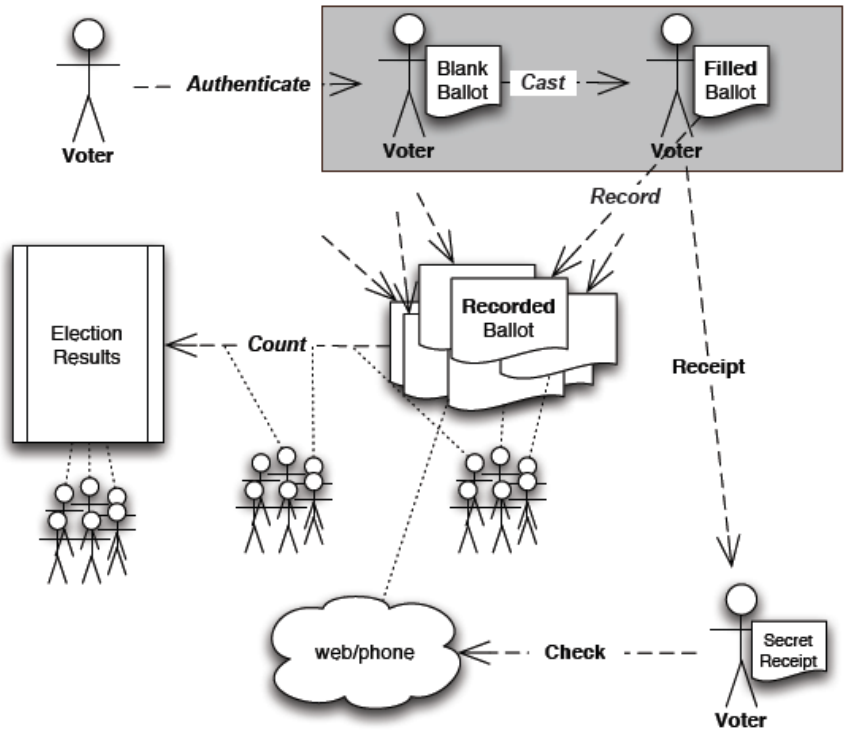

Figure 2.12: A system where the voter can audit all steps of election (universal verifiability) [36].

figure 2.12 where you can see the arrows showing which parts the voter can audit.

E2E voting systems is a hot discussion topic now witht the development of Internet voting systems, and NIST recently arranged a 2 days workshop in the field [32]. Several topics regarding E2E verifiable voting systems were presented, including requirements and desired properties of such systems. Integrity, privacy and accessibility are requirements, while the paper trail a detail being discussed to be or not be integrated into the system. Properties of an E2E verifiable system also brought with it several issues for discussion. Security schemes providing ballot secrecy together with auditability are necessary, and at the same time provide usability for every participant in the system (voter, poll worker, auditor, etc.) It was discussed how electronic end-to-end voting systems look like, and a couple systems having been deployed and testet were also presented, including the Helios voting system.

NIST has also requested feedback from government and election authorities about the interest in such systems among both voters and election officials.

The purpose of the works shop was to "begin a discussion on these matters, including experts in the different fields of cryptography, security and usability, and encourage research on the development of electronic end-to-end voting systems with an emphasis on usability" [32].

As mentioned there currently exists some E2E verifiable election systems at various levels of implementation. An example of an open-audit voting system which has been implemented and actually deployed is the cryptographic Helios voting system described in chapter 4. The Helios system aims at specific target elections, and does for instance not take the problem of coercion into consideration.

2.2 Motivations of an E-Voting System

The motivations behind electronic voting are many. An improved system for counting ballots would speed up the process of counting, quicker display of partial or final result, and could reduce the errors associated with manual counting. People make errors, while computers don't, if configured correctly.

The use of electronic means to cast a vote has many advantages. Using an electronic voting machine at the precinct would reduce the use of paper ballots, as the machine displays the ballot electronically. It would also make it easier to prepare special ballots for other languages or visually impaired voters programmed into the system, instead of printing out several options.

The newest development in the area of e-voting is using the Internet for remote voting. By making the voter able to vote from his home computer one of the goals is to improve the accessibility for disabled voters, as they don't have to actually go to the polling station. The overall participation would probably also increase because of the easiness of voting from home, and it would maybe be more appealing to the youth doing the voting electronically.

The use of Internet for voting gives advantages both when it comes to speeding up the calculation of election results and regarding disabled voters. Remote voting also makes it possible to vote from your home computer, making it easier for disabled voters, and the electronic ballots can be adapted for people with a visual handicap (as with DRE-systems). This can increase the total participation in elections, not only for disabled voters but also for

absentees, for instance people on travel, in the military, students at college or abroad, etc.

As we see the use of computers for voting has many advantages, but a system for electronic voting requires means to preserve every aspects of a traditional voting scenario when it comes to security aspects like authentication, secrecy and anonymity. The system has to prevent attacks, errors and any electronic fraud.

2.3 Requirements of a Modern E-Voting Solution

A traditional voting scenario includes phases of preparing the election, setting up the lists of candidates and print ballots, identification/authentication of voters coming to the polling station to vote, checking the voter to the electoral roll, install the voting booth to let the voter cast his vote secretly and anonymous and be able to verify it (receipt), the secure transportation of all the votes (the sealed ballot boxes) to a secure location for counting, and off course for all this we need trustees (people we trust doing this, election officials). Then there is the process of tallying where they have to make sure the counting and the results are correct, and keep an audit-trail for a possible recounting.

The requirement and challenge is to maintain some and improve some of these "features" in the electronic voting scenario. An electronic voting system with voting over the Internet should fulfill certain requirements and have features similar to a traditional voting system. The desirable properties of such voting systems is security, by auditability and ballot secrecy, as well as usability and accessibility. The step of trustees mentioned in a traditional voting scenario can be improved by involving complete mechanisms of verification, where the voter don't have to trust election officials or even system components (as described with E2E verifiable voting systems earlier). An election model should summarized include [37]:

- Election Set up
- Ballot Casting and Recording (with audit possibilities)
- Ballot Tallying (with audit possibilities)
- Election Audit(s)

Table 2.1 shows the criteria a successful election system in the US should meet according to a report from the National Workshop on Internet Voting [38].

Eligibility and authentication	Only eligible voters are allowed to vote
Uniqeness	The eligible voter can only vote once (have one valid vote)
Accuracy	Votes are recorded correctly
Integrity	Votes cannot be altered or deleted
Verifiability and accountability	The system has the ability to verify that votes are correctly counted
Reliability	The system should work without compromising votes, even if system failures occurs
Secrecy and non-coercibility	Votes should be secret and a voter should not have a record of voting choice
Flexibility	The system should be usable by different type of voters (support multilingual ballots, accommodate handicaps by audio or visual features)
Convenience	The voting process should be convenient
Certifiability	The system should be testet by election officials
Transparency	Voters should be able to understand the system generally
Cost	*The system should not be too expensive*

Table 2.1: Requirements of an e-voting system [38].

"A real-world Internet voting system has significant functional constraints [1]." Not only, as mentioned in table 2.1 should the voting process be convenient, it is a functional constraint that the voter should not have to interact with the voting system more than once to submit a ballot.

Another functional constraint, not mentioned in the table is regarding performance. Not only has the system requirements of accurately recording ballots, ensuring the integrity and secrecy of the ballots, etc., it also has to keep up in performance. Most ballots will probably be submitted during peak hours, but they still have to be processed quickly. And also, once the ballot box closes, the system has to provide and make the result available as soon as possible.

In a voting system, an issue of great significance is trust. An important part of creating a voting system is that it has to gain trust among the voters.

If the property of trust is not satisfied, there is no point of creating such a system. The success of a voting system relies on the public trusting the system. According to a presentation at a NIST workshop, to trust a voting system is the same as assuming that [39]:

- Procedures are followed as intended, count is correct
- A secure chain of custody is provided
- Error free Software
- Secure Hardware
- Secure Cryptographic Algorithms
- Trusted specialized user interfaces

With the introduction of E2E verifiable voting systems, especially cryptographic E2E verifiable systems, the statement "I trust the voting system" is not sufficient. The possibility of verifying/audit any process of the voting system is a significant requirement to emphasize. The Goals of a secure verifiable voting system is more like the following:

- Cast-as-Intended
- Counted-as-Cast
- Verifiability
- One voter, One vote
- Coercion Resistance
- Privacy

The end-to-end verifiable voting systems not only has to include earlier mentioned requirements of voting systems, it also has to include several processes of verification means. The voter should be able to verify that his vote is cast as intended, by auditing the process of encryption, that the vote is counted as cast, by verifying his vote is recorded in the system and by verifying election tally. Since the development are heading against remote voting systems the coercion issue is a problem widely discussed, but as in earlier voting systems it has to be taken into consideration. We need receipt free verifiable elections.

A requirement that is not specifically included in table 2.1 is the requirement of usability. A requirement of flexibility, regarding languages settings is included, and it is said that the voting process has to be convenient. There

is important to emphasize the requirement of usability. In an presentation on the E2E NIST workshop a presentation specifies two desirable properties of a voting system [4]:

- Security (Including auditability and ballot secrecy)
- Usability and accessibility

In usability of a systems lays the properties of the systems learnability and efficiency. Other important factors is the users satisfaction when interacting with system and if any errors occur in any steps or processes.

When having auditability as a requirement of a voting system, this introduces usability requirements. It is the voter that will be the one auditing his vote to ensure the correct functionalities of system components, and the system has to be usable for this even though the voter does not have any computer knowledge beyond *normal*. The learnability should be good by guiding the voter through steps of auditing. There is a tension between usability and auditability. The voter would probably need to perform some extra tasks to enable auditability, and at least these steps should be convenient and efficient.

I only mentioned the usability vs auditability issue regarding the voter, but there is 3 types of users the system should provide good usability for [4]. In addition to the voters, the system should provide a user-friendly setting for the poll workers/administrators of the elections. It is also desirable that public auditors audit the system and the elections, and to request this it is a user-friendly environment has to be provided.

2.4 Security Threats of a Modern E-Voting System

As all information systems, an e-voting system is also vulnerable against computer attacks. Although Internet voting may improve several election factors, there are concerns that the benefits are outweighed by the issues of many potential security threats. The security flaws are often concerning the voter's home computers, and that these are the weakest link because people do not keep their personal computers secure [40]. "Everybody" has experienced some kind of virus infection to their personal computer. In this section I have described some of the most frequently cited security threats

to Internet voting, both related to the voters computers and the system [41, 40, 42].

Malicious Payload is a security threat to the voter's personal computer. The malicious payload is software or configuration designed to harm and could be a virus, worm, Trojan horse, or a remote control program which is maybe the biggest threat in a voting scenario. If a malicious program is installed on a voter's computer it could secretly change the vote. The owner of the computer might be unaware of even having a malicious program installed because these programs can be difficult to detect (run in stealth mode). Malicious programs like these have advanced in sophistication and automation the past years in a way that they can do more damage, is more likely to succeed and disguise themselves better. Even though an Internet voting system has strict protocols for encryption and authentication, the malicious code can do its damage before these other security features are applied to the data [43].

Spoof sites are malicious web sites that are created to look like legitimate web site, and in a voting scenario we understand that this could be really bad, as the site could be used to launch phishing attacks to collect voters' credentials like a PIN or a password needed to cast a vote. The web site can look exactly like a government voting site, but redirect the voter's browser to a malicious web server. There are several ways that an attacker could spoof a legitimate voting site. One way could be to send email messages to users telling the users to click on a link which would then bring up a fake voting site were the adversary could collect the user's credentials, steal the vote, and then use this to vote differently. An attacker could also set up a connection to the legitimate server and then feed the user a fake web page, acting like a man in the middle, transferring and controlling all the traffic between the user and the web server. By transferring all the information between user and server, the user's vote can be altered before further sent to the server [43].

Another threat to Internet voting is *Denial-of-service (DOS) attacks*. DOS attacks are carried out by automatically sending a flood of messages to a Web site, a server, over a channel or similar, to make it crash or decrease quality because it cannot handle all the generated traffic. By using a distributed DOS Attack (DDOS), attackers can cause routers to crash or election servers to get flooded, or it is possible to attack a large set of hosts for instance targeted demographically to cease the function of the election [43]. This can be a significant threat to the Internet voting if for instance the voting occurs on a single day. It is important to have extra bandwidth to handle the traffic and as some of the voting systems I will describe later, the voting can occur

over several days in advance of the election.

As mentioned, not only the users host machines can be attacked by malicious intentions, also the election server could be the object of attacks. The server could not only be attacked by various malicious programs and software intruders like a DOS attack, but it can also be affected by physical intruders and physically factors like power outage.

2.5 Cryptographic Theory

In an election system, cryptography can be used to protect the communication between the user's browser and the election server, to ensure privacy, and for verification purposes. The technology of cryptography is relied upon to ensure integrity and confidentiality of network traffic [43]. In this section I have described some background on cryptographic techniques and protocols used in electronic voting systems.

2.5.1 Symmetric Encryption

Symmetric encryption is a cryptosystem that uses the same key to perform both encryption and decryption [44]. Symmetric encryption transforms a plaintext into a ciphertext using a secret key and an encryption algorithm. The other way around, to decrypt the ciphertext, a decryption algorithm and the same secret key is used.

In this encryption scheme Alice and Bob share a secret key that they use for encrypting a message before sending it and decrypting a message when receiving it. [45] When Alice wants to send Bob a confidential message she transforms the plaintext message to ciphertext using a symmetric algorithm and the secret key. Alice transmits the ciphertext to Bob who then uses the same secret key they share and decrypts the ciphertext back into plaintext. The scheme is showed in figure 2.13.

The symmetric algorithms (i.e. AES) are primarly used to achieve confidentiality, as shown in figure 2.13, but may also be used for authentication and integrity. "Symmetric algorithms are ideally suited for confidentiality, and modern symmetric algorithms, such as AES, are very fast and strong" [44].

As mentioned symmetric algorithms can also be used to authenticate the integrity and origin of data sent from one place to another. This scheme

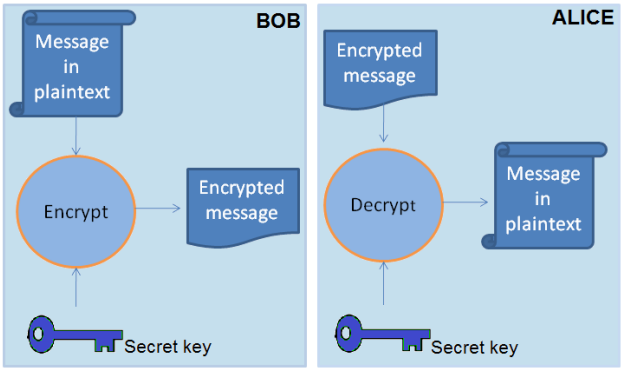

Figure 2.13: The symmetric encryption scheme [44]

works similar as the one for confidentiality. Alice still uses her secret key to generate a ciphertext for the entire plaintext like in figure 2.13. A fixed size of this ciphertext is then added to the plaintext, before transmission. [44] This portion of the ciphertext works like a message authentication code (MAC). When Bob receives the message from Alice, he can control the integrity of the data by checking the MAC. To do this Bob uses his copy of the secret key to generate the ciphertext and select the same portion of the ciphertext (MAC) and compare this to the MAC he received in the message from Alice. If the two MAC's are matching Bob is:

- assured the message is really coming from Alice, because no one else knows the secret key to prepare a message with the proper MAC.
- assured that the message has not been altered, because then Bob's generated MAC of the received altered message would not match the received MAC. We assume the attacker altering the message is not knowing the secret key and cannot alter the MAC to correspond to the altered message.

To use the symmetric encryption scheme and the MAC property, Alice and Bob need to share a secret key before any of them encrypts and send a message or generates a MAC for a message. This is called the key exchange problem or the key management problem.

2.5.2 Hash Functions

A hash function maps variable-sized amount of data, for instance a message, into a fixed-size output through a one-way mathematical function. [45] The

result is called the hash value or hash code, and can be thought of as a fingerprint of the data. The hash function can for instance take a message M as input and produce the hash code which then is a function of all the bits of the message.

The hash code can serve as the message authenticator, because if any of the bits in the message is changed, the hash value produced will be different. [44] By this hash values can be used to verify/authenticate date and for message authentication like the MAC code.

The message digest, the hash value of the message, can be reproduced by any party with the same stream of data, but it is virtually impossible to create a different stream of data that produces the same message digest. By this the hash value can also be used to provide integrity.

If Alice wants to send a message to Bob and wants to give Bob the possibility to protect against any changes in the message, she can send him both the message and the message digest. Then Bob can use the message and a secure hashing algorithm to recompute the message digest and compare it to the one he received. This protects against accidental changes in the data, but not an potential attacker intercepting Alice's message changing both message and digest. [45]

To protect against any interceptors, the secure hash can be used together with a shared secret key to create a hash-based message authentication code (HMAC) similar to the MAC. Then Alice can send both the message and the HMAC to Bob, and since no attacker knows the secret key, Bob can recompute the HMAC to protect against changes in the message.

Standards used are the Secure Hash Algorithms (SHA1, SHA256, SHA384, SHA512) [46].

2.5.3 Public Key Encryption

The concept of public key cryptography was introduced in 1976 by Whitfield Diffie and Martin Hellman to solve the key managment problem of asymmetric encryption. [47] "Public key encryption, also called asymmetric encryption, is a cryptosystem in which encryption and decryption are performed using different keys - one public key and one private key. [44]" In stead of having the sender and receiver of a message share one key for encryption and decryption, as in the secret key scheme or symmetric encryption scheme, two different keys are constructed using a certain algorithm. These two bit

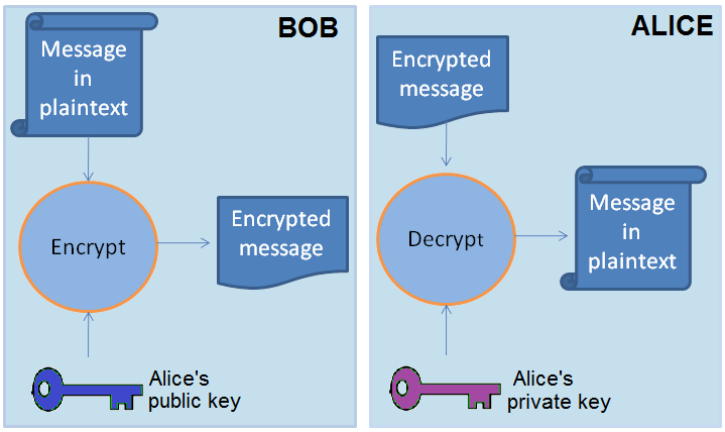

Figure 2.14: Public key encryption scheme [44].

patterns forms the key pair, where one of these keys is the public key while the other one is a key that is kept secret and never shared or transmitted. Public key encryption is based on mathematical functions and the private key is always linked mathematically to the public key. But deriving the private key from the public key is made as difficult as possible, for instance requiring the attacker to factor a large number, to make it computationally infeasible to derive the private key.

The RSA cryptographic algorithm meets the requirements of a public key cryptosystem and is often used. A RSA public key cryptosystem can provicde confidentiality, authentication and key exchange [44].

In the encryption process the plaintext is transformed into ciphertext using one key and an encryption algorithm. The plaintext can then be recovered from the ciphertext using the paired key and a decryption algorithm. Public key cryptography can be used to provide confidentiality, authentication or both [44]. More detailed the public key scheme works as follows:

If Bob wants to send a message to Alice that should only be readable by Alice, Bob can encrypt the message with the public key of Alice to ensure confidentiality. As showed in figure 2.14 the message can then only be decrypted using the secret key of Alice.

For authenticity Bob wants to be able to verify that the message is really coming from Alice. To achieve this, Alice has to encrypt the message with her own secret key, creating a digital signature. If Bob is able to decrypt the message with Alice's public key he can be sure the message is coming from Alice, and not anyone malicious disguising like Alice. The scheme is showed

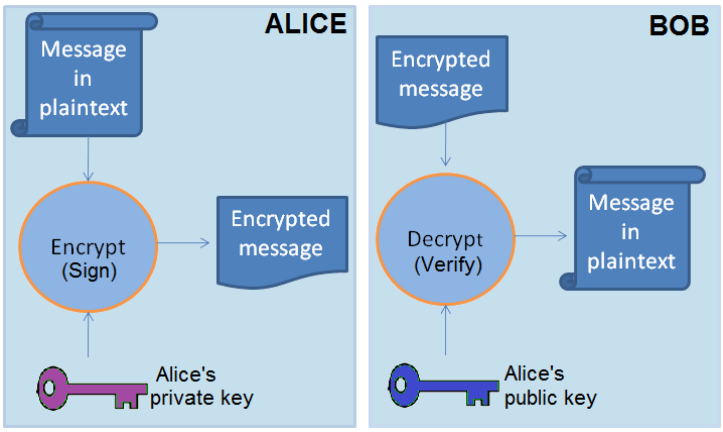

Figure 2.15: Public key encryption scheme for authentication by encryption [44].

in figure 2.15

Digital Signatures

As mentioned in the Alice and Bob scenario in 2.5.3 figure 2.15, a public key cryptosystem can be used to provide digital signatures. A digital signature scheme is a method for the receiver to verify and confirm the authenticity of a message received over an insecure channel. It enables the creator of a message to attach a code that acts as a signature, before transmitting the message [44]. The signature works as a proof of the source and guarantees the integrity of the message received. Digital signatures can also be used to prove the integrity of electronic votes, electronic ballot boxes or other data that has to be transferred during an election process.

According to [44] the digital signature should have the following properties:

- Verify the author and date and time of the signature.
- Authenticate the contents at the time of the signature.
- Be verifiable by a third party.

Since the signature must be a bit pattern that depends on the actual message and has to use a factor unique to the sender, a way to form a digital signature is by taking the hash of the message and encrypt it with the creator's private key. The process of producing digital signatures and to recognize and verify them should also be relatively easy, to make this technique efficient to use.

The digital signature sheme and the encryption algorithm has make it computationally infeasible to forge a digital signature, either by constructing a new message for an existing digital signature or by constructing a fraudulent digital signature for a given message [44].

Examples of digital signature schemes is the Digital Secure Algorithm (DSA) and the Digital Signature Standard (DSS) by NIST [44].

Digital Certificates

To verify that a public key really belongs to an individual or organization, digital certificates also called public key certificates, can be used [29]. These certificates are often issued by a trusted third party, called a certification authority (CA). It can be seen as an electronic document which uses a digital signature to associate a public key to an identity.

A digital certificate typically includes the public key and information about the identity of the party holding the corresponding private key. In addition also the operational period for the certificate and the CAs own digital signature are included. [45] The certificate might also contain additional information regarding the signing party or the recommended use for the public key. Certificates may be revoked, meaning it is cancelled, for example, if the owners private key has been lost or suspected compromised, or if the user's relationship with the organization has changed [45].

An individual user or business entity can contract with a CA to receive a digital certificate verifying an identity for digitally signing electronic messages.

Public Key Infrastructure

In a Public Key Infrastructure (PKI) the techniques of public key encryption is used for encryption and digital signatures [29]. Public key cryptography supports several security mechanisms (confidentiality, data integrity, authentication, non-repudiation), but to successfully implement these mechanisms there should be an infrastructure to manage them [48]. Implementing a PKI is providing mechanisms to ensure trusted relationships to be established and maintained. The most important function of a PKI is probably the distribution of keys and certificates used for security and integrity purposes. Integrity can be accomplished by binding a unique digital signature (explained in 2.5.3) to an individual and ensuring the digital signature can not be forged, it enables other individuals to verify this public key (digital

signature) binding and the individuals can again digitally sign data to ensure integrity of its content.

"A public key infrastructure is a combination of software, encryption technologies and services that enables an enterprise to protect the security of their communication and transaction on networks" [45]. Public key cryptography, digital certificates and certification authorities can be integrated into a security architecture for an entire enterprise, making the mechanisms of secure communication (key distribution, authentication, integrity and confidentiality) easier.

A PKI is constructed by the following componentes [45]:

- Certification Authorities
- Registration Authorities
- PKI repositories
- Archives
- PKI users

A *Certification Authority* (CA) is the component issuing public key certificates for each identity [44].(Recall digital certificates in 2.5.3). By this CA confirms the identities of the communicating parties sending and receiving data. CA also process and publish something called certificate revocation lists (CRLs), which are lists of certificates that has been revoked [45]. These lists are also signed by the CA that issued them.

"A *Registration Authority* (RA) is an entity that is trusted by the CA to register or vouch for the identity of users to a CA" [45].

A *repository* is a database in the PKI containing all the active digital certificates the CA has issued. The repository provides the data allowing users to confirm the status of the digital certificates for individuals or businesses that receive digitally signed messages. These users are called relying parties. (The CAs post certificates and CRLs to the repositories). [45]

The entities defined as *PKI users* are organizations or individuals using the PKI [45]. These are entities that not issue certificates, but rely on other components of the PKI to obtain certificates and verify the certificates of other entities they are communicating with. The end entities include the relying party and the certificate holder. The relying party relies on the certificate to know the public key of another entity. The certificate holder has been issued a certificate from the CA and can digitally sign data. For

various application an entity can serve different roles (for some applications be the relying part and for its own services be the holder of a certificate).

Above I presented an overview of the components in a PKI to understand the idea of the infrastructure, and I will not go further into the details regarding their operations.

There exists different PKI architectures. "A PKI is typically composed of many CAs linked together by trust paths" [45]. The certificate holders can obtain their certificates from different CA's depending on organization. A trust path links a relying party and one or more trusted third parties, to always ensure the validity of the certificate. In the next section i briefly describe an example PKI.

The X.509 Standard

An example of a PKI is the X.509 standard for public key infrastructure [49], which is the PKI standard to be used by the Norwegian e-voting system. X.509 works as a framework so that a public key of an entity can be obtained and trusted. The key can then be used to encrypt information to be decrypted by that entity, or for verification of an that entity's digital signature [49].

The X.509 standard has a system of certificate authorities (CAs), and these CAs issue certificates by binding a public key to a specific entity. These X.509 certificates bind a public key to an identifier, e.g. an username, an user number, web address or similar, which points to the PKI user with the corresponding private key.

In a X.509 PKI the end-user (PKI user) generates his own pair of public keys (one private and one public), and then the public key is made available to all other end-users in the PKI. The private key is stored secretly (in an encrypted file) on the PKI user's computer.

Since all PKI users know each others public key, they can ensure secrecy when sending a message encrypted with a users public key that can only be decrypted by the intended receiver. The other way around, an end user can digitally sign a message or document with his secret key, and the other users can trust this came from the claimed user. The X.509 infrastructure with the CAs works as a trusted third party, so all the users of the PKI can be sure everything is as it appears. It is also assumed that each PKI user knows the public key of its CA [49].

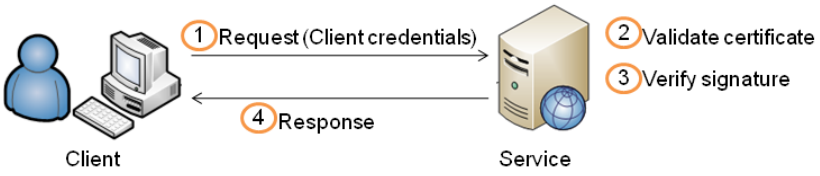

Figure 2.16: Authentication by X.509 certificates [50].

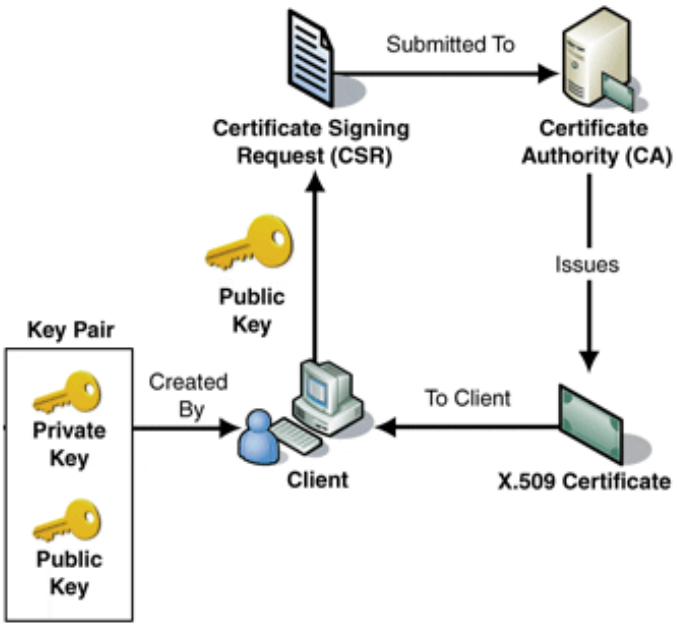

Figure 2.17: PKI user requesting a X.509 certificate [50].

The authentication scheme, using X.509 certificates are shown in figure 2.16. Before the authentication process can be performed the CA has to have issued certificates for both parties, a process shown in figure 2.17. In the scheme a client (person or application) wants to gain access to a web service that requires authentication prior to authorization [50].

To authenticate, the client sends a message with its credentials, digitally signed with its private key corresponding to the public key in the certificate (1). The service then validates the certificate with a number of checks (2). The validation includes checks to verify that the certificate has not expired and that it has not been tampered with (the certificate content is checked against the signature of the issuing CA). The CA having issued the client's

certificate is also verified by comparing the issuer signature on the client's certificate with the certificate of the issuing CA. In addition it is controlled that the issuing CA has not revoked the certificate (that it does not appear on the CRL published by issuing CA). Then the service verifies the client's signature with the public key in the client's certificate to ensure the data sent has not been tampered with (3). Now the client is authenticated to the service, and a response can be sent back to the client (4).

2.5.4 Homomorphic Encryption Functions

An homomorphic cryptosystem have the property that the (encrypted) sum of encrypted values may be found without decrypting them. An encryption function $E()$ is homomorphic if it is possible from the given $E(x)$ and $E(y)$ to obtain $E(x \perp y)$ without decrypting x or y for some operation $\perp$.

$$E(x) \perp E(y) = E(x \perp y)$$

An example of an encryption function that has multiplicative homomorphism is El-Gamal. If the encryption of a message $m1$ and the encryption of another message $m2$ is multiplied together, the result is the encryption of the product. What happens with El-Gamal when two encrypted messages are multiplied together, is that the randomization gets added up in the exponent and the messages get multiplied.

$$E(m1) * E(m2) = E(m1 * m2)$$

Example of two homomorphic cryptosystems are RSA and El-Gamal [51].

With RSA for instance, after encrypting a plaintext P into ciphertext C, you can multply C with 2 and then decrypt 2C and you would get 2P This would not be possible by a normal symmetric cipher (DES, AES) becuase multiplying an AES ciphertext with 2 and then decrypt it would give you some random nonsense, not P.

This is homomorphism with respect to multiplication. No standarized fully homomorphic cryptosystem with respect to both multiplication and addition has yet been created , so in a voting scenario were addition is the desirable property, a variant of the El-Gamal cryptosystem can be used [52].

The El-Gamal Crypto Protocol

El-Gamal is a cryptosystem based on the difficulty of the discrete logarithm problem, and is named after its inventor Taher El Gamal. The El-Gamal cryptographic algorithm is a public key system based on the Diffie-Hellman key agreement protocol [51]. El-Gamal is used by the Helios voting system explained in chapter 4.

El-Gamal Encryption

The El-Gamal encryption scheme works as follows:

Two known parameters are selected: a prime q (512bit) and a large prime p (1024 bits), where $p = 2q + 1$.
Then a generator g of the q-order subgroup of $\mathbb{Z}_p$ is chosen.
A secret key $x \in \mathbb{Z}_q$ is selected. (x is an integer between 1 and $p - 2$).

The corresponding public key y is computed by $y = g^x \mod p$.
The public key for El-Gamal encryption is the triplet $(p; g; y)$.

A random r relatively prime to $1 - p$ is picked at the time of encryption ($r \in \mathbb{Z}_q$) so that:

$$a \leftarrow g^r \mod p$$
$$b \leftarrow My^r \mod p$$

The ciphertext C then consists of the pair (a, b).

$$\text{Encryption(M ; r)} = (g^r, My^r) = \text{C}$$

Releasing $y = g^r \mod p$ does not reveal the secret key x.

El-Gamal Decryption

The decryption of the ciphertext C = $(a; b)$ in the El-Gamal scheme, to retrieve the plaintext M again, is then:

$$\text{M} \leftarrow b/a^x \mod p$$

$$\text{Decryption(C)} = My^r/(g^r)^x = My^r/(g^x)^r = My^r/y^r = \text{M}$$

Using the secret key x, we can effectively factor out the randomization from the second part of the ciphertext and get the message.

The correctness of the El Gamal encryption scheme is easy to verify. ($b/a^x \mod p = My^r \mod p / g^r x \mod p = My^r \mod p / y^r \mod p$ = M)

Using El-Gamal for Digital Signatures

To provide a digital signature a variation of the El-Gamal encryption and decryption scheme is used. A signature of a message M is a pair (a, b), generated using the same generator g and the parameter p.

The signature is obtained by the following:

A random integer k relatively prime to $p - 1$ is selected.
Then the pair of a and b is computed using the secret key x:

$$a \leftarrow g^k \mod p$$
$$b \leftarrow k^{-r}(M - xa) \mod (p - 1)$$

$$\text{Digital signature (M ; k)} = (g^k \mod p, k^{-r}(M - xa) \mod (p - 1)) = \text{S}$$

The digital signature S = (a, b) can be verified using the public key y. The El-Gamal verification is as follows:

$$y^a a^b \equiv g^M \mod p$$

El-Gamal Re-Encryption

El-Gamal cryptosystem offers simple re-encryption.

We have the ciphertext C = $(a; b)$
A random number $s \in \mathbb{Z}_q$ is selected. Then a ciphertext C' can be computed by C' = $(g^s a, y^s b)$.

Both C and C' decrypts to the same plaintext M.
C using r and C' using $r + s$.

El-Gamal in an Election Perspective

One of the good things about El-Gamal, in an election perspective, is the randomness.

When using a random number as input to the encryption process, there can be many different ciphertexts for the same plaintext. This is important in an election, where it can be for instance only 3 candidates. If an encryption

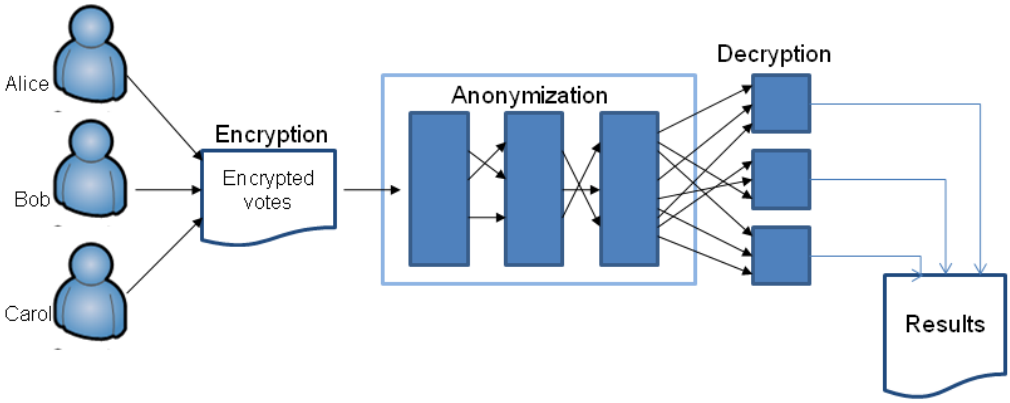

Figure 2.18: The anonymization component in a voting process

of the input word "Obama" always produced the same ciphertext, it would be easy to tell from a list of votes or a bulletin board who voted the same. An encryption of a vote option has to give a different ciphertext every time, and the El-Gamal encryption/decryption makes a plaintext be any string of bits up to a certain size.

The homomorphic property of El-Gamal is also a method to be used for tallying, described in section 2.5.6.

2.5.5 Mixing Networks

A mix net is a technique based on public key cryptography which includes mixing components between the corresponding parties in the cryptosystem, that process each sent data before it is delivered [5]. A mix net takes encrypted data as input, shuffles and reorganize the data and decrypting it. The purpose of a mix net is to hide the correspondences between the items in its input and those in its output. The idea of mix networks were invented by David Chaum in the early 1980s with his mix net based on nested encryption, but it also exists several types of mix net based on re-encryption, verifiable shuffles or probabilistic verification. Figure 2.18 shows how a mixnet can be included as a component in the voting process to ensure anonymization/privacy(verifiability).

Decryption Mix Nets

The method by David Chaum, based on nested encryptions as mentioned, includes a mixing computer that process all the data sent between the parties, and the concept is as follows [5].

Alice prepares a message M to be delivered to Bob, by sealing it with Bob's public key K_b (Recall the scheme of public key cryptography). Before sending the message, Alice also appends the adress B of Bob and then seals this result with the mix component's public key K_m. The input to the mix component will then be:

$$I = K_m(R_1, K_b(R_0, M), B)$$

Where R_1 and R_0 are random strings of bits.

The mix component then decrypts the input with its private key, throws away the the random string and outputs the following remainder [5]:

$$X = K_b(R_0, M), B$$

Then any mechanism in the system can forward the output X to Bob, which then can decrypt it using his own private key.

In addition, if Alice gets a signed receipt when submitting a message to a mix component, then she can also provide substantial proofs if the mixing component outputs the data X incorrectly or not at all [5].

If Alice has been wronged she can supply the receipt of:

$$Y = ({K_a}^{-1}(C, K_a(R_1, K_b(R_0, M), B)))$$

where ${K_a}^{-1}$ is Alice's private key and C is a large constant C (recall signing messages). Alice can also supply the missing output:

$$X = (K_b(R_0, M), B)$$

and the retained string of R_1, such that:

$$K_a(Y) = C, K_a(R_1, X)$$

Because a mix net will sign each output batch as a whole, if the item X is missing from a batch it can be substantiated by a copy of the signed batch [5].

It is a series of such mixes that creates a mixing net, with better robustness, that can provide the secrecy of the correspondence between the inputs and output, because any of the single mixes can provide the secrecy of the entire mix net.

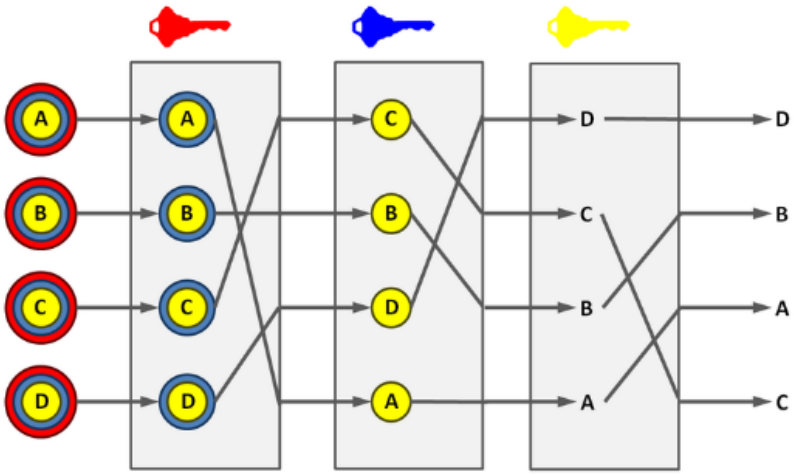

Figure 2.19: The idea of a decryption mix net (ref: wiki)

A data item from Alice to Bob is prepared for a series of N mixes the same way as for a single mixing components, and sealed for each succeeding mix by the encryption function, that encrypts the input data by a concatenation of N encryption operations to each data message. The initial encryption component knows the public keys of all the mixes $(K_1, .., K_k)$.

$$K_n(R_n, K_{n-i}(R_{n-1}.., K_2, (R_2, K_1(R_1, K_b(R_0, M)B))..))$$

And then each of the mixes in the mixing network peels of one of these encryptions by applying a corresponding encryption algorithm and then mixes all its decrypted inputs by applying a secret random permutation to them. A describing figure of a decryption mixing network is showed in figure 2.19. Every mix component has its own pair of private and public keys and mix and partially decrypt the data items. The resulting outputs of the last mixing component would be of the same form as the outputs from a single mixing component:

$$X = K_b(R_0, M), B$$

Similar as with one single mix component, also a series of mixes failing to properly process a message/item X can be proven. This only requires a receipt from the firs mix component in the mixing network because a mix N use the signed output of the N-1 mix to show the absence of an item X from its own input.

If this had to be used in a voting system a mix net would take encrypted ballots as the input and through its computations ensure that the ballots the

voting system records could not be traced back to the corresponding voters. If having a encryption receipt of the ballot cast, the voter could also detect if the mix net produces the output, because this output of the mix net would be displayed on a bulletin board.

Re-Encryption Mix Nets

A re-encryption mix net differs from a decryption mix net because the mixing phases are only mixing, no partially decryption. The decryption phase is added at the end for a final decryption [53], Since the resulting ciphertexts would not change in an only mixing scheme, it would be possible to recover the corresponding sender of a resulting cipher text. To mix in an unrecoverable way, the additional operation of re-encryption is needed.

In these mix nets, consisting of an encryption phase, several mix components scrambling and re-encrypting the set of received messages or data items,and a final decryption phase. The homomorphic encryption schemes described in section 2.5.4 allows such operations of a final decryption step of all encrypted data items. The encryption scheme typically used in re-encryption mix nets is the El-Gamal encryption scheme because of its nice re-encryption property. Recall details about the El-Gamal encryption scheme from section 2.5.4.

An El-Gamal based re-encryption mix net is explained with the following steps [53]:

1. An El-Gamal public-key (p, g, y) is generated
2. The initial encryption scheme of the system encrypts all the data items $B_1, ..., B_n$ using the El-Gamal encryption scheme with the public key (p, g, y) and posts all the generated ciphertexts $(C_{1,0}, ..., C_{n,0})$ on a bulletin board
3. The mix component i receives as input a set of cipertexts $(C_{1,i-1}, ..., C_{n,i-1})$. The mix component then re-encrypts each of the ciphertexts in the set and permutes (re-arranges) the resulting ciphertexts using a secretly chosen random permutation.
4. After processing through the mix components, the final decryption component receives a set of ciphertexts $(C_{1,k}, ..., C_{n,k})$. These ciphertexts are then decrypted and the original input data is obtained in a random order of how it was sent.

A voting system is said to be verifiable if all voters can verify that their vote

was counted, and for this mixnet to be sufficient for that property it also has to include proofs.

Provable Mix Nets

To be a provable mix net, each mix server have to prove it has done the correct operation. To prove this, each mix_i have to proove that there exists a permutation π such that $C_{j,i}$ is a re-encryption of $C_{\pi(i),i-1}$ for $j = 1, .., n$ [53]

Sako-Kilian Mixnet - an Example of a Provable Mix Net

The Sako-Kilian is the first provable mixnet based on El-Gamal re-encryption [54].

In the Sako-Kilian mixnet, all inputs are El-Gamal ciphertexts. The ciphertexts can all be re-organized and shuffled before jointly decrypted [41], and both the shuffling and decryption are connected to proofs of correctness. A Sako-Kilian Mixnet works like this:

A mix server takes N inputs (all El Gamal ciphertexts) and re-encrypts them using the re-encryption factors $\{s_i\}_{i\in[1,N]}$. Then the mix server permutes the ciphertexts according to a random permutation πN, so that $d_i = \text{Reenc}(C_{\pi(i)}, s_i)$.

To prove that it the inputs are mixed correctly the mix server also produces an additional "shadow mix". To verify the mixing computations the verifier can challenge the mix server to reveal the permutation and re-encryption factors of the "shadow mix" or he can ask it to reveal the difference between the two mixes. The difference between the two mixes is "the shuffle that would transform the shadow output into the primary mix outputs" [41]. This scheme is called the "Shadow-mix shuffle proof" and is sketched in figure 2.20. If the verifier challenges the mix server with a bit 0 it reveals the permutation and re-encryption factors of the "shadow mix" or if challenged with a bit 1 it reveals the difference of the two mixes as explained.

If the mix server is an honest mix server, it can answer to both these challenges, and verify to the auditor that the process of of the mix was done properly. "A corrupt or cheating mix can answer at most one of these challenges convincingly, and be caught with at least 50% probability." [41]

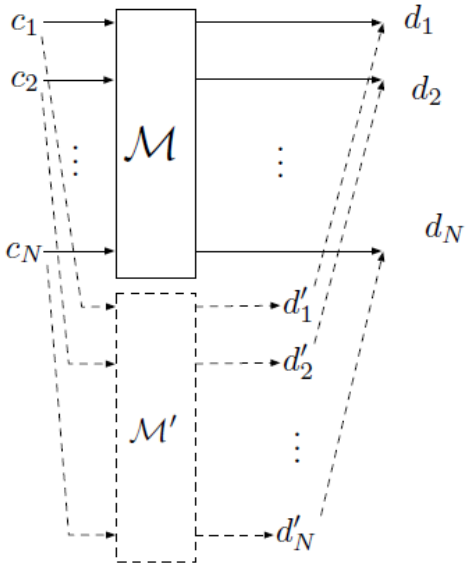

Figure 2.20: The "Shadow-mix shuffle proof" [41]

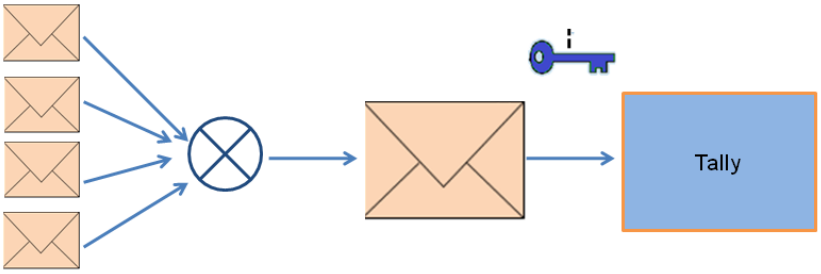

Figure 2.21: The idea of homomorphic tallying

2.5.6 Homomorphic Tallying

The idea of homomorphic tallying is that individual encrypted votes can be added together [55]. The result of all the votes added together can then be decrypted and tallied. The reason for using homomorphic tallying is to add all the votes from all voters together, before decryption, so know one can see the content of a specific voter's ballot. The idea is shown in figure 2.21

An election can be summed up in this form:

$$f(v_1, ..., v_n) = \sum_{i=1}^{n} v_i$$

This addition of votes can be defined in several ways. If for instance having

only yes/no votes this corresponds to integer additions of votes $v_i \in \{0, 1\}$. If having more complex ballots with multiple candidates, which is generally the case in government elections, the votes $v_i \in \{0, 1\}^m$ are binary m-tuples

The property of a homomorphic encryption system, like for instance ElGamal, makes is possible to take the encryption of one vote v1 and the encryption of another vote v2, multiply them together and get the encryption of the product. (The randomization factor of the encryptions gets added up in the exponent and the two plaintexts/messages get multiplied together.) In a cryptographic voting system, the voters identity is used in the encryption process of a vote so the voter is able to verify his his recorded vote and that is correctly recorded. By using the method of homomorphic tallying the vote's origin is not revealed like it could have been if the encrypted plaintexts were encrypted separately. The point in voting is of course not to multiply votes together, but use the process similar to add votes to get the total result.

2.5.7 The "Double Envelope Scheme"

The Double envelope scheme is a scheme of protecting the senders privacy, and is actually used in the Estonian voting system mentioned earlier and the idea is also used in the "E-Vote 2011" project.

A double envelope system can be created using asymmetric encryption and at least two key pairs. [29] The one key pair is used to protect the actual vote (the inner voting envelope) and the other to verify the vote is really coming from the given voter (the outer envelope). After the voter has issued his choices, the ballot is encrypted by the election's public key. This encryption seals the inner envelope and the vote can only be read (envelope opened) after decryption with the paired secret key.

The key pair is generated by a security module just before the voting process begins. The public key of the election is distributed to the voters through the software used when voting, while the secret key is kept in the security module. The secret key is only made available after the election polls closes, and it can also require a procedure where a threshold of people with contradictory interests, for instance from different parties or the election board, have to be gathered to open the security module using a number of individual keys (shares of the secret key).

Before the encrypted vote is transferred from the voting client, it is also put in a sealed outer envelope. This envelope is given a digital signature of the voter by adding the voter's credentials and encrypting it using the voter's

private key. To open this outer envelope it has to be decrypted using the voter's public key which the e-voting system can obtain from the electoral roll or from a national PKI system. By using the digital signature the e-voting system can verify the vote is really submitted by the given voter.

2.6 User Authentication

"User authentication is the process of verifying the identity of a user" [44], to verify whether someone (or something) is in fact what it claims to be. Authentication technology provides the basis for access control in computer systems [56]. A user has to authenticate to a system to convince the system of his identiy before the system will act on the users behalf or grant the user access to system resources.

In private and public computer networks (including the Internet), authentication is commonly done through the use of secret login passwords, but sometimes it also requires stronger security with accompanying mechanisms. This can be the use of smart card, generated PIN codes, certificates, etc. Sometimes it requires that the user's computer also verifies its identity together with the user.

[44] divides user authentication methods into three categories:

1. What you know
2. What you have
3. What you are

What you know is the same as providing a password or a PIN code. What you have is regarding the use of a token you possess, like the extra mean of a card reader or a code calculator. The identity is then based on possession of some object, often also combined with a password. What you are, means identity verification based on physical characteristics or involuntary response patterns known as biometrics. This can be fingerprints, speech, signature, face profile and more.

In a voting system, as with any computer system to handle personal information, authentication is an extremely important mean to ensure the correct function of the system and gaining the user's trust. The obvious mean is that the user has to authenticate to cast a vote in the system, but is is also

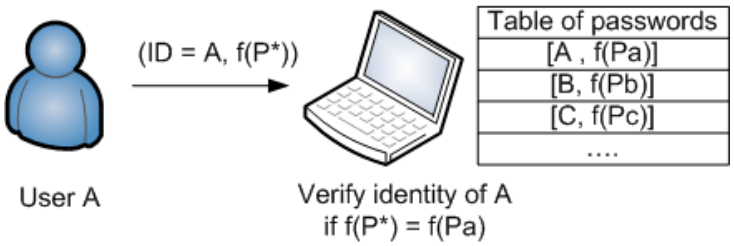

Figure 2.22: The idea of a user authentication by password[44]

desirable that the vote recorder authenticates to the user so the voter would know it is not a false (phishing site) system claiming to be the voting system.

2.6.1 Password

The web site or application asks the user for a login name and password, and then the identiy of the user is verified by checking that he can provide the correct password.

The passwords are usually not stored in the clear, to avoid a compromise of all the users of a system in the case of intruders. Often a one-way function is used (output makes it hard to find input value) with a fixed sized input or a hash function to create a value of a variable sized input [44].

A secret password scheme relies on the user to carefully select a strong password to reduce risk of exhausted search. Figure 2.22 shows the idea of password authentication.

Passwords transferred over an insecure network (wireless network, Internet) are vulnerable to eavesdropping. And an extra security could be to use one-time passwords [44]. Then the password is only used once for a single log in or authentication, and for next time a new password is used (with no relationship to the previous).

This can for instance be done by the user keeping a printed list of passwords corresponding to a list of password in the system's database. This is used by some Banks, were you receive for instance a plastic card with 50 printed codes to use for authentication.

2.6.2 Smart Cards

Smart card (in authentication context)is a type of token-based authentication [56], where the token is the plastic card (size of a credit card) containing a microprocessor, that the user possess. The smart card chip can contain specific user authentication information coded that can be recognized by the system, and work like an electronic ID. This can be used to gain access to systems as well as for digital signing purposes.

To authenticate using a smart card, the user at least needs his card and a card reader that can interface with a computer using a USB port. The smart card chip can store multiple identification factors (for instance password and fingerprint), and when put into the card reader it can implement multiple factors of authentication providing two-factor authentication (for instance the card information and the users password or pin code).

In a smart card authentication scheme, the card reader and the user's computer can communicate with the microprocessor. The microprocessor controls the computer's access to the data on the smart card. The data on the smart card can be encrypted, and additionally a password can be employed to prevent unauthorized reading of the data. The interaction between the user's computer include steps to determine if the card is authorized to be used in the system, and checks if the user can be identified and authenticated [57]. A control that the user presents the appropriate credentials to conduct a transaction has to be performed. The card can also demand the same process from the computer before proceeding with a transaction. This features can ensure that for instance a banking application has been authenticated as having the appropriate access rights before accessing financial data or functions on the card [57].

The newest smart cards support most authentication technologies, including storing password files, seed files for one-time passwords, biometric image templates, PKI certificates and supports the generating of asymmetric key pairs [58]. When using a smart card in combination with one or more of these authentication technologies, it can provide strong multi-factor authentication.

Since the user information is processed on the smart card, it never has to leave the card or be transmitted to another machine. This eliminate the problem with password authentication where the password has to be transferred and can be eavesdropped by attackers. With smart card technology all authentication factors can be included in a single smart card, improving

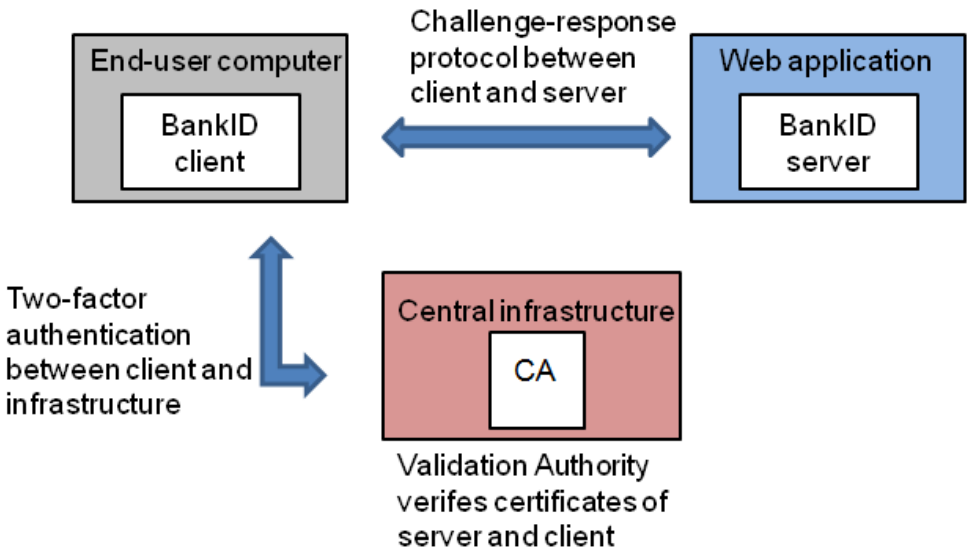

Figure 2.23: The BankID authentication procedure [60]

the security and privacy of the authentication process and provide stronger access security [58].

2.6.3 BankID

BankID is a mean to provide digital signatures and identification of a user on the Internet. It works as an electronic ID when the user logs in to a resource with BankID, and can work as a digital signature when used to sign a deal or perform for instance a money transfer [59]. BankID can be seen as a substitute for a PKI. The mechanism is widely deployed by Norwegian banks and approximately 2.5 million people users have a BankID [59].

To authenticate with BankID the user have to provide his social security number, together with a code retrieved from a security card and a personal password chosen by the user. The security card can be a plastic card with pre-printed codes, an electronic code calculator generating security codes, or a card used together with a card reader.

The technology is based on PKI, using key pairs (one private and one public) for security functionalities together with a digital certificate for electronic ID. The authentication scheme is shown in figure 2.23.

The user's keys are stored in the central infrastructure, and the certificate authority (CA) creates signatures on behalf of the user when this is requested.

To make the CA create signatures, the user must authenticate himself. This is done through a two-factor authentication where the user provides a password and a one time pin (from the security card or generated by a code calculator).

2.6.4 MinID

"MinID" is an electronic ID provided for Norwegian citizens to gain access to several public services like revenue services, university admission services, etc. using the same authentication information [61]. MinID is available through a new common platform for electronic ID in the public sector called "ID Porten"

The authentication scheme is based on mutual trust and federate identity with "MinID" as the Identity Provider The user accounts of a user in different systems are connected together making it possible for the user to be uniquely identified as the owner of the accounts across these systems. When a user is logged in with one service provider, he will also have access to services from other service providers without having to authenticate again (Single sign on and single sign out) [61].

To communicate safety and identity between the service owners/providers and MinID, a XML-based frame work is used. This framework is called SAML v2.0 (Security Assertation Markup language version 2)(REF!). SAML is a protocol for communication identity from one domain to another. The idea of the federation model is showed in figure 2.24.

SAML is a neutral platform with no need for maintenance or synchronization by the service providers, but does neither provide any means for authentication and access control. This has to be configured above the platform.

MinID is using two central SAML 2.0. profiles; Web browser single sign-on profile and single log-out profile. These two profiles are described in [62] , and will be described further here.

A user registers his "MinID" using parameters including; social security number, a chosen password, PIN codes from a letter received in the mail, cellphone number and optional an email address. To authenticate using MinID the user provides his social security number and the chosen password. When submitting this the user receives a "one-time" password by SMS, that also has to be typed in as an extra security feature to prove his identity.

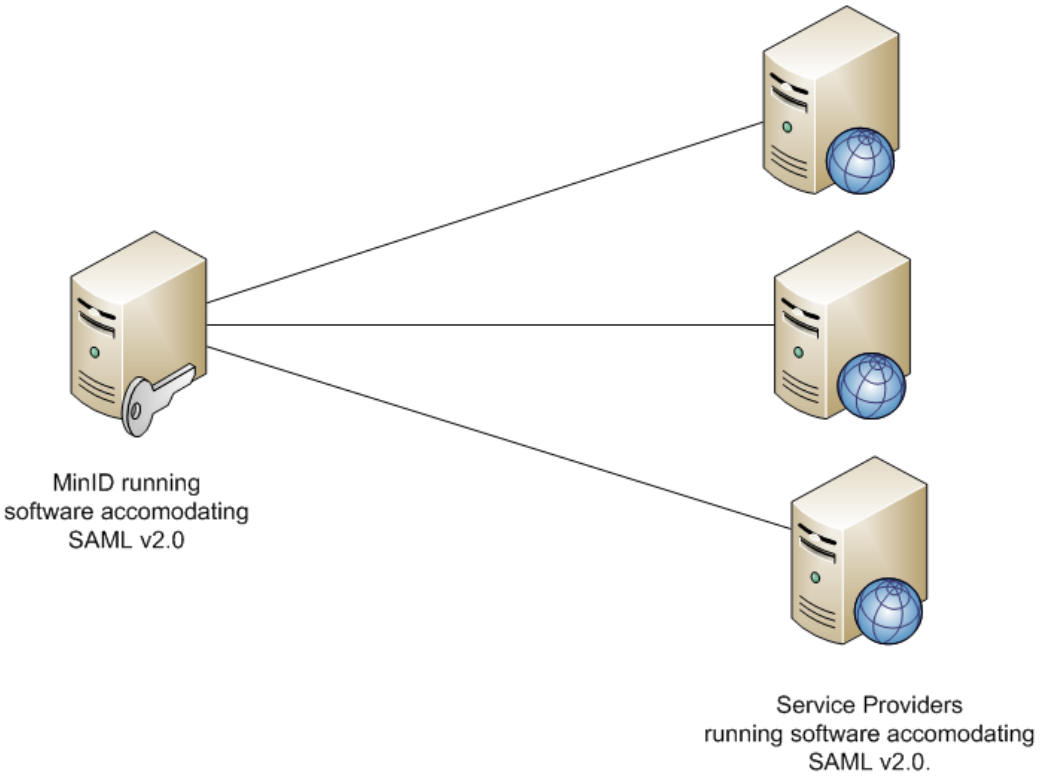

Figure 2.24: Federate identity with "MinID" as identity provider (REF: Implementation guide for federation with MiniD)

The new eID card DIFI is working on developing is a card containing a chip for biometrics and a public issued eID.

2.6.5 Biometric Authentication

Biometrics are physical characteristics or involuntary response patterns [44]. Authentication by biometrics verifies the identity based on some of these characteristics, including signature, fingerprint, speech, face profile, etc. Biometrics provide a very high level of security because the authentication is directly related to a unique physical characteristic of the user which is more difficult to imitate.

The unique pattern that identifies a user is formed during an enrollment process, producing a template for that user [56]. If a user wants to authenticate to a system , a physical measurement is done to obtain a current biometric pattern for the user. This pattern (for instance a fingerprint) is then compared against the enrollment template to verify the user's identity. (REF: NIST)

Smart cards can provide me mechanisms to securely store biometric templates and perform biometric matching functions [58]. These features can be used to improve privacy in systems that utilize biometrics. For example, storing fingerprint templates on a smart card rather than in a central database can be an effective way of increasing privacy in a single sign-on system that uses fingerprint biometrics as the single sign-on credential.

The "easiest" pattern to use for authentication is the fingerprint, and this is used for check-in at some airports. Then the only a computer connected to the airlines system, a biometrics software and a fingerprint scanner is needed to authenticate.

In Bulgaria, they have introduced biometric authentication to voting schemes, and thereby calling it biometric voting. [63]. The mechanism was introduced to make it impossible to vote with some other persons voting card. The electronic voting machines at the polling stations authenticates the user with biometrics using a connected "finger reader".

Chapter 3

E-Vote 2011

Norway is also engaged in the field of Internet voting and is now working on developing an e-voting system for the 2011 municipal elections [64]. The Government will trial Internet voting as a supplement to existing paper based in some selected municipals in a single county (approx 200.000 possible voters), and if the result is as expected it will be developed for a full scale integration for the general voting public in all elections. In the pilot only eligible voters in the municipals of Bodø, Bremanger, Hammerfrest, Mandal, Radøy, Re, Sandnes, Tynset, Vefsn, Ålesund and a part of Oslo, are able to vote electronic in 2011. One of the goals of the pilot is to establish routines for e-voting that will not only ensure a correct result but also build trust among the public. Beside satisfying laws and regulations, the system has to satisfy several important security requirements. In this chapter I will describe the planned Norwegian e-voting solution, security features and threats.

3.1 Motivation and Goals

The purpose of developing an e-voting system in Norway is to create a secure electronic election system for both the general and municipal elections. The system is going to simplify voting and give better accessibility for all groups of voters, which includes improved accessibility for disabled voters. In traditional voting scenarios the municipalities prepare the polling stations for disabled voters, but for some disabled voters it is even difficult to get to the polling station or vote without help. By introducing voting via Internet, functionally disabled people would have a better opportunity of voting without help [65].

Another goal is to make it easier for absentee Norwegian citizens to vote. Voting over the Internet for people that resides, would be a better option when it is hard or not possible to get to a polling station.

The fact that we are in an Internet era, and voting should follow the evolution of the computerized society, is also an important motivation. The youth (and others) has expectations to electronic solutions and when being able to do "everything" over the Internet, voting should also be possible. The participation in some groups might increase when voting over the Internet is a possibility [65].

Reducing the costs of carrying out an election process is also a factor of motivation for developing such a system. The development cost will probably be relatively high, but costs of the the election process itself can be reduced by mean of less polling stations, less manual preparations and less election officials. In an environmental perspective it can reduce environmental costs with less paper printing and transportation.

3.2 Description of system

The Norwegian e-voting system includes all the four phases of an election process; necessary preparations before the poll, the actual voting process, counting of votes during and after closed poll and the settlement. The system also have mechanisms to audit every phase of the entire election, to ensure everything works as intended. The overview of the four phases are showed in figure 3.1, but preparations and settlement will not be further covered in this thesis.

In this chapter the e-voting system is described, including authentication mechanisms, the voting protocol and tallying. Administrative steps (preparation phase and settlement phase) are not further considered.

3.2.1 Architecture

The Norwegian voting system is a system including both the use of paper ballots and electronic ballots. A voter can still cast a vote at a polling station, using a traditional paper ballot, but this vote will also be scanned and recorded electronic for electronic counting. The system offers the voter three options on how to cast his vote; with a paper ballot at the polling station, through a computer set up at the polling station, or from any remote personal

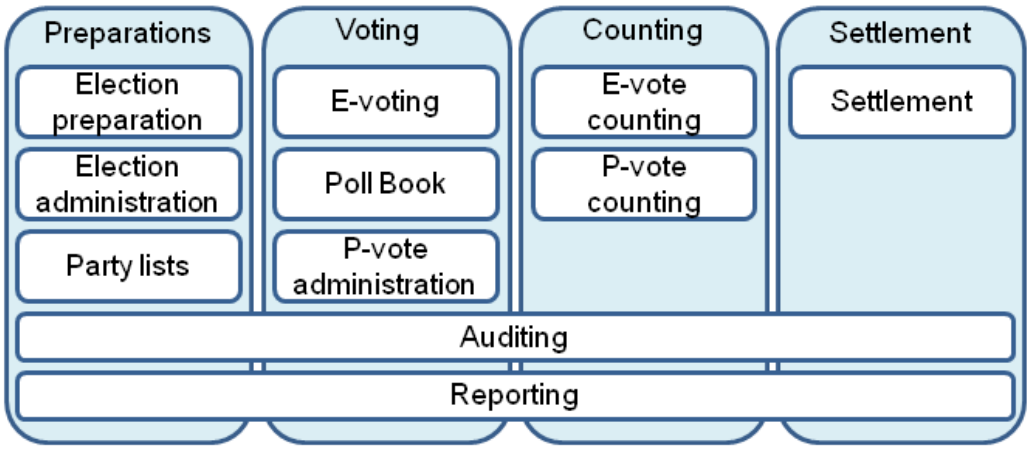

Figure 3.1: The lifecycle/phases of an election!! IF USE..FIX! REDRAW

computer connected to the Internet. Figure 3.2 from Ergo Group shows the cooperation of the e-voting system, including voters casting ballots from computers from both controlled environments (polling station) and remote locations, and e-counting. Paper ballots are cast at polling stations, and recorded using optical scan systems.

A very high level security scheme is showed in figure 3.3. The components of the solution consists of:

- Voting Client, running at voter's computer
- Vote Collection Server
- Receipt Generator
- Decryption component
- Counting component

The the voting protocol is further described in 3.2.4.

3.2.2 Voting process

The process of casting a vote in the Norwegian e-voting system, starts with the voter receiving a voting card (showed in figure 3.4) by mail some time ahead of the election period. The voting card contains necessary information for the voter, as well as pre-computed random generated receipt codes corresponding to possible voting options on the election ballots. These receipt

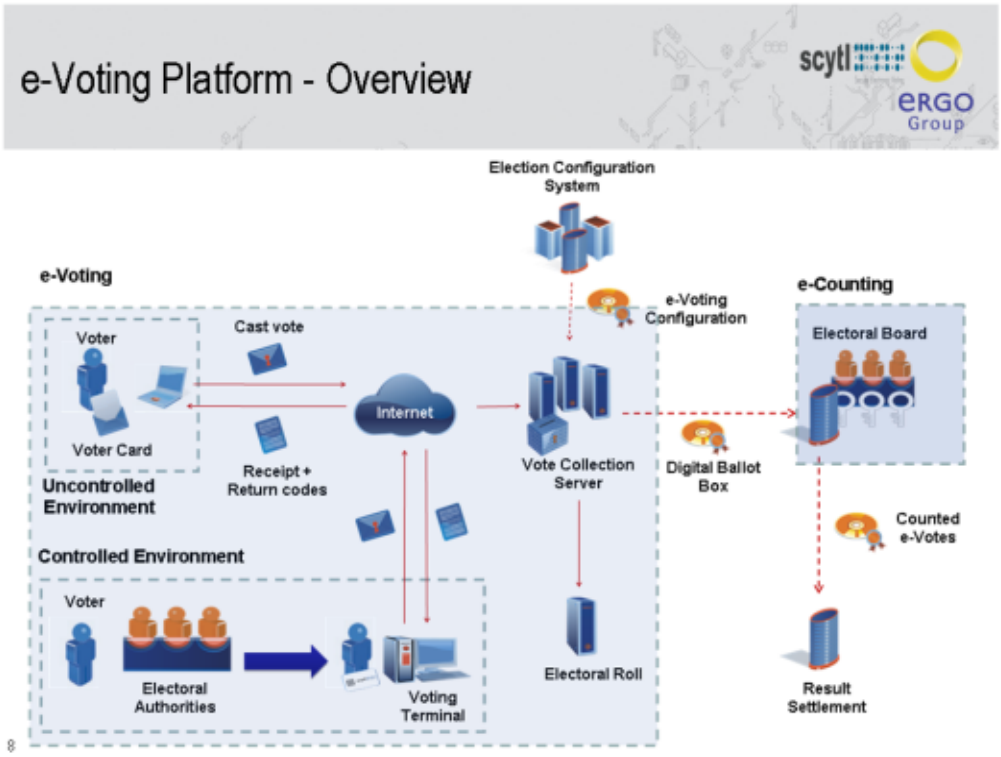

Figure 3.2: Overview of the election system by Ergo Group [12]

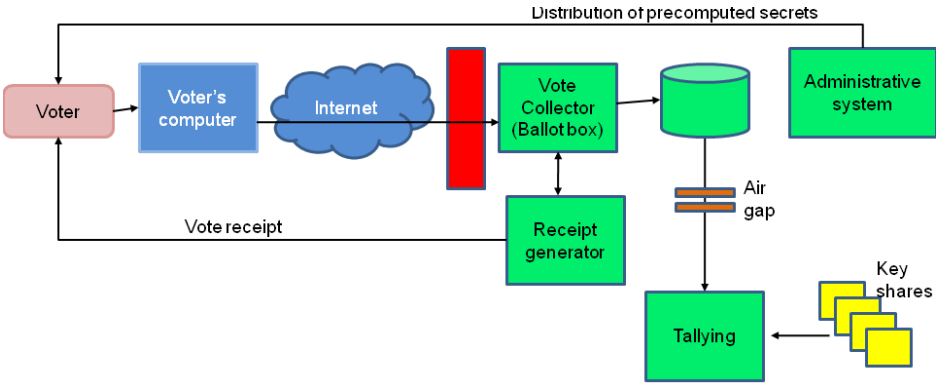

Figure 3.3: Overview of security solution [12]

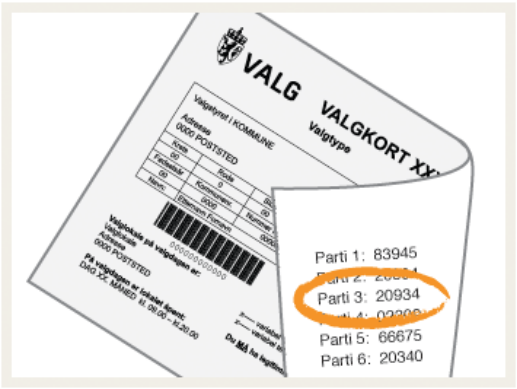

Figure 3.4: The voting card with receipt codes sent to the voters by mail [12].

codes are individual and specifically generated to the voter, but are random and can not reveal what the voter voted [65]. Receipt codes are further explained in the cryptographic voting protocol (SECTION ???).

When the period of advanced voting online opens, an eligible voter can use his browser to visit the election web site to cast a vote. The process is based on a standard Scytl voting protocol [66], but in addition also includes receipt codes:

1. The voter provides his eID credentials and the voting application (Voting Client) authenticates the voter to the voting server.
2. The Voting Client displays the list of parties and candidates in the election the voter has permission to participate. The voter mark options (or leaving options blank) to issue his choices for candidates and parties.
3. When issued his choice of candidate/party the voter clicks to submit his vote. The ballot is then encrypted using homomorphic encryption, zero knowledge proofs are attached and everything is digitally signed with the voters eID (as a double envelope scheme). This is done to protect the voter's privacy and the ballot from being tampered with. The encrypted ballot is sent to the vote collection server, and since using homomorphic encryption the server knows how to map codes and parties/candidates.
4. When the vote is received by the system, the digital signature and the cryptographic zero knowledge proofs are validated. If accepted, the system updates the electoral roll and issues a voting receipt of the vote.

The voter receives an SMS containing the receipt code corresponding to the vote he issues, and can verify this code is correct against the codes on his voting card. By this mean the voter can for instance detect if a malicious program on his computer has changed the ballot. The receipt code can also be used for the voter to verify his vote is present in the latter e-counting process.

3.2.3 Ensuring Integrity

The Voting Client, which is a digitally signed applet running on the voter's machine, executes the client side of the secure electronic voting protocol before the vote is submitted [67].(The Scytl/Norwegian cryptographic voting protocol is described in section 3.2.4). The vote is encrypted with the election public key, with corresponding private key does not exist before the polls close. The voter digitally signs the encrypted vote, and it is sent to the Vote Collection server. An overall simplified figure of the high level security architecture is showed in figure 3.3.

When the digitally signed encrypted e-vote is received at the Vote Collection Server, the correct construction of the encrypted e-vote has to be verified (through several validation processes) [67]. The Vote Collection Server verify the digital signature to ensure the validity, integrity and authenticity of the encrypted e-Vote. The Vote Collection Server executes the server side of the secure electronic voting protocol. This part for instance validate the voter authentication and the opening token, as well as ensures a secure storage and processing of the e-vote received. The vote collection server also verifies the cryptographic zero knowledge proofs to ensure the encrypted e-vote contains valid options and that it has not been duplicated, for instance using an old session token.

When the validation has succeeded, the e-vote is stored and the voter is marked off in the Electoral Roll. The system then sends a receipt message back to the voter that his vote was cast and successfully recorded. This feedback includes a hash of the encrypted e-vote and is digitally signed by the Vote Collection Server (further explained in 3.2.4). The digital signature proofs the integrity and authenticity of the receipt code, while the hash of the encrypted e-vote is used to guarantee voter privacy and vote secrecy, because it cannot be used in any way to reveal any vote options issued by the voter [67].

After the vote casting process is done, a hash of all encrypted e-votes is

generated. This hash is sent to the cleansing process (described in 3.2.5). The system also publishes a list of all generated hash values, to provide voters with the possibility of verifying their e-votes were recorded correctly and successfully reached the electoral authorities.

In the next section the voting protocol and receipts are considered in greater detail.

3.2.4 The voting protocol

An Internet voting system depends on a secure and robust cryptographic protocol. Security of the protocol is the necessary factor to get people to trust the voting solution. In this section I have described the simplified Internet voting protocol to be implemented in the "E-Vote" solution, based on a published paper of the solution [1]. The cryptographic protocol is invisible to the voter, hidden in the bottom, and the user interface is designed as best suitable on the top [1].

To cast a vote, as already described, the voter uses his computer to submit a vote to the election infrastructure and then receives a feedback of generated receipt codes to verify his vote. It is the voter's computer that encrypts the ballot and submits it to a ballot box in the election infrastructure. When the ballot box closes, the submitted ciphertexts are decrypted based on a reencryption mix net. The cryptoprotocol has to have functions to allow the voter to submit repeatedly ballots consisting of a sequence of options (chosen from a set of options $\mathcal{O}$) to the infrastructure. The cryptosystem used is El-Gamal, and the receipt codes are generated using exponentiation in the ciphertext(since El-Gamal is homomorphic). [1]

Figure 3.5 from [1] shows an overview over the connection between the different actors in the protocol. The voter is denoted $\mathcal{V}$ and the voter's computer is denoted $\mathcal{P}$. Inside the dotted line is the system's infrastructure actors consisting of the ballot box $\mathcal{B}$, the decryption service $\mathcal{D}$ and the receipt generator $\mathcal{R}$. (Which communicate together over secure authenticated channels.)

To create a ballot the voter chooses a sequence of options $(v_1, \ldots, v_k)$ from a set of options $\mathcal{O} = 1, 2, \ldots$. In the computer $\mathcal{P}$ the ballot is padded with zeroes up to a fixed length k_{max} and encrypted with the election encryption key, before it is submitted to the ballot box $\mathcal{B}$.

The ballot options are encoded as Diffie-Hellman group elements [1].

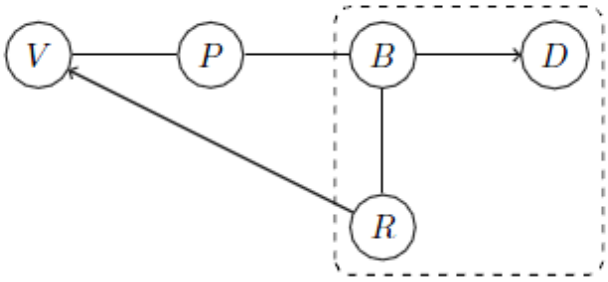

Figure 3.5: The actors in the cryptographic voting protocol [1].

When the ballot box receives the ballot, it together with the receipt generator compute a receipt code. The receipt code is sent directly to the voter, as shown in figure 3.5, and this has to be done through an independent channel (SMS). After receiving this feedback the voter has a correspondence between his choice of options and receipt code, and can perform a control. If the receipt codes matches the selected options, the voter can accept his vote is recorded as intended.

Generation of keys and precomputed receipt codes

The voting protocol uses a finite cyclic group $\mathcal{G}$ of prime order q generated by g and a pseudo-random function family $\mathcal{F}$ from $\mathcal{G}$ to $\mathcal{C}$ [1]. Additionally, a function f: $\mathcal{O} \to \mathcal{G}$ is chosen.

Before an election, three secret parameters are generated: a_1, a_2 and a_3
Such that: $a_1 + a_2 \equiv a_3 \pmod{q}$

These parameters are given to different actors in figure 3.5. The decryption service gets a_1, the ballot box gets a_2 and the receipt code generator gets a_3.

In addition three public parameters for the election is computed by:

$$y_1 = g^{a_1}$$
$$y_2 = g^{a_2}$$
$$y_3 = g^{a_3}$$

To generate receipt codes for each voter, a parameter s is sampled from $\{0, 1, \ldots, q-1\}$ and a parameter d is sampled from $\mathcal{F}$ for every voter. Based on the composition of f, the exponentiation $x \to x^s$ and the parameter d, a function $r : \mathcal{O} \to \mathcal{C}$ is given by: $r(v) = d((f(v))^s)$.

Before the election the set $\{(v, r(v)) \mid \in \mathcal{O}\}$ is computed. This set of receipt codes are sent to the voter in advance of the election.

The vote submission process

The voter's ballot, as mentioned, consists of a sequence of options $(v_1, \ldots, v_k)$. The options are encoded as group elements, with a random injection $\mathcal{O} \to \mathcal{G}$ chosen as encoding function f

When the voter wants to submit this ballot, the protocol goes through these following steps [1]:

1. The voter sends $(v_1, \ldots, v_k)$ to his computer $\mathcal{P}$. The computer pads the ballot by setting $v_i = 0$ where $i = k + 1, \ldots, k_{max}$.

2. For $1 \leq i \leq k_{max}$, the computer samples the parameter $t_{1,i}$ from $\{0, 1, \ldots, q-1\}$. It also computes $(x_i, w_i) = (g^{t_{1,i}}, {y_1}^{t_{1,i}} f(v_i))$. Then it sends $((x_i, w_i), \ldots, (x_{k_{max}}, w_{k_{max}})$ to the ballot box $\mathcal{B}$.

3. With these parameters the ballot box computes $\check{x}_i = {x_i}^s$ and $\check{w}_i = {w_i}^s {\check{x}_i}^{a_2}$. Then the ballot box sends $((\check{x}_i, \check{w}_i), \ldots, (\check{x}_{k_{max}}, \check{w}_{k_{max}})$ together with the voter's name (voterID) to the receipt generator R, for generation of receipt codes.

4. The receipt generator $\mathcal{R}$ can then compute $\check{r}_i = d(\check{w}_i {\check{x}_i}^{-a_3})$ and send the values $(\check{r}_1, \ldots, \check{r}_k)$ to the voter.

5. When receiving the values of $\check{r}_i$ the voter can verify that every pair of $(v_i, \check{r}_i)$ is in the set of receipt codes he has received in advance of the election.

If the verification is fulfilled, the ballot is considered cast. When the election has come to an end and the ballot box closes, all the voter's last submitted and encrypted ballot is sent to the decryption service $\mathcal{D}$ in a random order.

The entire process is fulfilled when the submitted ballots are correctly decrypted and the receipt coded sent to the voter matches the expected values. The receipt code received by voter $(v, \check{r})$ always has to be in the computed set of receipt codes.

Security of the protocol

In the process of development, the security of the protocol has been analyzed [1]. The protocol has the following alleged properties:

- The voter will most likely discover if his computer is corrupt and modifies his ballot.
- No non-trivial information about the ballots can be acquired by a honest but curious infrastructure player.

To prove this, the analysis consider 3 corruption models:

1. The voter is corrupt (by this also his computer).
2. The voter's computer is corrupt.
3. One of the infrastructure players is honest, but curious.

If first considering the scenario of a voter being corrupted, the analysis of [1] claims he will not affect the election system. By the assumption of authenticated channels, the ballot box receiving the votes can ensure that at most one ballot is counted per voter, making ballot stuffing "impossible". If a corrupt voter are trying to submit malformed ciphertexts this will at most affect the system turnout by invalidate the voter's ballot (which is expressly permitted).

The second scenario is about the case of the voter's computer being corrupt. Could this affect the election turnout? And would the voter discover the computer is modifying his ballots?

[1] looks at an example if the computer submits the ballot $(v'_1, \ldots, v'_{k'})$ instead of the ballot $(v_1, \ldots, v_k)$ the voter wants to submit. Recall from 3.2.4 that the exponentiation map is a permutation on $\mathcal{G}$ and that f is an injection. d is sampled from the pesudo-random function family F and will look like a random function. Since f is composed with a permutation composed with the random looking function d, f will look like a random function from $\mathcal{O}'$ to C. The voter will accept the manipulation if and only if $(v'_1, \ldots, v'_{k'}) \sim (v_1, \ldots, v_k)$. But as long as the set $\mathcal{C}$ is sufficiently large, the probability of this similarity is small.

The third corruption model is considering the system's infrastructure players not being corrupt, but curious. Three of the infrastructure actors are analyzed (the ballot box, the receipt generator and the decryption service) , simulating the input they would normally see. The security is proven by using

the Decision Diffie-Hellman problem (a computational hardness assumption involving discrete logarithms in cyclic groups).

An honest, but curious *ballot box* B^*, can after an election look over the ciphertexts received and output some information about the ballots that has been submitted. A tuple (g, y_1, u_1, u_2) of elements from G is given, and B* is employed as follows [1]:

1. a_2 is generated and sent to B^*
2. $y_3 = y_1 g^{a_2}$ is computed
3. Now, instead of encrypting the encoded option $f(v_1)$ (as described in point 2 in 3.2.4), the encryption $(x_i, w_i) = (g^{t_{1,i}} u_1^{t'_i}, y_1^{t_{1,i}} u_2^{t'_i} f(v_i))$ is computed, for some random t'_i

If the tuple (g, y_1, u_1, u_2) given in the beginning is a Diffie-Hellman tuple, the calculations of point 3 above will simulate the ballot box input. If not Diffie-Hellman the ballot box will contain no information about the ballots. So, If the curious ballot box B^* can extract information about the ballots, it is a distinguisher for the Decision Diffie-Hellmann problem [1].

To consider a honest, but curious *receipt generator* R^* the analysis first includes an assumptions [1]; the familiy of functions from $\mathcal{O}$ to $\mathcal{G}$ given by $v \to f(v)^s$ is a pseudo-random function family. The analysis is using the fact that a pseudo-random function family is functions sampled uniformly at random from the family are indistinguishable from functions sampled uniformly at random from the set of all possible functions from $\mathcal{O}$ to $\mathcal{G}$). This assumption can prove privacy against the receipt generator.

A function f and a function $p : \mathcal{O} \to \mathcal{G}$ are given. The honest, but curious R^* that after the election outputs some non-trivial information about submitted ballots, are tested as follows [1]:

1. For the voters $V_1, V_2, ..., V_{j-1}$ he chooses a random function $p_l : \mathcal{O} \to \mathcal{G}$ where $1 \leq l < j$.
2. Keys for $V_{j+1}, ..., V_N$ are generated as normal (recall section 3.2.4) using the functions $p_l : v \to f(v)^{s_l}$ were $j < l \leq N$.
3. For V_j the given function p is used.
4. For every ballot receieved $(v_1, ..., v_k)$ from a voter V_l with function p_l, $(\check{x}_i, \check{w}_i)$ is computed as $(g^{t'_i}, y_3^{t'_i} p_l(v_i))$.

"If the function p comes from the family and $j = 1$ this, described, will simulate the receipt generator input perfectly. But if the function p is a random function and $j = k_{max}$ the receipt generator input will contain no non-trivial information about the submitted ballots. So if $R*$ can extract some non-trivial information about the ballots, they have a distinguisher for the function family. [1]"

The decryption service sees the the encrypted ballots in random order and have no possibility of seeing which ballot originated from which voter. An honest, but curious decryption service can therefore not extract any information about which ballot corresponds to which voter.

3.2.5 Counting votes

The system supports both the recording of paper ballots and electronic ballots, before jointly counting them. An overview of the counting process, where votes are validated, is showed in figure 3.6. The counting of electronic ballots is divided into three phases [67]:

- The cleansing phase
- The mixing phase
- The counting phase

The counting of paper ballots first involves a scanning process with manual verification, and then the vote is recorded electronic to be counted the same way as an originally electronic ballot.

Recording and verifying electronic ballots

The electronic ballot box and other data collected by the Vote collection server is transferred securely (off line, for instance external hard drives) to a new system in a secure location for counting. This system has (never?) been connected to the Internet. The system components in this counting systems are only connected to each other, and can not be reached from the outside.

The electronic ballot box, which is digitally signed to ensure its authentication and integrity, contains the digitally signed encrypted votes and cryptographic proofs generated by the voters.

In the cleansing phase the digitally signed ballot box is received from the Vote Collection Server, and the content is validated through some pre-established

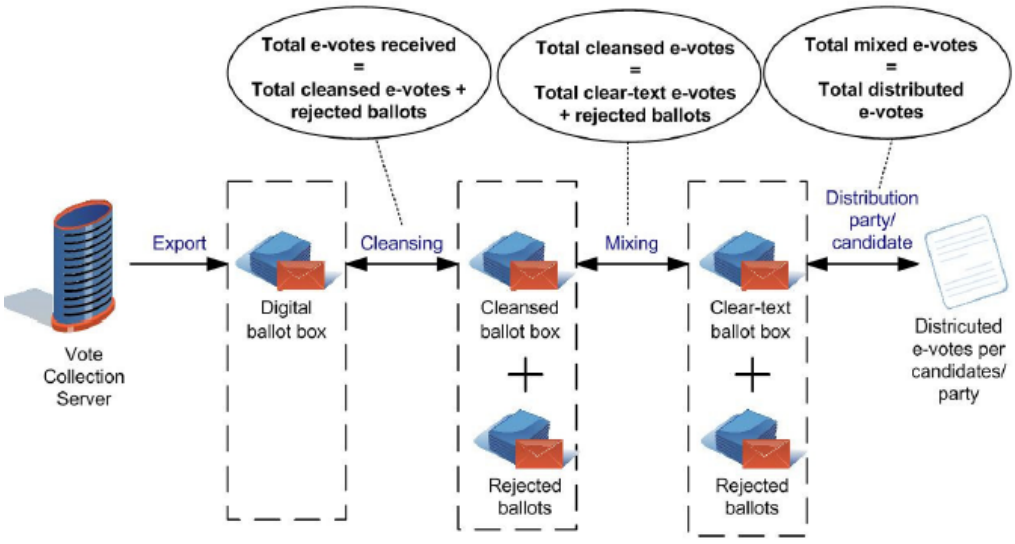

Figure 3.6: Overview of the counting process with validation [67]

rules of the election. The digital certificates are validated based on the X509 standard (REF!). The cleansing process is executed in an air-gapped environment and it can import a digitally signed copy of the Electoral Roll. The cleansing phase produces a digitally signed, cleansed ballot box (with only valid votes) for each municipality/county as defined in the election configuration. Also the digital receipts from all the electronic votes in the digital ballot box are kept to allow voters to verify that their votes has reached the counting process. After the cleansing phase the digital ballot box are transmitted to the mixing service following some air-gapped approach.

In the mixing phase, the digital signature of the ballot box is first verified to check the authenticity and integrity. To check that all of the votes still belong to valid voters, that no rogue votes has been added or modified, the digital signatures of the votes are also verified.

After these verifications, the digital signatures from the votes are removed so the votes now are anonymous. The votes are then processed by a re-encryption universal verifiable Mix-net which breaks any correlation between the votes and their voting order. The result, or output, of this mixing process is that all the votes are re-encrypted and shuffled, and in addition a set of cryptographic proofs (zero knowledge proofs? EXPLAIN!) of correct mixing behavior is produced.

After also the integrity of the mixing process has been validated, the thresh-

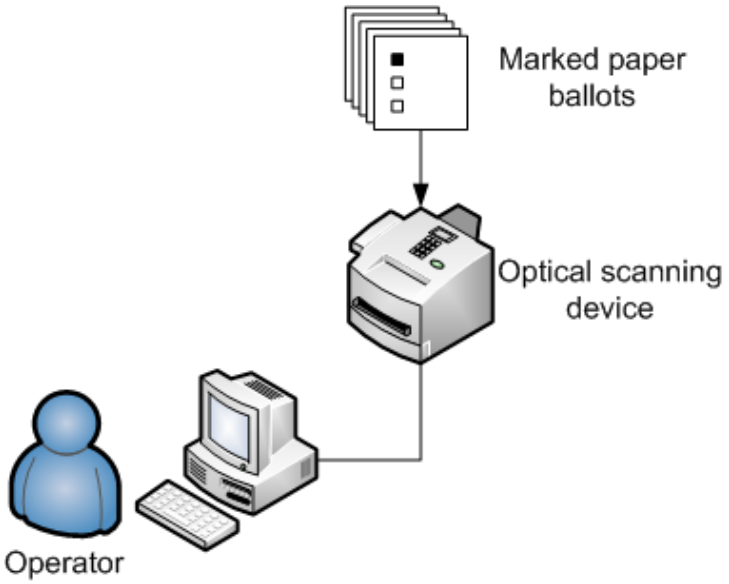

Figure 3.7: The scanning workstation

old of all the parts of the secret key can be combined and votes can be decrypted (by the Electoral Board). (Secure multiparty computation schemes. DESCRIBE! Then the list of all decrypted votes are digitally signed by the Electoral Board and provided to the counting process.

Recording and verifying paper ballots

The paper ballots are scanned in batches using optical scan technology. The scanning workstation showed in figure 3.7 consists of a document scanner connected to a computer installed with Readsoft Scan Software (ref!), and controlled by an operator. Optical character recognition (OCR)(briefly described in section 2.1.1) is performed real time to extract data from the images. The scan results in images in TIFF-format of the paper ballots and values extracted from the scanned images, which is transferred to another workstation for manual verification, called the verify workstation [67].

The verify workstation is installed with Readsoft Verify Software (REF), and is also controlled by an operator. The software installed can display uncertain results from the OCR, and the operator can make a judgment and then make decisions based on directions given by the Election Committee.

The resulting image files of the scan are transferred for electronic storage while the validated data are transferred to a Quality Assurance workstation

for further validation. In the Quality Assurance workstation two separate scans are compared for validation to ensure correct results. The operator of the quality assurance workstation control the results and can choose to discard the results if the two scans differ, or accept the results it the two scans produce an equal result. After the result is accepted the data are transferred to files in the same format as electronic votes. The data is now in EML-format(DESCRIBE!), identical to the data format from the electronic voting console and are exported, encrypted and transferred to a central collection server for counting.

Counting the votes

In the counting phase, votes are counted per candidate/party in order to create the results of the e-votes. This output is digitally signed by the Electoral Board and recorded in a removable device for being transferred to the settlement system. The system provides publishing of the voting receipts obtained from the e-counting process on a website. Voters can then individually verify whether their voting receipts are included on the published list.

Decryption and tallying

Because the Government should not have be able to trace a vote back to a voter, but at the same time they have to mark of against the electoral roll that the actual voter has cast a vote. The e-voting system is using a cryptographic double envelope scheme (described in 2.5.7 together with a mix net (described in 2.5.5) The process of a mix net can both prove the vote can not be traced back to the voter and that no vote has been lost or changed in the process [65].

The systems also uses threshold decryption, meaning that the decryption key does not exist until the votes have been randomized (outer envelope removed) and can not be traced back to voters. The decryption key is divided into several parts, shared by different parties of the election (with different interests). When it is time to decrypt votes these parties in possession of the key shares, have to come to the secure location to combine the parts and create a usable decryption key. This key can then be used to decrypt (open the inner envelopes) and the votes can be counted.

After the decryption, the person identifiable ballots and decryption key is deleted, so no one afterwards can decrypt the person identifiable ballots and

learn the content of these signed ballots from voters.

The decryption service of the election system is a standard system consisting of a mix net followed by a verifiable decryption [1]. No further details are published.

This decryption service decrypts all the ciphertext and publishes the resulting ballots in a random order.

Auditors supervise the processes of encryption and decryption (as explained in the next section).

3.2.6 Auditing

As noted, auditing is an important mean to ensure every step of the election is working according to planned. Not only should the user verify his vote was correctly recorded by the system, he would also want to know that his vote was counted in the tallying process. All the processes of the system has to be audited, and in the Norwegian system auditors supervise that the recording and tallying of ballots are correctly.

After the polls have closed the auditor receives:

1. The entire content of ballot box
2. A list of hashes of encrypted ballots (Generated/seen by the receipt generator)

Then the auditor can compute its own list of encrypted ballots that should be counted by the system.

The auditor verifies:

1. The content of the ballot box (signatures and proofs)
2. That no ballot has been inserted or lost, compared to the receipt generator list
3. That the list he has computed is the same as the list of ciphertexts inputs to the mixnet
4. The proofs offered by the mixnet and the decryption service

After verifying these points above, the auditor publishes hashes of every ballot. The voter can then also verify that his ballot was included in the counting process.

3.2.7 Authentication

The creators of a Governmental election system has to ensure that the election system has the necessary access controls needed to meet security requirements. To get access to cast a vote from the election web site, the voter has to authenticate himself using an electronic ID to prove he is eligible to vote in the actual election.

A new Norwegian electronic ID is in development, based on a common identity provider (a PKI called Common Authentication Infrastructure (CAI)). The authentication scheme is planned to include a national ID card (smart card) together with a card reader, to let Norwegian citizen identify on the Internet [65]. As with "MinID"(2.6.4) this scheme also federates authentication, and CAI authenticates using the SAML 2.0 protocol. The system requires two-factor authentication, with a password to be used together with the eID. According to [67], these passwords will be stored as "the hash-value of at least two secure one-way hashing algorithms" and a unique salt will be used when hashing each password.

A simplified use case of the process of authentication is showed in figure 3.8, where the identity provider (CAI) is handling the authentication. Both voters and election administrators, as well as auditors has to authenticate to the system. These users will have different roles in the system, authorized to perform different tasks in the system.

If eID is not developed and ready for the test project in 2011, authentication would be done using "minID" 2.6.4. By basing authentication on login using "minID", the voting system can then be sure that voter A actually is voter A because of log in through "ID Porten", using voter A's "min ID".

To vote in the election the voter himself has to make sure he has "minID" (but a lot of citizens already have "minID" now, because it is used in so many other public services).

The system stores details about the voter authentication process and any certificate that was used. This, and any stored additional information is digitally signed by the system, to guarantee its integrity and authenticity.

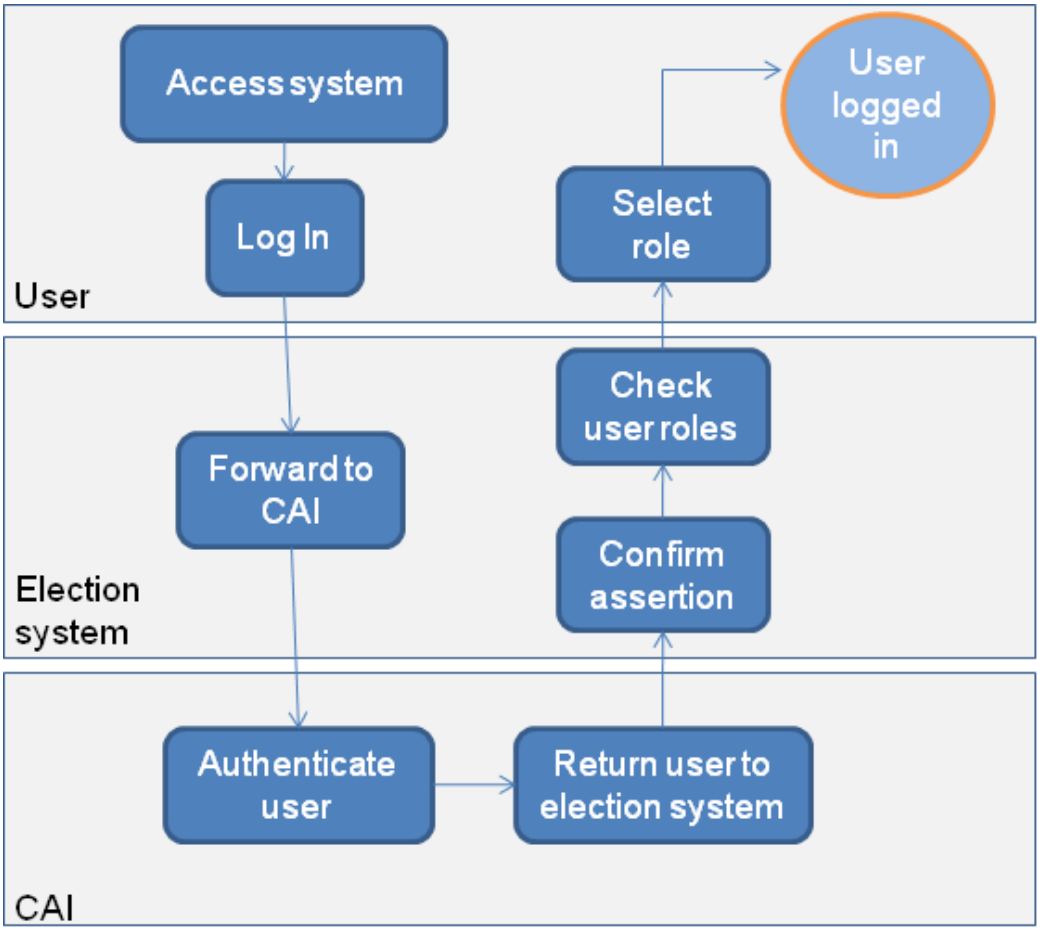

Figure 3.8: A simplified use case of the authentication process [67].

3.3 Threats, attacks and countermeasures

In this section I present some of the threats of an Internet voting system and the countermeasures the Norwegian e-voting system should hold. The voting system has to have countermeasures against several possible attacks.

"In practice, the two most significant security problems are compromised computers and coercion" [1]. No cryptography can protect a voter from coercion when voting from home or some public location, but the system should include features to hinder coercion. A "solution" to the coercion problem is the possibility of submitting multiple ballots [67] [65]. The system allows the voter multiple re-voting only conting the final ballot. It is also possible to vote at the polling station, and a paper ballot would overwrite any e-vote no matter what timestamp (submitted before or after a submitted e-vote).

The previous matter is a measure against coercion from external "attackers", but coercion can also be done by election insiders/officials [1]. A voter authenticates before casting a ballot, and the election official having access to the authentication system could detect any electronic e-voting by a voter. The election official can not see the content of the casted ballot, but it could see/detect the coerced voter casting a new ballot. A coercing election official having access to the counted ballots could also verify that the coerced voter(s) (his victim(s)) did not revote. If forcing the voter(s) to submit a ballot with the desired effect the coercer can observe the counted ballots and check if the ballot(s) are present among these.

Compromised home come computers is the other significant threat [1]. As mentioned in chapter 2.4, a notable fraction of home computers are compromised, and the Norwegian protocol has to provide the voter with a possibility of detecting ballot tampering without relying on the computers. This is complicated since the voter cannot perform any cryptographic computations without a computer, and this is were the method of using a receipt generator and pre-generated receipt codes is involved. The voter receives precomputed receipt codes on his voting card (as mentioned in section 3.2.4) with his voting card and after casting a ballot, the receipt codes generated by the receipt generator and the ballot box is sent to the voter not via the system and the computer, but through an independent channel (postal service). If a voter's computer is corrupt, the attacker can be able to see the voter's ballot, and the attacker can also modify the ballot. And therefore these security mechanisms allows the voter to notice tampering with high probability [1].

The system's countermeasures against other players in the protocol are discussed further in section 3.3.1.

The infrastructure is divided into a small number of separate players, based on technical measures, it can be assumed that an inside attacker can compromise at most one infrastructure player. There is one attack model the cryptographic voting protocol implemented for "E-vote 2011" can not protect against [1]. Assume the following scenario:

If the voter's computer and for instance the receipt generator responsible for sending the receipt code to the voter are corrupt problems arise. The computer can send the voter's real ballot to the infrastructure, but the corrupt receipt generator can then delay the receipt code to the voter. In the meantime the compromised computer can submit a forged ballot to the infrastructure, leading to the computation of a new receipt code. The receipt generator can then discard this new receipt and the first computed receipt code is sent to the voter. The voter believes his ballot was correctly received, but is really replaced by a forged ballot.

The cryptographic voting protocol can not protect a voter if both one particular infrastructure player and his computer are corrupt.

The e-voting protocol deployed for the Norwegian system will have a static corruption model, stating that an attacker may corrupt [1]:

1. any single infrastructure player (see figure 3.5) and any subset of voter's computers

2. or the receipt generator

In the election scheme the receipt generator will be so protected that it is very unlikely a possibility of it getting compromised.

In the description of attacks in the following section a simplified attack model is used. In this model an attacker may corrupt one of the following [1]:

1. Any subset of voters and computers

2. Passively* any one infrastructure player

*meaning eavesdropping of compromising privacy, but no active operations.

3.3.1 Attacker controlling Parts of the Infrastructure

In this section, based on the the description of the voting protocol [1], I explain how the full protocol will defend against potential active attacks from the infrastructure players. Here the voter and the voter's computer are also included (even though they are not defined as infrastructure players 3.5) because they have roles of attack or verification purposes in the protocol. Some of the active attacks the analysis mention could be:

- Ballot stuffing by corrupt voter
- Ballot stuffing by a corrupt ballot box
- Corrupt ballot box falsely claiming that a given ballot belongs/corresponds to another voter than the correct one
- Corrupt ballot box using a corrupt voter to submit the ballots of honest voters as it owns, and then learn the ballot content from the receipt codes

The countermeasure against active attacks from infrastructure players is based on all actors having to prove that they have executed the protocol as instructed. [1] For instance, the voter's computer has to prove it has knowledge of the ciphertext content and the ballot box has to prove the accuracy/correctness of its computations. As additional security measures to provide the integrity of a ballot and the sources, the use of digital signatures from a PKI hash functions are included.

If the receipt generator is corrupted it can leak the number of options on a submitted ballot, because the number of receipt codes are equal to the number of options chosen [1]. But this is not discussed any further because the receipt generator will be well protected and a make compromise of this component quite unlikely.

To prevent a corrupt ballot box from inserting forged ballots the mean of digital signatures is used. The voter, in cooperation with his computer, signs the submitted ballot with a digital signature from a PKI. The corrupt ballot box could create a fake digital signature, but does not the possibility of creating a digital signature correctly corresponding to this voter and his computer. The digital signature can also immediately prevent a corrupt ballot box of falsely claiming a given ballot to correspond to a different voter than it actually does.

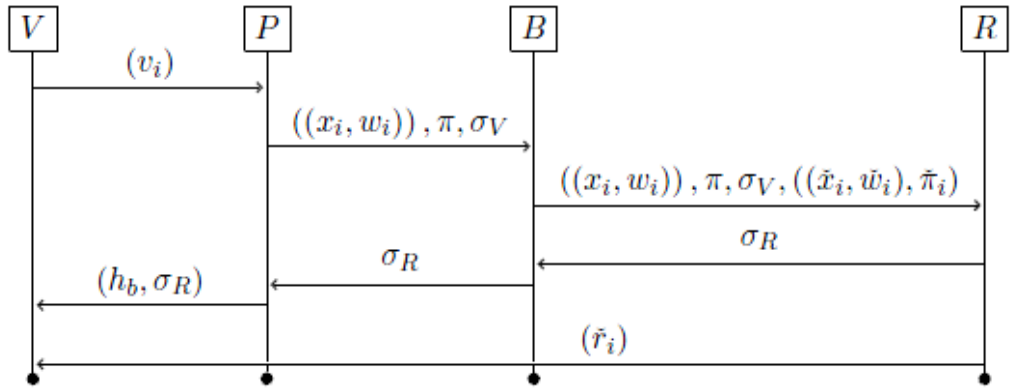

Figure 3.9: The "Shadow-mix shuffle proof" [1]

The feature of digital signatures also prevents a corrupt voter from executing ballot stuffing. Using signed ballots it is trivial to ensure that a voter can only register with one vote. "Any attempt to add rogue votes (digitally signed by untrusted digital certificates or voters not belonging to the Electoral Roll) is detected and prevented" [67]. An eligible voter can of course cast as many votes as he like, but the digital signatures ensures that only one ballot is counted per voter.

If a corrupt ballot box using a corrupt voter as a fellow actor submits ballots generated by honest voters as its own, the ballot box can theoretically learn the ballot contents from the receipt codes generated [1]. The countermeasure to prevent the chance of this is based on the computer submitting the vote(s) having to prove that it knows the content of every submitted ballot/ciphertext (x, v).

Not only the computer, but also the ballot box has to prove itself to prevent this attack. "The ballot box has to show the receipt generator the signed ballot and also prove that $(\check{x}_i, \check{w}_i)$ is computed correctly" [1](Recall step 3 in the vote submission process in 3.2.4). By doing this the corrupt ballot box and voter can not take advantage of the receipt generator' decryption capability to learn the content of honest voter's ballots.

The verification of the computation is done by the receipt generator based on the ballot box creating a proof of correct computation. To do this the receipt generator needs see the entire ballot and know the voter's digital signature and the computer's proofs of knowledge (created as mentioned earlier). The processes are showed in figure 3.9.

First the ballot box receives the voter's encrypted ballot from the voter's computer, together with a proof π and the voter's signature σ_V.

"To simplify the security proof, the ballot box also randomizes the ciphertext $(\check{x}_i, \check{w}_i)$"[1]. The ballot box computes the following:

$$(\bar{x}, \bar{w}) = (x^s, w^s)\ \hat{w} = \bar{x}^{a_2}\ (\tilde{x}, \tilde{w}) = (g^{t_2}, y^{t_2}{}_3)\ (\check{x}, \check{w}) = (\bar{x}\tilde{x}, \bar{w}\hat{w}\tilde{w})$$

Both $\hat{w}$ and $\tilde{w}$, which has connection to the voter's ballot, has to be hidden [1]. This is done using El-Gamal encryption with a random public key as a commitment scheme (keeping the value hidden, binded until revealed). Proofs of correct computation are created by the property that El-Gamal commitments are homomorphic, computationally hiding and unconditionally binding [1]. A binding property means that when decrypted, we can know that the outcome is the same committed to originally.

After the ballot box has computed a proof $\check{\pi}$ of its computation of $(\check{x}_i, \check{w}_i)$ everything computed up to now is sent to the receipt generator. The receipt generator has access to the entire ballot information and verifies the voter's digital signature and every proof.

After verification, the receipt generator creates a hash of the ballot and returns this signature σ_R to the ballot box which is instructed to forward the signature to the voter's computer. The voter's computer will not inform the voter that the ballot has been accepted before receiving this signature. If the ballot box for some reason discards a voter's ballot, the voter (together with an auditor) can prove his ballot was discarded. When receiving the signature σ_R the voter's computer "forward" this and a hash of the encrypted ballot h_b to the voter.

There are also security issues that would certainly be detected, but that is hard to do anything about. A corrupt infrastructure player would usually cause suspicious integrity failures and thereby stop the election. The Norwegian election system's countermeasures to detect such attacks is based on verification (see 3.2.3) of the processes and components with digital signatures. If an an attacker controls any of the infrastructure players, he can make the corrupt infrastructure player halt and thereby stop the entire election [1].

The system's solution against attacks that brings down the election system might be the use of an N-structure architecture, or the possibility of advance voting over a period of time so a system breakdown/corruption can be checked and does not ruin the entire election [12]. The scenario of an attacker controlling network or parts of it is discussed in the following section.

3.3.2 Attacker Controls Network

An external attacker or organization can be able to compromise a number of voters and a large number of computers, and thereby launch network attacks (like a DOS attack described in 2.4). An attacker controlling a large amount of computers can perform a DDOS attack to bring down the election server or other system components, by generating a such a load of traffic that the components cannot handle the legitimate traffic of ballots or other requests, or the components respond so slowly they are rendered unavailable.

A DOS attack can prevent the election web site or systems components from functioning efficiently or functioning at all, both temporarily or indefinitely. If an attacker controls the network between a voter's (or several voters') computer(s) and the election infrastructure he can also delay or block the submission of ballots [1].

To decrease the efficiency of a DOS attack, the Norwegian election system allows the voter to vote in a 3 month advanced period, making it harder to launch a DOS attack. In addition, to reduce the damage of such an attack, the system is considering to have an N-version architecture, in any parts of the system it is possible(REF: Christian Bull, security E-vote "2011"). By doing this there would be N versions (N being larger than 2) of a system component working in parallel, making it harder for an attacker to affect the system performance.

The use of an N version architecture, where the different could be made by different contractors, would also provide a better and more accurate election result. If one of the versions present a result that differs from the majority, it can be assumed this version has an error and the result can be excluded.

3.4 Discussion and Criticism

In the planning and development phase of the Norwegian e-voting system they have to take all issues with an electronic voting scenario into consideration. The system should fulfill all requirements of a modern voting system (recall the requirements listen in table 2.1), and especially address the security threats significant for remote voting over the Internet.

The Norwegian system offers all the phases of an election scenario, from preparing an election (not looked into in this thesis), casting a ballot, tallying and presenting the result. In the phases of a voting scenario and tallying,

mechanisms for providing the requirements of authentication, secrecy, integrity and verifiability are included.

The system , for the test pilot, is probably providing access control with an electronic ID (MinID), a method "most" Norwegian citizens have used before or are familiar with. We assume the voters in a larger degree will trust the security of access control in the system, when using this kind of mechanism. The problem with trusting the system will probably in a larger degree be regarding privacy and ballot tampering; if anyone (outsider or election authority) can see what ballot the voter cast, and that the vote is recorded as intended.

The double envelope scheme alone is not sufficient in an Internet voting scenario [1]. In such an approach to e-voting, the voters computer is fully trusted in the system. We do not consider this to be sufficiently in an election scenario, because as mentioned in 2.4; the voter's computer can easily be compromised. When casting a ballot, the ballot information would be available in the voter's computer memory in plaintext for a period of time, making it possible for any eavesdropping malicious application to retrieve the information. For instance a Trojan could be used to post on a bulletin board who voted for which party.

Therefore additional security mechanisms in the system are needed. The potential voter himself are not able to do any computations, so everything has to be delegated to the voter's computer. Compromised computers is one of the most significant threats to an Internet voting system ([1]), and therefore the Norwegian voting scheme puts no trust in the voter's computer. The Norwegian voting protocol uses mechanisms of proofs to ensure components are behaving as intended (no malicious behavior) and the user have possibilities of verification to ensure the ballot is recorded properly. Secrecy is provided with El-Gamal encryption, and a mix net providing proofs of operation that can be validated for the tallying process.

The compromised computer problem can in a certain degree be solved using the cryptographic receipt method, but relies on voters actually auditing the voting process. To detect if a compromised computer has altered the ballot, the ballot box and the receipt generator cooperate to compute a sequence of receipt codes for the submitted ballot. These codes are sent to the voter through an independent channel (SMS). The method of verification in the Norwegian voting protocol relies on a cryptographic assumption from the receipt generator. For this method to work sufficiently, the receipt generator has to be very well protected, so a compromise of the receipt can be assumed to be very unlikely [1].

Secrecy and means of integrity and verification is in the Norwegian system provided by cryptographic mechanisms. These mechanisms are considered and analyzed [1], and appear very strong, but there are still other issues that need to consider. The verifying part of the system depends on the receipt generator, as a trusted component, not being compromised and have to be strongly defended. The system would still be vulnerable to attacks by trusted insiders.

Usability is an important factor to consider when creating a computer system for the public to use. The developers of the Norwegian system are taking carefully precautions to create a user-friendly voting system for the public, but we have no insight to the look-and feel of the system other than described in the voting process. In the Norwegian system they assume the voter would verify his vote with the SMS received after casting ballot, which should not be a difficult process for the potential voter if guidance and a user-friendly UI is created. The security regarding attacks of ballot tampering relies on the voter actually auditing his vote.

Verification of the entire system and tallying process are done by independent computer experts, controlling the system before, during and after an election [65]. The system will be completely transparent and not only informing the general public about the concrete solution but also including a certification of the solution by a third party. The project will also be open source, so anyone who are able and want to can verify that the system actually does what it is supposed to do correctly. The idea is that complete transparency will lead to increased security in the long run, but with having an open source project an attacker would also have insight to the system construction, so this again requires stronger countermeasures.

The Norwegian system, being an government election system, has to provide measures against coercion. The system will allow the voter to submit multiple ballot, as many as he wish, for the entire advanced voting period, with only the last cast vote counting. This will make it harder for a coercer, because the voter could cast a new ballot later to replace the coerced ballot. It is also assumed that all voters in Norway will have the opportunity to go to a polling station, either in advance or on the polling day itself [65]. Casting a paper ballot will definitely overwrite any cast, coerced or not, electronic ballot independent of it is cast in the advanced period or on Election day.

Coercion is basically something that has to be dealt with outside the features of an remote Internet voting system (REF: security and trust in the Norwegian system). It is impossible to provide a coercion-free Internet voting scheme ensuring secrecy of a democratic vote, and now a group of represen-

tatives from one of the Norwegian political parties have put forward a motion to stop the "E-Vote 2011" project [68].

They argument that one of the most important democratic principles is the secrecy of an election. Coercion in any form should be impossible, and documentation to provide an evidence of a ballot cast (vote-selling) should not be possible to obtain. With an Internet voting scheme, the methods of casting multiple electronic ballots or paper ballot on Election day, makes a control of voter's ballots harder for an attacker, but not impossible. The representatives are strongly against any voting outside a controlled environment with an election official to control secrecy, will not ensure the voters' democratic rights.

As this thesis is written, it is still planned to carry out the test election. It is the authors opinion that it would be very interesting to perform the trial, and analyze the outcome of accuracy, attacks, and the public's feedback. The project is for now a pilot project that will be tried out on a rather small scale, but it will be large enough to gain a valuable realistic experience.

Chapter 4

Helios voting system

The Helios voting system [10] is a web-based open audit system developed by Harvard University and Ben Adida [69]. The voting system opens for new possibilities of verifying and auditing ballots cast over the Internet. Using Helios, anyone can create an election, invite voters to cast a secret ballot, compute a tally and then generate a validity proof of the entire process [41].

The Helios system is built on existing web programming techniques and cryptographic voting protocols, and is using the Google App Engine [70] to run the voting application. In this section I will describe the Helios v2, but the latest developed version is Helios v3. The change in Helios v3 from previous versions is the introduction of modularity of both services (election preparation, ballot preparation, ballot casting) and algorithms [71]. (Using different modules for the components of an election.)

4.1 Motivation and Goals

The motivation behind creating the Helios voting system was to create an universal verifiable voting system with public auditiability. "If my vote is supposed to stay secret, how can I verify that it was counted correctly?" [10] This question is proposed on the web site of the Helios Voting System. And this is a question a voter should ask himself. In a traditional paper based election you can be sure of the secrecy of your vote when you insert your paper ballot into the ballot box, but you have no possibility of verifying that your vote is actually recorded and tallied, you have to trust the election officials and the manual system behind the election. In an electronic

cryptographic voting system you can also maintain ballot secrecy with cryptographic techniques, but what about verification of the cast electronic votes? This is what the Helios voting system aims to achieve; secret ballots AND verifiable votes. You don't have to trust Helios system components, because the system also provides a mathematical proof that the election tally was correctly computed, and by this they call the Helios voting system an "open audit" election. An open audit voting system is the same as an end-to-end verifiable election.

The Helios system "provides a unique opportunity to educate people about the value of cryptographic auditability." [41] The development of this system aims to get "people more comfortable with a new way of voting" [72]. Helios does not have any expectations about becoming the system to be used for government elections, but that it is a step of influence so that open-audit systems may become the standard of voting systems in nearby future.

The long term goal of the Helios project is to "introduce a process of voting where people can get used to a tracking number for their vote to check and verify the vote online." [72] And by this give the individual citizen more control in future electronic government elections.

4.2 Description of system

The Helios voting system is an open-audit verifiable cryptographic voting system which allows voters to track their ballots, and also makes the election results open for public auditing by anyone. [41] The voting system is an open source online voting system, that can be used from a voter's home computer. The web-based system is accessed by the voter through his web browser. The Helios voting protocol is simplified compared to most complete cryptographic voting protocols, because it aims to focus on the central property of public auditability. [41].

The system is a so called open-audit voting system, universally verifiable, with several features included in the system for verification and control purposes. Helios being an open-audit system is based on the system providing the following features: [71]

- The voter can verify that his vote was correctly captured (Ballot casting assurance)
- All captured votes are displayed (encrypted) to the public

- Anyone can verify that the captured votes were correctly tallied (universal verifiability)

The core concepts of the Helios voting system are [10]:

- *The Benaloh protocol* [73], wich makes it possible to challenge the system and give ballot casting assurance. When using the system you can choose to cast a vote and/or only audit the process. A user is only asked to identify himself when casting a ballot.
- The use of *Homomorphic Tallying*, were proofs of correct plaintext is produced.
- That *integrity comes first, privacy second*. The voting system is all online so secrecy is hard, and some administrator, Helios or the election creators, have to know the private key of the election. But the integrity is provided through the mean of cryptography and mathematical proofs.
- Software logic and cryptography given by *Python & JavaScript*. This is used to encrypt votes in the users browser, for authentication, displaying cast votes, etc.
- *Open source* and publicly available system specifications. This makes the system transparent, open for analysis, and is making it possible for an able programmer to build a complete program to verify.
- For now, Helios is deployed on Google App Engine. This is to provide a certain scalability and take use of some of the existing defenses provided by the engine. (It is planned to deploy on Appache/Python/SQL [41]
- Easily customizable regarding the *look and feel*, language options and authentication possibilities. Authentication are by default done by email and password, but for instance in a student election the authentication can be done by university web credentials (students ID and password for university services).

The Helios voting protocol is closely related to Benaloh's simple Verifiable Voting Protocol [73]. It is originally based on the Sako-Killian Mixnet (section 2.5.5), but changing to use homomorphic properties for tallying.

Helios is using Google App Engine [70] which offers the ability to build and host web applications on Google's infrastructure, with scalability. The election server is running on Google App Engine, while on the client side a javascript/java combination application is running in the voter's browser. If

Helios is deployed for an election having infrastructure, the election server can be run on the organizations server.

Helios datastructure/infrastrucure consists of 4 main components:

- An Election Builder
- A Voting Booth
- A Ballot Casting Server
- An Audit Server

The system is using JavaScript, JSON datatypes, encryption is done in the voter's browser using LiveConnect. Technical details can be found on [71]

4.2.1 Casting Ballot

The feature, proposed by Benaloh's Simple Verifiable Voting [73], that Helios uses is the separation of ballot preparation and casting. The voter has to authenticate himself to cast a ballot in the election, but a ballot can be viewed and filled in by anyone at any time, without authentication. With this openness anyone, not only a voter eligible to vote in the election, is able to audit the entire ballot preparation process; the election questions and the encryption of a single ballot.

The voting process of preparing a ballot to cast is as follows between the voter and the Ballot Preparation System (BPS) [41]:

1. The voter begins the voting process by indicating in which election he wishes to participate.
2. The BPS leads the voter through all ballot questions and recording his answers.
3. Once the voter has confirmed his choices the BPS encrypts his choices and commits to this encryption by displaying a hash of the ciphertext.
4. The voter can now choose to audit this ballot (optional). The BPS displays the ciphertext and the randomness used to create it, so that the voter can verify that the BPS had correctly encrypted his choices. After this process is carried out the BPS asks the voter to generate a new encryption of his choices.
5. If the voter chooses to seal his ballot, the BPS discards all randomness and plaintext information, only keeping the ciphertext ready to be cast.

6. To cast the ballot, the voter is asked to authenticate himself. If the authentication is successful, the encrypted vote is recorded.

Helios elections are uniquely identified by a cryptographic hash of the different parameters that characterize the election. This includes the public key that is used for encrypting votes, the election questions and possible answers, and the minimum and maximum number of answers that can be provided for each question. This information can also be published in the voter's browser offering the voter a verification opportunity.

4.2.2 Cryptoprotocol and Tallying

The Helios Voting System implements advanced cryptographic techniques to maintain ballot secrecy while providing a mathematical proof that the election tally was correctly computed [41].

In Helios, the votes are encrypted using the browser-based ballot encryption program, where the representation of choices (the plaintext) is encrypted with El-Gamal encryption [41]. When the encryption process of a vote is requested, the ballot encryption web application is initialized. When this application is initialized, it does not access the Internet again until the vote is completely encrypted and ready to be cast.

Each ballot is encrypted using web-cased encryption. Encryption within the web browser is achieved using LiveConnect to access the Java Virtual Machine from JavaScript, for better performance [41]. This thesis will not go further into programming techniques.

Since the encrypted votes, when submitted are associated with a voter ID for verification, Helios has measures to ensure anonymization upon tallying. The voter has to be able to verify his vote, while no one else are able to learn which vote corresponding to which voter. In Helios the tallying process of encrypted votes has been both tried out implemented as an anonymizing mixnet and as a homomorphic aggregation (Respectively in Helios v1 and Helios v2).

Mixnet

To achieve anonymization, Helios v1 was using a mixnet to shuffle and re-randomize the votes (the cast ciphertexts) before joint decryption of all of the votes [41]. The shuffling and re-randomization was done by each trustee, and

both the shuffling and the decryption is connected to proofs of correctness (to be verifiable).

For this procedure,Helios used a Sako-Killian mixnet to provably shuffle the votes. The idea of mix nets and the Sako-killian mix net is described in section 2.5.5. Since Helios is using El-Gamal encryption, and the Sako-kilian mix net takes El-Gamal ciphertexts as inputs, these scheme was chosen for its simplicity and because it is easier to explain than some more complex protocols that could have been used. [41].

In addition to the integrity a sako-kilian mix net already can provide with the use of one "shadow mix"(section 2.5.5), in Helios the mix server is asked to produce a few shadow mixes [41]. Then the verifier in Helios provides the appropriate numbers of challenge bits, one for each shadow mix, challenging with a bit 0 or 1 depending on which challenge you want to utilize (i.e. reveal permutation and re-encryption factors on the shadow mixes or compare the difference between a shadow mix and the original mix).

If the mix server can correctly answer to all challenges the primary mix is correct with a probability $1 - 2^{-t}$ where t is the number of shadow mixes the mix server has produced. With a larger number of shadow mixes (t=80), integrity can be guaranteed with a high probability.

To avoid such heavy computations, described above, for every voter who requests it, the protocol is transformed so that the challenge bits are computed as a hash of all the shadow mixes [41]. Then these hashes can serve as non-interactive proofs, and be the basis of verification. Again this scheme only works if the number of shadow mixes is sufficient to provide a high enough probability of integrity.

Once all proofs are generated and the result is tallied the server deletes the permutation, randomness and secret key of the specific election, leaving only the encrypted votes, their shuffling, the resulting decryption and the publicly verifiable proofs of integrity.

Helios v2.0. has shifted to homomorphic tallying and this scheme is described in the next section.

Homomorphic Tallying

The first version of Helios, as described in section 4.2.2 were using the mean of a Sako-Killian mix net to provide privacy. Because of greater simplicity the second version of Helios has shifted to be using homomorphic tallying

instead. The reasons for this is both because it is easier to implement and easier to verify [52]. (It is easier for a third part to write verification code).

Recall that El-Gamal is as homomorphic encryption system which (as explained in 2.5.4) means that you can take the encryption of for instance one vote v1 and the encryption of another vote v2, multiply them together and you get the encryption of the product. The randomization factor of the encryptions gets added up in the exponent and the two plaintexts get multiplied together. In Helios they take advantage of this property of El-Gamal, but with a twist. [74]

In an election point of view, the necessary property is not to multiply things together but of course to add votes together. To maintain the secrecy of the vote in the tallying process, votes can be added together under the cover of encryption, and the ciphertexts can be added together without knowing the plaintexts. To use El-Gamal for this purpose, the El-Gamal encryption function can be modified into an Exponential El-Gamal function [52]. Recall the "traditional" El-Gamal cryptoprotocol in section 2.5.4.

In the El-Gamal exponential function g^m is encrypted instead of m, to achieve an additive homomorphism. This gives this encryption function:

Encryption $(m; r) = (g^r, g^m * y^r) = C$

As shown, the message m is put in the exponent, and it is possible to add things together rather than multiply.

The decryption function of the exponential El-Gamal involves a discrete logarithm operation.

Decryption $(c) = g^m = M$

To take the discrete log base g is ok for short limited size messages, but the discrete logarithm is a hard operation. For the function of homomorphic tallying it would improve they system using better cryptosystems which lets you do the same additive operations much more efficient, even it the messages are thousands of bits long [74]. The reason Helios is using El-Gamal is because it serves as a good example and it is easier to implement than more advanced systems [52]. ElGamel is also easy to use with joint key generation.

A ballot to be tallied in Helios is consisting of: [52]

- One ciphertext for each available answer to each question*
- A separate(disjunctive) zero-knowledge proof that each of the ciphertexts encodes to a 0 or a 1

- A separate zero-knowledge proof that the homomorphic sum of all ciphertexts for a given question is the encryption of one out of 0;1;...to the maximum, ensuring between 0 and maximum answers are selected for each question.

**Since Helios is using El-Gamal, the system is using a single ciphertext for each option of each ballot question. This is for simplicity and to easier perform the discrete logarithm computation needed for tallying.*

By this Helios is both providing a mathematical proof that ciphertexts were correctly decrypted and that the election tally was correctly computed. [41]

The proof the decryption of each of the ciphertext (as mentioned in the second point above) is based that the decryption of an El-Gamal ciphertext can be proven by the Chaum-Pedersen protocol [53]. Given a ciphertext $c = (a, b)$ and a claimed plaintext m, the prover has to show that $\log_g(y) = \log_a b/m$ (Recall the parameters of El-Gamal in 2.5.4). This scheme of decryption proof can be described as follows(DIRECT from [41]):

1. The prover selects a $w \in \mathcal{Z}_{\Pi}$ and sends both A $= g^w$ and B $= a^w$ to the verifier.
2. The verifier challenges with $c \in \mathcal{Z}_{\Pi}$
3. The prover responds with $t = w + xc$
4. The verifier checks that $g^t = Ay^c$ and that $a^t = B(b/m)^c$

Helios uses this this scheme to catch a cheating power, but again it is transformed into a non-interactive form with the Fiat-Shamir(REF!!!) so that the proofs of decryption can be posted publicly and re-distributed by observers [41].

Threshold decryption

Another security feature, regarding the tallying of elections that Helios provide as an option is threshold decryption, or distributed decryption [52]. After shuffling/homomorphic tallying the ballots, the election server are ready to decrypt the shuffled votes/the homomorphic tallied votes. If the election is configured with multiple trustees a process of threshold decryption is required. The election server needs all the trustees to "participate" to perform the decryption. This ensures that only the homomorphic tally of all votes is decrypted and never any individual ballot.

These trustees each have to have a public key, a decryption factor and a proof. Each trustee generates a typical El-Gamal public key (described in chapter 2.5.4) all using the same parameters of $(p; q; g)$. These public keys are then combined using simple multiplication to be used for decryption [52].

4.2.3 Auditing

The main purpose of the Helios voting system is the public auditability, and the system provides two verification programs. On to verify a single encrypted ballot and one to verify the processes of shuffling, decryption and tallying. "Both verification programs are written in Python using JsonLibrary for JSON processing" [41]. The voter can both audit the encrypted ballot before casting, and verify the cast vote in the system. The Benaloh cast-or-audit voting protocol [75] is implemented in Helios by the ballot verifier which ensures that an audited ballot corresponds to the fingerprint generated before the cast-or-audit choice. A bulletin board is created by Helios for the voter to check his vote is recorded.

Audit encryption

In figure 4.1 from a presentation by Ben Adida the idea of verifying an encrypted ballot is showed. The role of a ballot verifier is also showed as a helper to provide audit mechanisms in figure 4.2. In Helios this "helper" it is a part provided by the system itself, but if not trusting the system, a separate ballot verifier can be created by an individual competent programmer or organization, to run separate verification.

The ballot verifier provided by Helios, takes as input the JSON data structure returned by the voting booth audit process[41]. This data structure contains a plaintext ballot and its corresponding ciphertext. In addition it also contains the randomness used for the encryption and the election ID. Based on the election ID the verifier downloads the election parameters prints out: The hash of the election (election fingerprint), the hash of the ciphertext (ballot fingerprint) and the verified plaintext result of the ballot.

The election fingerprint, the voter can check against the fingerprint in the "voting booth", and the ballot fingerprint can be checked against the receipt received after encryption.

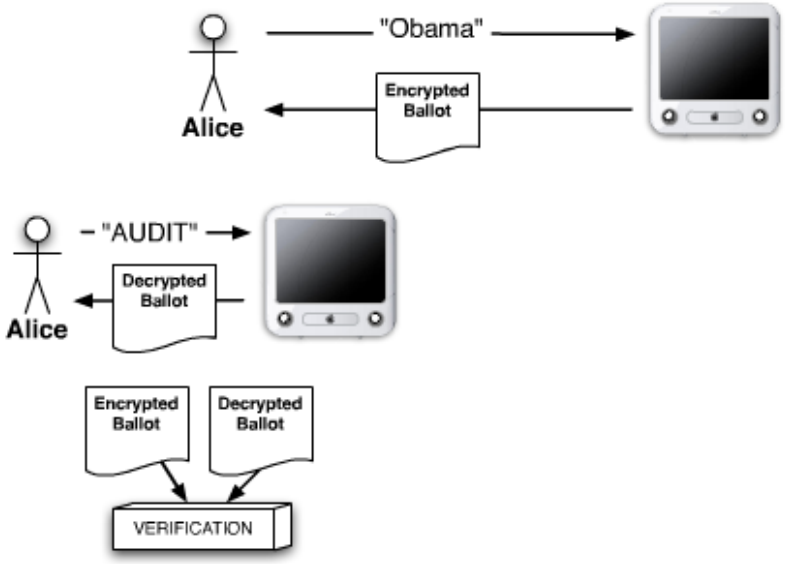

Figure 4.1: The idea of verifying an encrypted ballot [36]

Verifying vote is correctly recorded in system

After casting a vote with Helios, the voter receives a confirmation email containing the ballot fingerprint (and also the fingerprint of the election). Helios has a bulletin board of all cast votes in an election. On this bulletin board all the ballots are posted for everyone to see. A publicly available bulletin board is a standard feature of many cryptographic voting protocols [41]. Since Helios forgoes complexity, the bulletin board they have created is the simplest possible bulletin board, run by a single trusted server.

On this bulletin board in Helios the "ballot fingerprint" of the cast votes are displayed together with a voter name or a voter identification number. On this list the voter can find his voter ID and not only check that his vote is on the bulletin board, but also that the ballot fingerprint matches the one he cast and received in the confirmation email.

Verifying a complete election tally

Helios provide an application to verify a complete election tally. The auditor submits the election ID to the Election Verifier Program, and the process fist displays the corresponding computed election fingerprint.

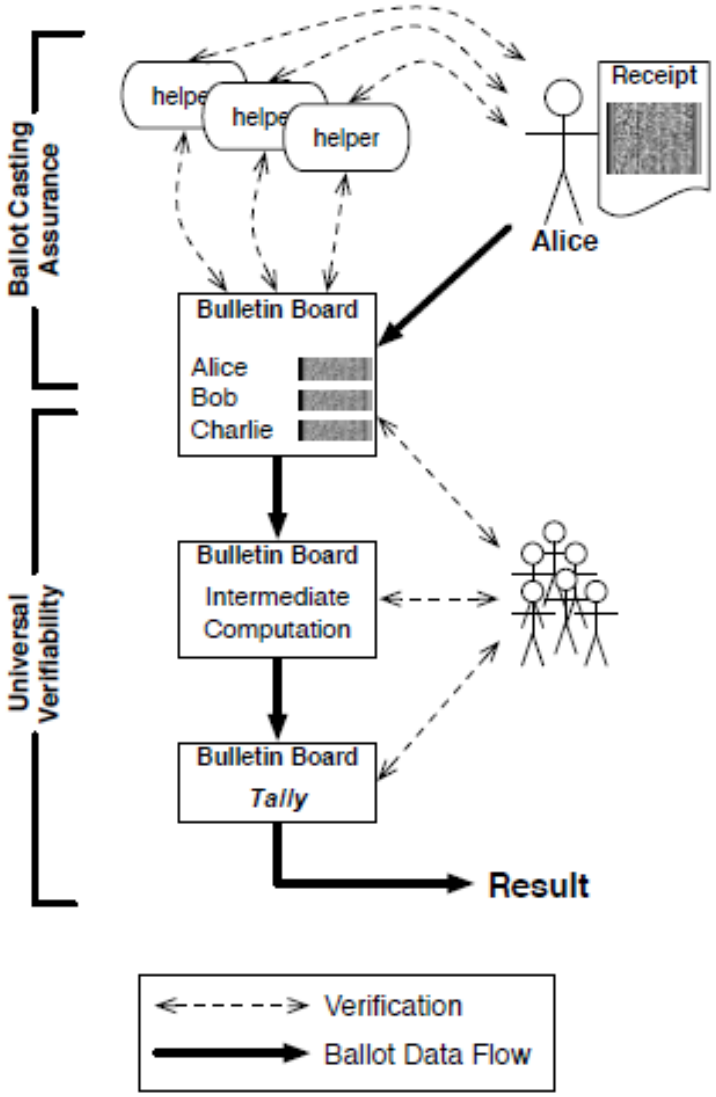

Figure 4.2: The idea of verification and ballot assurance of an auditable voting system! [36]

The verifier further ensures that the list of voters match the election voter-list hash. In the process the program displays the fingerprint of each cast ballot.

By verifying the proofs of encryption, it is check that each cast ballot is correctly formed

The program then homomorphically computes the encrypted tallies.

Each trustee's partial decryption is verified.

Then the partial decryptions are combined and it is verified that those decryptions, the homomorphic encrypted tallies, and the claimed plaintext results are consistent.

The complete results of a verified election includes; the election fingerprint, the list of ballot fingerprints, the trustee decryption factors and proofs, and the final plaintext countt. Any party who verifies the election should republish all of these items, as they are meaningless without one another. [12]

Part of this re-tally requires checking a partial decryption proof, which is almost the same, but not quite the same, as checking an encryption proof with given randomness.

It means that you can go to the audit web site. There, youll find a detailed specification that describes the file formats, encryption mechanisms, and process by which you can audit the election. Youre able to download every encrypted vote. You can verify all of the vote fingerprints by recomputing the fingerprint yourself. Each voter can check that their ballot is on that list, under the correct voter identifier. Then you can check that the encrypted tallying was done correctly, simply by recomputing it. And you can check that the decryption proofs check out.

And in the end, you can declare, with full confidence, because you coded it yourself and ran the code yourself, that given the published list of vote fingerprints, which individual voters checked, the result of the election was correctly computed.

The verification process of the homomorphic tallied elections are different than the previous mix based system.

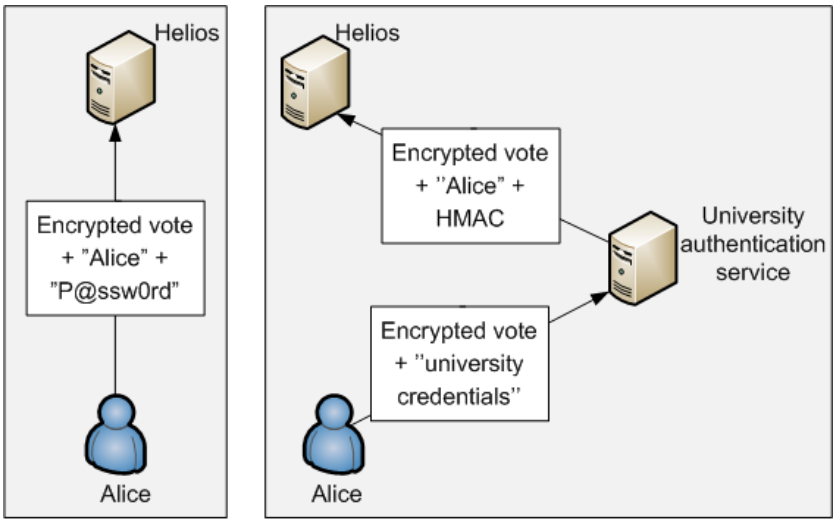

Figure 4.3: Authentication in Helios. (a) showing authentication with generated password. (b) showing authentication through separate university authentication service [52]

4.2.4 Authentication

The default property in an Helios election is to authenticate a voter by email and an election specific password, as shown in scheme (a) in figure 4.3. The password is generated by Helios upon voter registration and sent by email to each of the participants.

Using this form of authentication, a voter is identified with a name and email address, together with a random 10-character password generated by Helios.

The process of authentication can be modified to fit an institution's authentication infrastructure (for instance university user name/password) [71]. To enable this approach in a modular fashion, Helios is using a separate, trusted server to submit ballots on behalf of voters, as showed in figure 4.3 (b). This trusted server is configured to perform voter authentication accordingly, and then submit the results of this authentication action to the Helios server when the ballot is to be cast [52].

The authentication protocol that can be used to authenticate a call from the organization's authentication system to the Helios back-end server is the standard oAuth Protocol. [76](REF! oAUTH) "The protocol enables applications to access protected resources from a web service via an API, without

requiring the users to disclose their service provider credentials to the applications." The user can gain access to protected resources without sharing credentials with the application, and this mean is used in the voting scenario.

OAuth Authentication is done in three steps:

1. The application obtains an unauthorized request token.
2. The user authorizes the request token.
3. The application exchanges the request token for an access token.

A Token is generally a random string of letters and numbers that is unique paired with a secret to protect the Token from being abused (REF: oAuth). The token requests are signed by the application and verified by the server. The use of oAuth sends an HMAC of the request using a shared secret between the two servers.The signature process encodes the application secret and token secret into a verifiable value which is included with the request.

In the voting scenario with Helios, as shown in figure 4.3 (b) the authentication mechanism of the organizations infrastructure authenticates the user with his credentials. In Helios any user have access to the voting web site to prepare ballots, the authentication is not requested before upon ballot cast. So, for instance when a student wants to log in to cast an encrypted ballot Helios directs the voter to the university log in site. The user logs in using his university user name/password. The university server can handle the authentication and then send the encrypted vote and A voterID to the Helios server, together with an HMAC of the request as a digital signature using a shared secret between the university server and the Helios server.

In Helios v3 a separation of the ballot preparation server from the ballot submission server is created, so that preparing the ballot is independent of the eventual authentication request at submission time.

4.3 Examples of Helios in use

The Helios voting system was launched in 2008, public available on the web site for anyone to test [10]. But Helios has also been tested in more formal election scenarios. Both Princeton University in New Jersey (USA) and Université Catholique de Louvain (UCL) in Louvain-la-Neuve (Belgium) has deployed Helios in online university elections. [10] [52] A test election has

also been run by the International Association of Cryptographic Research (IACR) [9].

4.3.1 University Presidental Election

In 2009 the Helios voting system was deployed in an University President election at Unicersité Catholique de Louvain. An election open for 25.000 eligible voters, gave the Helios system a challenge of providing a trustworthy result. At this time Helios, based on mix net decryption, was not ready for deployment at UCL were votes were to be given different weights based on voter category. In the system, until now, verifiability was implemented as a proof of concept and not scalable for tens of thousand votes, this lead to the development of Helios v2.

In this election, as a student election, they did not care about coercion. The problem of comprised computers, botnets,virus, were mentioned, but the UCL election was not seen as a high stake election. Just in case, the election committee made available a set of secured client machines for voters wishing to use an official voting machine. The university's cryptographic department was running the election with guidance and help for implementation. The system was accommodated with specific customization and usability to fit the election, and figure 4.4 shows an example of how Helios was adapted to suit an UCL election.

The result of the election was that 5000 registered, 4000 voters participated in each round of election [52]. The accuracy of the system was crucial, because the winner differed only with 2 votes from the next candidate. Even though the polling stations with secure configured computers were open both days of election, more than 97% of the voters chose using a computer at hand. Approximately 30% of the voters checked their vote on the bulletin board, and no results raised any suspicion about the behaviour of the voting system.

All the data needed to audit the election were made publicly available, together with complete specifications describing the operations needed to audit the election. An independent company wrote a second instance of the audit code (in Python) on behalf of the election commission and successfully verified the election. "The use of two independently produced election tally and verification codes, written in two different programming languages, relying on different arithmetic libraries, provided strong evidence of the correctness of the decryption process" [52].

Figure 4.4: Screenshot of voting interface for choosing candidate option in the UCL election [52]

4.3.2 IACR Test Election

For over two years, the IACR board has been debating a proposal to replace the current mail-based system that is used by the IACR for its internal elections with a cryptographic e-voting system. Helios was one of the systems presented to the IACR on a workshop and it was chosen to perform a test election with the system [77]. Out of 1542 eligible voters, 379 votes were cast [78].

They chose to keep the Helios election server running on the Google App Enginge Infrastructure and not the IACR server, for the reason that it is a low stake election. There is no gain to obtain for external attackers, while members of IACR, that have almost unlimited control over the IACR server, are the ones running as candidates in the elections. This could, even though tallies are universally verifiable, offer opportunities for ballot stuffing or targeted DOS attacks.

To minimize the threats of attacks bu IACR insiders, authentication was not done by IACR reference number and passwords, which could be seen by the IACR server administrators. Instead an IACR server administrator pulled a list of all IACR members and sent this to the election administrator, which

uploaded it to the election server. Then one time usernames and passwords for this particular election could be generated by the election server and sent to members via email.

Tallying was initiated by the IACR returning officer, with all the members of the IACR election committee as trustees. In addition to the audit functionalities provided by Helios, the Crypto Group Universite catholique de Louvain also operated an separate independent election monitoring system (*http://www.uclouvain.be/crypto/electionmonitor*).

The test election went as planned, no issues with the functionality other than issues regarding having JAVA installed was a necessary condition for encryption. They concluded that Helios was an appropriate tool if providing more client options to offer to the members of IACR.

4.3.3 University Student Election

The University of Princeton has also deployed Helios for a student election of approximately 16.500 voters. The election went according to plan, and no significant problems occurred. The system was running on Google App Engine which provided scalability when the load of voters was high, and since they were using multiple trustees their biggest concern was to be a 100% sure none of the trustees lost their part of the key which would prevent the election from being tallied. The only complaint was from a student that did not have the JAVA platform installed on his computer. For usability the system was not significantly adapted, almost used as it is online, only including university logo and steps of information. In addition videos for guidance was created. The election committee of Princeton summarized using Helios for the election as "ease of use, lack of problems!" [79]

4.4 Threats, attacks and countermeasures

In this section I have described some of the attacks the Helios voting system can be vulnerable to, and attacks they have taken certain precautions to prevent.

Attacks on Helios v1 using mix net based tallying could include a corrupt Helios server providing incorrect shuffling or decryption. A corrupt Helios server that attempts to shuffle votes incorrectly or decrypt the shuffled votes incorrectly, would probably just take one auditor to detect the tampering.

So the probability of catching these types of attacks via cryptographic verification is good.

A requirement to prevent attacks regarding the mixing phases, is to create a sufficient number of shadow mixes for verification. If not having a sufficient amount there can be a non-negligible probability of cheating. "If a cheating power can produce many shadow mixes until it finds a set whose hash provides just the right challenge bits it can cheat without being detected" [41].

Some other attacks the Helios system is vulnerable to is [41]:

- Ballot tampering
- Voter impersonation
- Displaying corrupt ballot

4.4.1 Ballot tampering or voter impersonation

If Helios (server) is corrupt, it could try to submit/change a vote for a given registered voter. It could substitute a new ciphertext to replace the actual ciphertext of a voter, or it could inject a ballot for a real registered voter that was eligible to, but did never cast any ballot in the election [41]. If Helios is corrupt, it might also ahead has tampered with the step of encryption so even if the voter's browser is offline when encrypting, it might record the vote differently. By performing the auditing step, the voter could discover if the encryption process was erroneously performed.

If Helios tampers with the cast ballot, the voter can discover this if he uses the possibility of controlling his receipt against the bulletin board of Helios. A voter could then see that his vote is not recorded as intended in the system. But as I will describe later this may not be a sufficient step of verification.

The Helios server also knows the user name and password for all registered users, and if corrupt it could authenticate and cast a ballot for any user. The ballot submission server of Helios is hosted separately, and could even be distributed among trustees, but knowing the generated user names and passwords the system could still authenticate and cast ballots on behalf of the registered voters [41]. In this cast the ballots are corrupted before submitted to the system, and no errors regarding the tallying and decryption phase would occur. An alert voter, controlling the bulletin board at a later time, would still be able to see that a new vote is registered in his name. But if

not casting a vote in the first place, the voter would not likely even look at the bulletin board.

So if the server like this is able to authoritatively claim that a ballot comes from a particular voter, the chance of ballot stuffing is possible. The Helios server could even stuff the ballot box near the end of the day, to decrease the possibility of voters detecting the tampering.

In the case of an open-audit voting system, like Helios, the defense against this is to ensure that the voter list (bulletin board) is published and then include a period of time for an audit phase after the election ends. Then verification could be done during this audit phase, requesting the voters to verify the hash of their ballot, and ballot stuffing would most likely be detected at this time.

4.4.2 Displaying corrupt ballot

Another issue if the Helios server is corrupt is if the system is presenting a false ballot to the voter. The voter might think he is selecting one candidate when marking the option next to the presented candidate's name on the ballot, but is really selecting a different one [41].

This kind of attack would not be detectable by the verification method with the hash value presented on the bulletin board, independent on how many voters auditing the bulletin board. The receipt generated and sent to the voters during the ballot cast process would be incorrect, but again match the value displayed on the bulletin board.

One measure of protecting (trying to protect) against such attacks is through the mean of "authentication upon cast". Since Helios is offering public auditability, a user is not asked to authenticate before actually casting the ballot. This makes it harder to launch attacks were voters are being individually targeted and presented with false ballots, but not impossible (can authenticate voters or for instance IP information, etc). The only countermeasure Helios can really provide against such attacks is the ballot verifier and trust that voters actually uses this mechanism for verification. A voter can audit as many times as he likes before choosing to submit one of the ballots.

Helios could also include a browser based verification, to verify for the voter that Helios is presenting authentic ballots(using digital signatures), or some kind of two-way authentication mechanism.

4.4.3 Countermeasure by audit

In some of the problems mentioned earlier, the verification using the bulletin board is seen as a solution. But since the bulletin board is displayed by Helios, a corrupt Helios could also display a falsely bulletin board.

So, basically the countermeasure Helios has against these attacks are the auditing feature. By using the ballot verifier created by Helios (open source so everyone can verify it is actually performing as intended) or by creating an independent ballot verifier these attacks can be countered.

At the end of an election, Helios provides the tally together with the list of voters and corresponding encrypted votes (bulletin board of cast votes). Using the mix net, Helios can also provide supporting evidence for the tally by providing a mix-net and decryption proof corresponding to the list of encrypted votes [41]. The proof can be checked by auditors to verify the integrity of the results, and the election tally and the list will be published again for voters to audit that their ballots were recorded correctly as well.

Helios v2 and v3 are using homomorphic based tabulation, which is easier to verify. As the votes are tallied as a whole and no single vote is revealed, the secrecy is even better provided.

Another potential attack against an election, not being a technical issue, could be the case were a voter exploits the verifiability property of Helios [78]. If a user claims the fingerprint or code is wrong, or a component is malicious, he can cast doubt upon the whole election system. And as I mentioned, the public's trust in the system is an important requirement of success.

Links to the additional information that are included on a ballot question, could lead to attacks. For instance could targeted malware be injected, presenting a fraudulent Helios client to the voter, register what he is voting.

Helios does little to specifically counter the threat of client-side operating-system or web-browser compromise; a specifically targeted virus could surreptitiously change a users vote and mask all of the verifications performed via the same computer to cover its tracks [52].

4.5 Discussion and Criticism

The Helios voting system is a voting system targeting low-coercion election without putting high risks at stake (student elections, organizations, etc). The system therefore focus on mechanisms regarding auditing so the voter can ensure his vote is encrypted, recorded and tallied correctly. The system offers complete transparency, making all technical specifications, source code, etc. available to the public. The online system provides a great advantage to an election, since voters can access the system from any computer connected to the Internet, which allow allows for greater flexibility of the election period, less pressure on the voter to complete their vote quickly and should encourage more people to vote [7].

Since Helios does not assume the threat of potential external attackers to be very high, the means of attack prevention relies on auditing. The cryptographic protocol does not prevent ballot stuffing by external attackers or the Helios server itself. Only individual voters can check the validity of cast votes, and it the server is corrupt ballot stuffing or impersonation can be carried out right before an election ends. Helios also accepts the risk that if someone compromise the Helios server before the end of an election, the secrecy of the individual ballots may be compromised.

Helios depends on having a large number of voters actually auditing their ballots, if not Helios does not provide any verification beyond classic voting systems. Verifying the ballots can be done by the Ballot verifier provided by Helios, but if this component is corrupt, external software written used Helios specifications can also be used [80].

When a voter wants to check his vote is correctly recorded the Bulletin Board can be used. If Helios is corrupt and present one code to voters, another to auditors this would lead to problems, but an independent bulletin board can also be created, like they did for the UCL election. They created their own copies and requested voters to check at the end of election, before tallying [52]. Helios expects the voters to check their ballots, and auditors to check the bulletin boards integrity over time. This simplification of the bulletin board is in order "to focus the user on the major advantage of the system; auditable by anyone!" [41]

The means of verification to ensure that the tallying is performed correctly are strong, with a fairly low possibility of any erroneous procedures not being detected. Even if all election administrators, a fully Helios, are corrupt they cannot convincingly fake a tally and cheat the election results without a high

probability of getting caught [41]. The entire digital audit trail can be verified externally and provide an independent confirmation of tally correctness [80].

In cryptographic voting protocols, there is an inevitable compromise; "unconditional integrity or unconditional privacy" [41]. If every component of a voting system is compromised, only one of those two properties can be preserved. Since the main focus of Helios is open-audit voting the creators of Helios hold the opinion that the more important property is unconditional integrity. Integrity does not depend on trusting Helios, election results rely on the property of universal verifiability, while privacy is ensured by recruiting enough trustees and hoping that a minimal subset of them will remain honest. In a traditional paper based election you can be sure of the secrecy of your vote when you insert your paper ballot into the ballot box, but you have no possibility of verifying that your vote is actually recorded and tallied. This is what the Helios voting system aims to achieve; secret ballots AND verifiable votes.

If choosing Helios to administrate the election, there is only one trustee; the Helios server itself, and privacy guaranteed only if trusting Helios. As a E2E verifiable system we have noted that "trusting the system" is no longer sufficiently. A voter does not have the necessary knowledge to actually verify that processes is done correctly, and rely on independent parties they trust to audit the system and provide separate ballot verifiers.

A cumbersome property about using Helios for election is regarding the administration access []. The user logging in to Helios with a Google Account and actually creating the election, is the only user allowed to invite voters or create and make changes to the ballots in the preparation phase. Even though mulitple trustee's is chosen to administrate the tallying, these are not allowed to do any other administrative steps than to create keys and upload for tallying. This should be modified, since organizational or university elections often are created and performed by a committee, and can not share a Google login account.

It is assumed that open-audit voting will lead to more complaints and potentially DOS attacks on the auditing process [52]. But findings by Helios in use instead showed that complaints are likely to be more easily handled in open-audit elections because evidence and counter-evidence can be presented.

The creator of Helios states that the system in its current form is not ready to handle elections for public offices, even at a local level. Helios is one of the most complete implementations of an open audit election, but the creator has no expectations that Helios ever becomes the system for government

elections [72].

Chapter 5

Usability Test - Helios

The Helios voting system described in chapter 4 is public available on the Internet for everyone to test. Everyone can create a test election, invite voters to cast ballots and perform a tally of the results.

We created and ran a test election with Helios with ballot casting, verification and tallying, and then performed a usability test of the system. Usability versus auditability is an issue discussed [37], and Helios being an open audit voting system, so we wanted to test some user feedbacks on the systems properties and usability.

The usability test does not take the administrative parts into consideration, so the election results and the process of tallying has not been included. Neither has the verifying election mechanism, since the usability test was carried out in connection with the voting process.

5.1 Creating the test election

To start create an election using Helios a log in using a Google account is required (since Helios is running on Google App Engine) before you can to choose to create an election. The further steps of creating a test election to administer was as follows:

- Name the test election
- Choose between three options of election administrators

  Administer the election yourself

Figure 5.1: Screen shot of administrator options when creating an election

Let Helios administer the election

Have multiple trustees to administer the election

- Create ballot, add questions/options to the election
- Add voters to the election

  Configure the election with open or closed voter registration

- Freeze the election and it is ready for voting

When using Helios for an election, depending on the security of the election, the option of multiple trustees is optional, as shown in the screen shot in figure 5.1. The administrator of the election can choose to generate one private key for the tallying, keeping this key hidden, or he can also choose to trust Helios with the key administration. After login in and naming the election I had to choose between the three options of election administrators.

When choosing the creator to administrate the election alone, the system generates a new El-Gamal keypair for the election (public key and secret key). The election administrator is then the only one in possession of the election's private/secret key. Helios would not save the secret key in it's system so the administrator have to save it for the later tallying.

If choosing Helios as the administrator, Helios would possess the secret key of the election, and can perform a tally upon request.

If choosing the option of having multiple trustees to administer the election you can choose up to 5 trustees for this. The trustees are added in the system by email and every one of them receives an email with a link to their "trustee homepage" of this particular election. Each of the trustees has to go to this "trustee homepage" and generate their individual public and secret key. The trustees then has to upload their public key to the system, by a password they also received in the email, while copying and saving the secret key on their computer. When all trustees have generated and uploaded their public keys, the keys are combined into to one public key. The election can not be initialized before the combined public key is created.

Then as point number 4 and 5 indicates, the next step was to create the ballot and add voters. The ballot is created by adding "questions" and corresponding options. You can also indicate how many options/answers the voter is allowed to mark off and for each of the options it is possible to include URLs to additional information. This could be a link to a web site of the political party or a the statement of a certain candidate.

The voters to participate in the election are added writing in a voter name/voter id and a corresponding email address. In this voter list you have options to categorize the voters, see the status of voted/not voted, and of sending the voters emails containing their voter password. If using the email sender provided by the system to invite voters, the same email is also sent to the administrator letting the administrator (in this case me) know the passwords of all the participating voters, generated by the system. Another option on the "Voters" site is to choose between open or closed voter registration. Having a closed voter registration means that after freezing the election, which is the last step before voting can begin, no more voters can be added and allowed to vote in this particular election. Having an open voter registration just means that the list of voters is not fixed at the time that the election begins.

Freezing the election the administrator receives a SHA1 hash of this specific JSON object, which is the elections fingerprint.

5.1.1 The test election

The test election created for the usability test only consisted of a total of 2 ballot questions. The first question requested the voter to issue his choice of

candidate for prime minister, and had 6 corresponding options. The voter was asked to check 1 option, and the options consisted of fictitious candidates (Donald Duck, etc.). The second question requested the voter to check 2 parties to form a coalition Government. Also here there were 6 possible options to choose from (all with ficous party names).

For the ease of operation of the test election, we chose having one election administrator and configured the election with open voter registration. This was done to be able to keep adding test persons that accepted to participate during the entire test period, to get as many participants as possible. And also for the simplicity of tallying.

5.1.2 Vote in test election

To cast a vote in the test election the voter followed the URL in the invitation email to the election's home page. From the election home page he chose to vote in the election and followed the URL to the "voting booth" (which is a single-page web application).

When following the link to the voting booth the application was loading and downloading the election parameters including the El-Gamal public key. The questions of the election was displayed, and choices can be marked among the options and click back and forward between these ballot questions.

A screen shot of how one of the ballot questions looked like is shown in figure 5.2

During this process the application are supposedly not connected to the voting system until after the ballot is encrypted and it is requested to submit the ballot. Transitions are regarding to Helios specifications [41] handled by a local JavaScript function call and templates, and the ballot question choices are recorded in the local JavaScript scope [41].

5.1.3 Encrypting and verifying ballot

When encrypting ballots a loading bar with percentage showed the progression of encryption. When clicking the encryption button this triggers the JavaScript code to encrypt the selections via LiveConnect [41]. After complete encryption the ballot fingerprint of the ballot was displayed. The ballot fingerprint is the SHA1 hash of the resulting ciphertext after encryption of ballot. After encryption an audit of ballot could be done to see the encrypted

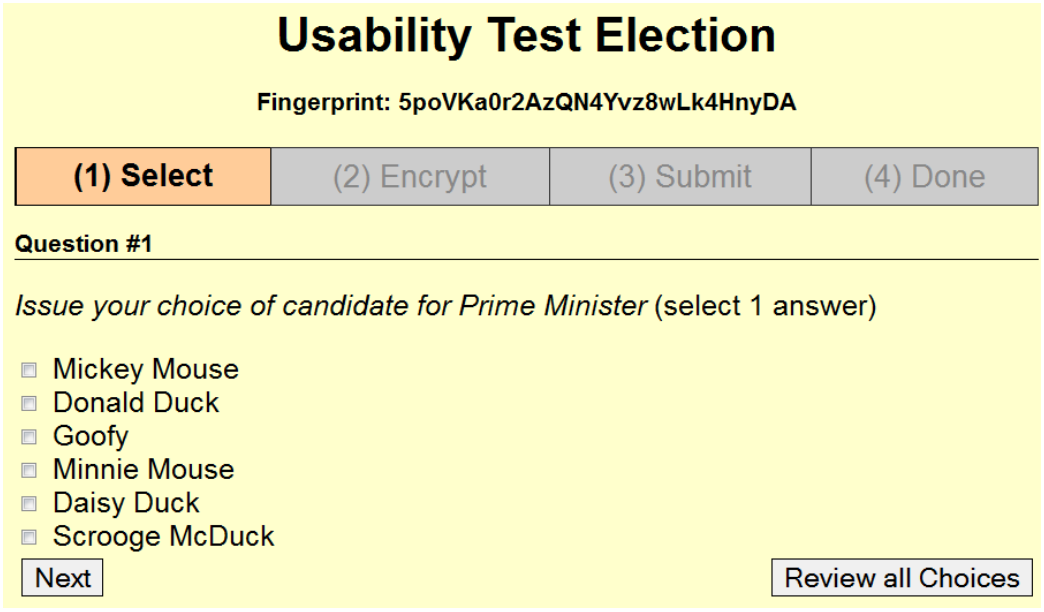

Figure 5.2: A ballot question displyed in Helios

ballot corresponded to issued choices, or the already encrypted ballot could be submitted.

The process of encryption displays a status bar as shown in figure 5.3

To verify the ballot, the system displayed a text field with ballot information, containing the randomness used in the encryption of the options. By pasting this information into the Ballot Encryption Verification Program (in Python), provided by the system. When auditing the ballot the ballot fingerprint is checked against the election fingerprint, shows the decrypted choices

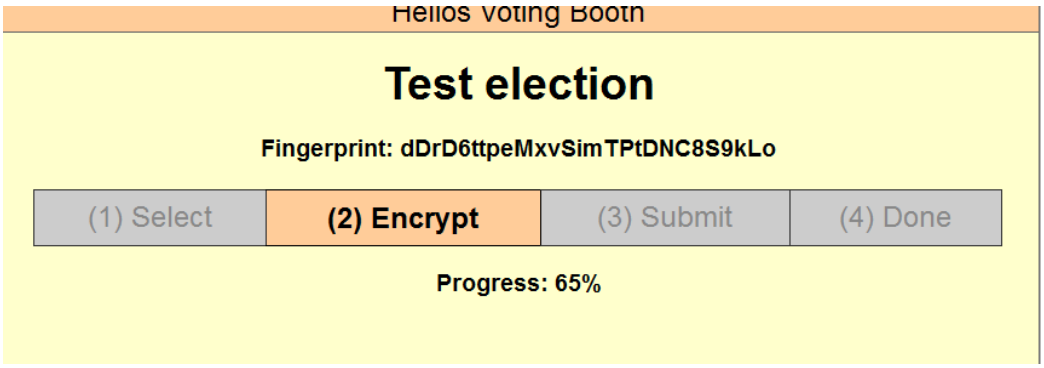

Figure 5.3: The process of encryption

and gives a feedback that encryption and proofs was OK. The information is displayed like this:

Election fingerprint: hbsGEZqp8k4nAj9BHwlDK9nN8ug
Ballot fingerprint: zIJVICG+oMtYpCv8mHW2J3DpK04
Election fingerprint matches ballot
Ballot Contents:
Question #0 - President : Candidate I chose
Question #1 - Party : Party I voted for
Encryption Verified
Proofs ok.

When finished the auditing process and going back to submit the ballot, the JavaScript code has cleared its encrypted ballot data structure, so the ballot has to be encrypted again before submission with a new randomness. This results in a new ballot fingerprint.

5.1.4 Authenticating to submit ballot

To submit an encrypted ballot authentication is requested. A voter registered in the system can request a password for authentication by email. When requesting password Helios sent out an email containing election fingerprint and generated password.

When submitting login information from the generated email, a success message was displayed by the system. What happens in the background is the JavaScript code intercepts the form submission and submits my email, password and encrypted vote [41]. The Helios server also generated an confirmation email of the encrypted vote and the fingerprint (SHA1 hash).

After submitting ballot, the fingerprint received in confirmation email could be compared against the Bulletin Board of the election. The ballot fingerprint is displayed next to voterID.

5.1.5 Convenient Voting Process

The voting process should according to the requirements of a modern voting system (table 2.1) be a convenient process. It should be easy and efficient, not too time consuming. In the next section we look into the usability aspect of a voting system, but first some measures of efficiency and convenience are mentioned.

To cast a ballot using the system, there should not be to many steps to go through. Table 5.1 shows the number of clicks to perform certain tasks in the Helios voting system. The estimates are done from the home page of an election (after you have chosen which election to participate in) and the election consists of 2 questions with 1 option to mark on each.

	Clicks to cast ballot
Knowing authentication info	9
+ auditing encryption	17
Requesting authentication info*	10
+ auditing encryption	18

Table 5.1: The number of clicks of casting a ballot (*requesting authentication information, also requires the step of the voter opening his email)

The process of casting a ballot with Helios does not require many steps. When introducing the steps of verification, to provide the necessary security, the simplicity is affected. The process of auditing the ballot encryption before casting is less convenient for a voter, but an important step of verification. Usability, considered in the next section, is also an important factor to provide a convenient voting system.

This thesis has not explored performance of voting systems, so measurements of shuffling, decryption, tallying or verification of tally are not included.

5.2 The Usability Test

A voting system is designed to be used by the public. People with different backgrounds, language, and technical skills should be able to cast a vote using the system. A handful of open-audit election systems exists (DEFINE!), but very few of them have been critically looked at in the term of end user usability. And there is clearly issues regarding auditability versus usability. The question if auditability and secrecy limit usability? Or if the complexity of casting a vote or auditing it defeats the purpose of having such systems? [37] In this part of the thesis I carried out a usability test of the Helios voting system. The overall purpose of this test was to get an idea of how user friendly the Helios voting system is for a group of potential voters.

The following sections are a compilation of the methodology used to conduct the test, the test subjectives, the data gathered during the test, and a evaluation of the results. Such a questionnaire has not (that I know of) been

performed on the Helios system before, but in section 5.2.6 I include about a related usability study of the system done by observing voter behavior.

5.2.1 Test Procedure

The test procedure consisted of the following two main parts for the participants to go through:

1. Vote in a test election (created as described in 5.1).
2. Answer an online questionnaire about the voting process (created with *Kwiksurveys* [81]).

Invitations to participate in the test were sent out by an email containing information about the test with links to the election and the survey, and instructions of the user tasks. The user tasks are described in 5.2.4. I first started generating invitation emails (with election passwords for the voters) by the system, but this process was so inefficient I changed to have the users request their own passwords upon authentication and just including the election URL and test steps in the invitation email.

The test was designed to evaluate the usability and effectiveness of the following elements of the Helios voting system:

- Creation of ballot (selection of options).
- Encryption and verification of ballot (audit with ballot verifier).
- Ballot submission (with process of authentication).
- Verification that ballot is correctly recorded (Bulletin board).

In addition I included some extra questions in the survey to see if the system gained trust with the participants, and how the participants considered internet voting in general.

The questionnaire consisted of 17 statements regarding both the overall voting process and specif parts of it. The statements were to be graded from strongly agree to strongly disagree, and an optional field for additional comments was also included below each statement. In the end a yes or no question about Internet voting was also included.

The experiment was carried out until I exceeded 40 participants, which was approximately 2-3 weeks. Since I could not force people to participate in my

test and answer the questionnaire, I had to recruit more participants during this period if people did not respond.

5.2.2 Limitations

The usability testing of the Helios was done using the public available voting systems through the website. The system was not implemented using the open source code, and neither adjusted regarding user interface. To improve the usability, the system can be adapted for every individual election with different targeted user groups.

The usability test was aiming to get a potential voter's perspective, and did not go further into the usability of the administrative parts of creating an election. The focus was only on the voting process.

About the accuracy of the test, It is important to note that the testing process was entirely orchestrated and the findings does not provide exact information about how all voters will react to the web site and the voting process. The results are only based on the participants constructive feedback, no observation actions were performed.

Since this test considers the voting process itself it does not include any usability analysis regarding the process of verifying the tally.

5.2.3 Test participants

To carry out the test, 60 emails were sent out to "recruit" individuals to participate, and resulted in a total of 41 participants during the test period. Table 5.2 shows the overview of test persons.

5.2.4 User tasks

To test the functions of the Helios voting system, some steps to go through was given to the test individuals to carry out. The goal was to analyze the usability of casting a vote, verifying the vote and also control that the vote was correctly recorded. The questionnaire was about usability, the ease of orientation on the web site, and effectiveness of these processes.

In the invitation email the user was instructed to follow a link to the "Test Election", and then the tasks/steps for the user to carry out included:

Gender:	Male: N = 26 Female: N = 15
Age:	18-20 years: N = 2 21-30 years: N = 33 31-40 years: N = 3 41-50 years: N = 3
Education:	Master level: N = 33 Bachelor level: N = 6 High School: N = 2 Technical education(engineering): N = 29 Non-technical education: N = 10

Table 5.2: Overview of test participants.

Task 1 Enter the "voting booth" and prepare an encrypted ballot.

Task 2 Audit the ballot to verify the encryption was correct and corresponds to your choices (using The Ballot Verifier).

Task 3 Encrypt the ballot again and authenticate to submit the ballot and complete your voting.

Task 4 After casting the vote, a confirmation email is received. The email contains a fingerprint of your ballot that can be checked against the election's bulletin board of cast votes. Verify your vote was correctly recorded.

5.2.5 Results from usability test

In this section the results of the questionnaire is presented. The data is presented in the form of a table followed by a discussion of the data and registered comments from the test participants. The result of the test election itself is not relevant, but a short introduction about the election turnout is included before the questionnaire results are presented.

Based on the bulletin board of cast votes, a total of 39 votes were cast and recorded in the systems database. In the invitation email I requested people to ask for help if something did not work and still answer the questionnaire about the system even if they had problems performing specific parts or processes. The reason the 2 last participants had not submitted their ballots were due to authentication problems, an error occured-

Statement	*5*	*4*	*3*	*2*	*1*	*Mean*	*Median*
The voting system was easy to use	20%	54%	10%	15%	2%	3,732	4
Instructions and guidance was easy to understand and follow	24%	41%	22%	10%	2%	3,76	4
The questions on the ballot were clear	46%	44%	7%	0%	2%	4,32	4
It was easy to mark my choices and submit the ballot	49%	41%	0%	7%	2%	4,27	4
I was easily able to correct my mistakes when marking options	22%	46%	22%	5%	5%	3,76	4
I felt the voting process took too much time	2%	5%	22%	56%	15%	2,24	2
I think the processes were to slow (i.e. encryption or verification)	2%	15%	27%	41%	15%	2,49	2
It was easy to verify that my ballot was correctly encrypted	22%	32%	20%	22%	5%	3,44	4
The process of authentication was easy and efficient	22%	54%	7%	15%	2%	3,78	4
I am confident my vote was recorded accurately	29%	44%	17%	5%	5%	3,88	4
I could easily verify that my vote was recorded correctly	20%	32%	34%	12%	2%	3,54	4
I trust the system will keep my vote secret	15%	34%	41%	7%	2%	3,51	3
There was sufficient information displayed on the pages	7%	59%	10%	20%	5%	3,44	4
I think I would need assistance to be able to use this voting site	2%	10%	5%	49%	34%	1,98	2
I found some of the steps very cumbersome	5%	20%	22%	51%	2%	2,73	2
I found the language in the system difficult to understand*	10%	17%	22%	44%	7%	2,78	2
In some steps, I did not understand what I was actually doing	15%	24%	10%	39%	12%	2,9	2

Table 5.3: Percentage results of answers from questionnaire. The values of 5 to 1 corresponding respectively to strongly agree, agree, indifferent, disagree and strongly disagree. *Too technical and difficult.

The statistical results of the questionnaire is displayed in table 5.3. The statements are graded from strongly agree to strongly disagree, corresponding to the numbers 5-1.

As shown in the table, 74% of the test voters found the voting system overall easy to use. Some of the test participants left negatively comments about the usability of the system. One comment was that the system was not very intuitive, so in a real election scenario some users would probably need a better guidance through the steps of the voting process.

About the instructions and guidance of the system, 65% stated they agreed (41%) or strongly agreed (24%) that it was easy to understand and follow. One of the test participants stated that the guidance was easy to understand, but that the buttons to guide through the web site and voting system could have had more describing (understandable) names. Terms like encrypting ballot and audit encryption is not intelligible by the general public, which is the targeted group for a voting system.

The statement about whether the ballot questions were clear, had a result of 90% agreement. The strongly disagreement of 1 participant (2%) might have been due to the english language of the questions or the formulation, rather than a usability factor of the system.

90% of the test participants stated it was easy to create a ballot for submission by marking choices for the questions. One of the disagreeing participants (7%) commented that he was suprised that he was not warned when he had only checked one party for the question were they were requested to choose two parties for a coalition government. He was surprised he was not given any error or was stopped when reviewing his question selection before encryption.

About correcting mistakes, 68% agreed or strongly agreed that it was easy to correct mistakes during the ballot preparation. In a ballot question were you are only allowed to check one option, you have to unmark a selected option if you want to change opinion and mark another option. A total of 3 test participants disagreeing/strongly disagreeing with the statement, commented this to be cumbersome with the de-selection. 22% remained indifferent to the statement, and since checking this correction possibility was not included as a task in the test procedure I assume not anyone had the need to correct ballot (also among those stating to agree or strongly agree).

71% disagreed (56%) or strongly disagreed (15%) that the voting process took too much time. Only 3 participants felt the process was too time consuming, and one comment was that this was due to the audit step, where the ballot has to be encrypted again after verifying encryption.This property is

also mentioned as strange in the first statement of the questionnaire (about how easy the voting system is to use), and the system should include an explaination to the voter of why this is necessary.

Wbout the speed and effectiveness of the actual encryption and verification processes, 56% did not disagreed with the processes being slow and a relatively large percentage (27%) were indifferent. One of the test participants in addition commented that in voting, as such a special case, encryption is of so high importance that the time of such a process does not matter. Regarding the one test participant (2%) answering he strongly agree the processes were to slow, the encryption of his ballot actually malfunctioned. He stated in the comment field that the encryption process froze at 0% complete. This test participant never asked for support so I can only guess he was using a web browser that is not supported by the Helios voting system.

The statement regarding that the process of auditing the encrypted ballot was easy, was barely agreed upon by the majority of the test participants (54%), and 20% remained indifferent. The "positively" results are probably due to the fact that the verifiaction is several steps to perform, but you are guided through the process with specific instructions, making it easier to carry out for the general user. The only comment made by the "agreeing" set of voters was that the process was easy to follow, but they did not understand the fact that they had to encrypt the ballot again after verification.

Several other comments were also stated around the confusion of why the ballot had to be encrypted again after verification, and also questioned the security of this process. The feedback for the voter/user is only that the election fingerprint matches ballot, the decrypted options of the ballot questions, and a feedback that encryption was verified and the proofs are ok. Some of the test participants disagreeing with the statement did not feel they got a proof of verification by getting this feedback, but this feedback can neither be too technical because there is no use to the general voter. One voter commented he did not understand anything of the verification step. In a real election scenario the technical system has to be created so it is trustworthy enough to the voter. He only needs a definite user friendly feedback that the encryption of his ballot was verified.

54% agreed and 22% strongly that the process of authentication was easy and efficient. One of the participants answering they strongly agree the process was easy and efficient, commented that a solution with "MinID" or "BankID" would be preferable, but that the solution with email and password is sufficient in this test voting scenario. Requesting passwords by email is a procedure "most people" is used to and probably why such a large percentage

is satisfied with this solution. One of the participants answering that he disagreed with the statement, comment that he had to ask for password multiple times and think the e-mail service had bugs. This may be due to the factors I mentioned regarding the test election outcome. Another user disagreeing with the statement, comments that it took to long to receive the password by email. Using a method like this can have delays, and if using the system for actual election other methods of authentication is desirable. The one person strongly disagreeing might be the other test participants which encountered authentication problems, since the majority thought the authentication process was sufficiently effective for its purpose.

To the statement "I could easily verify that my vote was recorded correctly", the majority of the test participants were indifferent (34%). The reason why so few disagreen to the statement was probably because the process of controlling the received ballot fingerprint against the fingerprint displayed on the bulletin board of cast vote is easy to carry out, also by the general voter. The large amount of indifferent statements can correspond to the fact that the comparison of a fingerprint does not really tell the general voter anything, even though you can control in the system that the fingerprints are the same a normal voter can not verify that these fingerprints correspond to what he really voted. The set of voter's that disagreed with this statement probably also includes voters with more technical insight that don't think a bulletin board generated by the system and displayed can serve as a proof that the vote was recorded correctly.

Another property commented regarding the usability of this verification process, was that the link to the bulletin board is located on the "home page/start page" of the election. Since a confirmation email is received shortly after the vote is cast, the link to this feature should have been displayed on the final page of the voting process, instead of having to close the window and go back to the start page.

The next statement that was included in the survey was regarding whether the users trusted that the voting system kept their votes secret, so no one can learn what candidate or party they are voting for. This was the statement in the survey to which the most user remained indifferent (41%), while only 9% stated they disagreed (7%) or strongly disagreed (2%) as disaplayed in figure 5.4 The general voter is not familiar with the technical concepts of cryptography and have to trust on the construction of the system, and this is probably why so many remained indifferent to this statement when not knowing any additonal information regarding the underlying technical functions. One of the indifferent test voters commented that the encryption seems fine but he

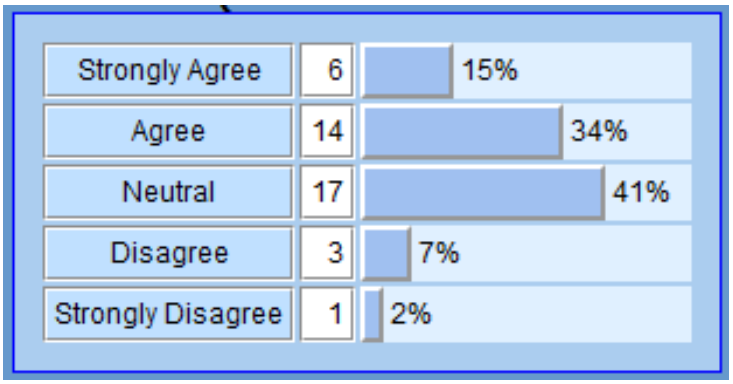

Figure 5.4: Result of statement: "I trust the system will keep my vote secret"

wouldn't know, and another test participant commented that there was no information regarding how the vote is encrypted, like what method used. The general voter would probably don't have a enough knowledge about the field to judge the safety of the process, but the test participants that were interested would prefer more information. Some of the disagreeing parts left comments about that they knew to little about the system to trust it.

The majority of the test participants agreed (59%) or strongly agreed (7%) there was sufficient information displayed on the web sites and during the steps of voting. Some of the test participants even commented that there was too much information displayed. One specified that there was too much technical information displayed, and that a voter not familiar with the meaning of the terms of a *fingerprint hash* or other extra information could be confused. Another user requested more information in clear text of the different steps during the voting process and what they mean, for instance what was happening when encrypting or when auditing.

The negative comments were again regarding no information about the fact that you had to encrypt your ballot over again after verification, and no information about login mechanism or other procedures (I assume the user means the processes of encryption and verification). Some of this, as mentioned before, is a request for more technical details and the system should maybe provide measures for voters interested in more in-depth information.

Since all the test participants that answered the questionnaire were able to record a vote (except the two users with authentication problems), this gave a certain idea of the help needed to cast a vote with the system. They followed the tasks given but did not need any further guidance. From the questionnaire 83% answered they disagreed (49%) or strongly disagreed (34%) that

they would need assistance to use the voting site. One user strongly agreed to the need of assistance, but that was due to the fact that the voting system did not support the browser he was using, and no processes functioned properly but when used in an election a help desk or support would probably be available.

Only a total of five test participants requested me for help during the test period. Two questions were regarding a problem with authentication, which I could fix by adding their email properly to the voting list. Two other questions were regarding problems of encryption process not proceeding as it should, it just stayed at 0% complete, but this was probably due to the use of browsers that the system did not support (these two users were using Opera and Chrome). The last user requested help regarding the verification of the recorded ballot on the bulletin board. As mentioned earlier the user was confused because he had to close the window and go back to the start page to find a link to the bulletin board.

About cumbersome steps in the system, some users commented they missed certain features or short cuts in the system to improve some of the steps. Again, also the fact with having to close the window and go back to start page after submitting vote instead of including a further directing link, was mentioned as cumbersome. Both the voters stating they strongly agreed (5%) and some of the voters agreeing (20%) there was cumbersome steps, were aiming at the process of auditing the ballot after encryption. There was again comments about the unefficient copy-paste method, and also as mentioned earlier about the extra step of having to encrypt the ballot again after verification. The reason for this "criticism" about the auditing process was probably because it was a relatively cumbersome step compared to the process of just casting a ballot, because you had to navigate away and copy and paste information to a ballot verifier. Also, the process were not understood by many of the test users and therefore seemed like a slightly unnecessary step.

The last statement in the questionnaire was regarding the understanding of the steps of the voting process and the other processes that comes with it. This statement was the statement with the most sporadic answers. We have already stated that a lot of the technical processes were not very comprehensible by the general voter. A total of 39% stated they agreed/strongly agreed with the fact that they in some of the steps did not understand what they were actually doing. Several of these feedbacks were related to the auditing process, "what does the numbers of the auditing process means", "I did not understand what happened when encrypted", etc.

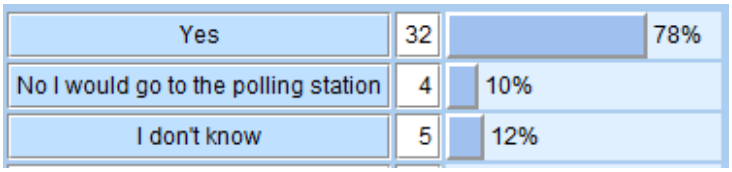

Figure 5.5: The resulting answers to the question:"If possible, would you vote from your home computer in the next Government election?"

At the end of the statements in the questionnaire, a question were added to see how the voter thought of using the mean of Internet voting in a real life scenario. The question and resulting answers are showed in figure 5.5, and shows that a total of 78% would vote over the Internet. The user group of my test are already using their computers for on a everyday basis, and voting from home would be easier than going to a polling station. The question did not specificly give any facts or requirements of the voting system to be used, but just read "if possible", and we assume a adequate system is developed. Some users answering yes, left comments to emphasise this was only if a safe and sound system was in use, and not using this system just evaluated.

One of the users that answered he would go to the polling station, specified in the comment field that this was only due to his tradition, and had noting to do with what kind of voting system deployed. Another user stated that voting should be confined to polling stations because of privacy issues and the coercion problem. A third user specified he did not trust electronic voting regardless of how it is created and who is the trusted parties behind it, but that he with more detailed information would consider it. Since I in this test scenario did not give the test users any background information about the system, the user had nothing to base his trust on. In a real life deployment of a voting system, the system had to carefully described and illustrated, as well as fully transparent for third party analysis.

5.2.6 Related work

While Helios is one of the most complete implementations of an open audit election, there has only been one other published study of its end user usability, carried out at the University of Waterloo [7]. The goal of this test was to "determine the target voter's ability to accurately record and verify their intended vote" [7].

A mock student election was created, and the evaluation was based on the

think aloud protocol with audio recorded during the voting sessions, and a short exit survey of opinions of the test system. 20 test voters in the age of 18-23 participated, all claiming to be very comfortable with new technology when questioned, and 8 of the participants had actually recently voted online in an election for the University's student government.

The Waterloo test resulted highly in favor of the voting system, with the benefits of convenience, accuracy and efficiency. Most of the test voters were confident their vote was accurately recorded and counted, and kept secret. A comparison of two of the same survey questions, displayed as percentage in figure 5.6, shows that my test participants probably were more critical to these statements. Participants in the Waterloo test trusted the system to keep their vote secret slightly more than they trusted the system to accurately count their vote (the opposite to my test).

A comparison of all the similar survey question results from the two tests are presented in table 5.4.

	Waterloo test		My test	
	Mean	Median	Mean	Median
The voting system was easy to use	3.75	4	3.73	4
I felt the voting process took to much time	2.4	2	2.24	2
The ballots were clear	3.9	4	4.32	4
I was able to easily correct my mistakes	3.95	4	3.76	4
I am confident my vote was recorded accurately	4.4	4.5	3.88	4
I trust the system will keep my vote secret	4.55	5	3.51	3

Table 5.4: Comparison of statements in the Waterloo test and the test I performed. (By calculated mean and median)

Aside from the different result in the two questions about recording and secrecy, showed in figure 5.6, and some authentication issues, the feedback from the test emphasized my results. As showed in table 5.4 the median of the statement trusting the system's privacy differs, but the rest of the results are very similar.

The users also agreed on easy ballot preparation, time efficient encryption, the technical language, little or no understanding of the encryption process, and many of the users specifically asked to audit the encryption were confused and defeated by the large page of ciphertext and lack of instructions. Only 2 were able to verify their ballot using the ballot verifier, but did neither understand the feedback. They agreed this technical means overall made the system offer a good sense of security improving peoples trust.

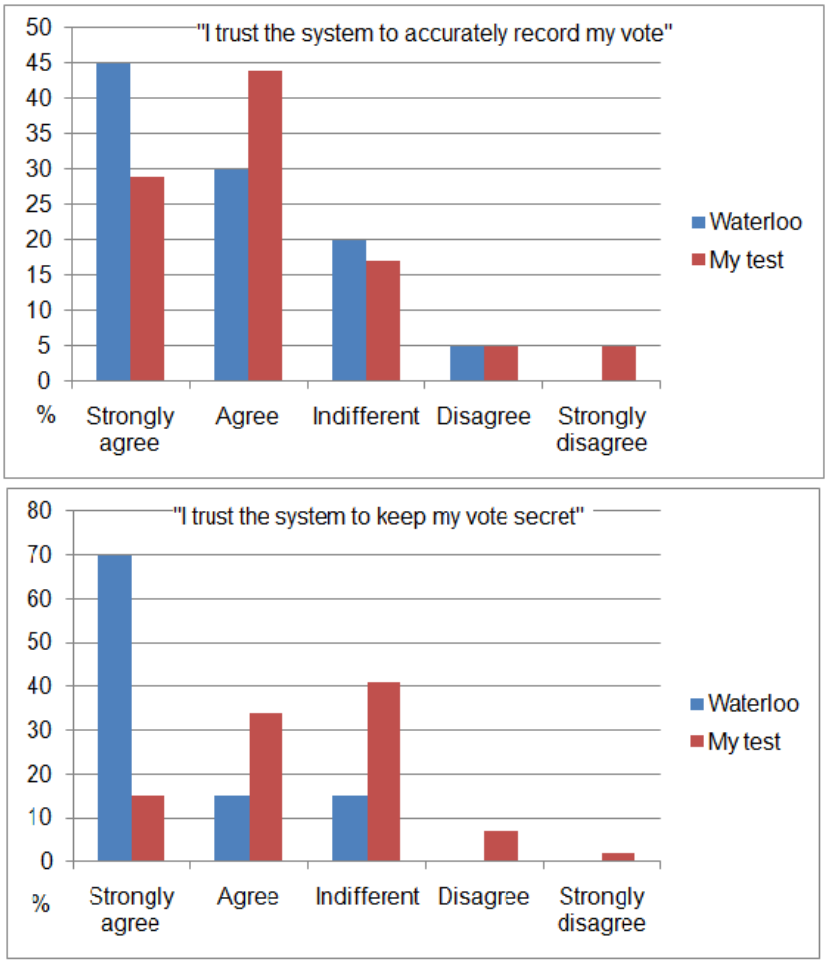

Figure 5.6: Graphical result of two of the statements from the tests.

Both the findings of the encryption and audit step are similar in my test and the Waterloo test. The Waterloo test was based partly on the think aloud protocol, while my test results are based on user feedbacks, and I have to assume that the test participants went through the task of auditing ballot.

The test participants of my test were, with exceptions, young people used to the computer tool, and some also with technical background, and these could supply interesting critical views of the system and the test process. In the Waterloo test, the test was carried out among some of the university's students and several of these participants had actually voted online before, which I assume was not so common among my test participants. But despite of some different test participants with more critical views maybe, the results turned out quite similar.

A last, but concerning fact to mentioned is that more than half of the participants (11 of 20) in the Waterloo test would not have completed their vote and submitted their ballot, had this been a real election scenario.

5.2.7 Summary

The results of the usability test were in favor of the voting system, the overall use of the voting system was stated as easy. The test voters were mainly positive to the voting system, however there were pointed out several less user-friendly properties, and concerns with encryption and verification. A lot of the results in this usability test emphasized points made by the test at Waterloo University [7].

There were found several non user-friendly aspects of the system, some of them being:

- The correction of options in ballot preparation (de-selection)
- Confusing encryption process
- Bad error feedback (especially when encryption halts)
- Poorly explanation of the audit mechanism and why to audit (just a button saying "Audit")
- Cumbersome to follow link to separate window for ballot verifier
- Poorly information of verification results

The system could clearly have introduced some more explanation or at least a way of obtaining info on several of the steps in voting process. Confused


Dear Test Participant,

Election URL: http://www.heliosvoting.org/elections/agxoZWxpb3N2b3RpbmdyEAsSCEVsZWN0aW9uGJ-0AQw/view
Direct Voting URL: http://www.heliosvoting.org/elections/agxoZWxpb3N2b3RpbmdyEAsSCEVsZWN0aW9uGJ-0AQw/vote
Election Fingerprint: skwlfgbYwUePatmjCeWMojV/QpQ

Your email address: jannaweber@gmail.com
Your password: cPwoh1qlda

--
VoterStudy@gmail.com
via the Helios Voting System
www.heliosvoting.org

Figure 5.7: Invitation email for a created election, generated by Helios

participants were often looking for more instructions or reasonings behind each step of the system, and it could be useful to add a help functionality, FAQs or better instructions through the processes. In a larger election I assume a help desk would be available during an election period for users in need of assistance.

The system gives no information for instance about what it is or why the voter should audit his ballot, and several participants commented they would just ignore this button and submit the ballot. This mean, even when having a test participant understanding encryption they seemed confused by the verification step, because they can see how their selections were encrypted, but the explanation is vague and it does not mean a whole lot to the voters.

The encryption process seems to be a step of confusion for the general voter and could as well be hidden, while the more technical interested voter would like to get more details about the process. The solution could be to name it something more user friendly for the public or display it graphically, while giving the interested voter a way to obtain more information. Even though the steps of privacy or verification not necessarily makes sense to the voter, it is stated that the more technical details give a greater degree of confidence in the security of the system.

The authentication step was not of big confusion in this test. I had all the test participants request their passwords upon authentication, and then cast their ballot using their email and password. The content of the generated email is shown in figure 5.7, and could lead to confusion with all the extra information.

The error message showed in figure 5.8 was displayed when the ballot information was not corresponding to the encrypted ballot, trying to auditing encryption. The error feedbacks in the system now has to be improved, both regarding encryption (what if it halts at 0%?) and verification. Clear er-

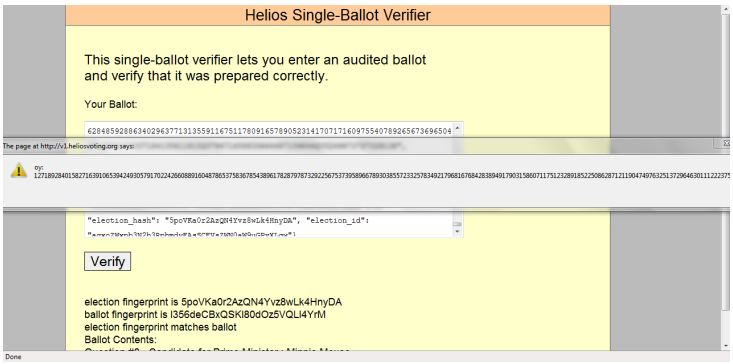

Figure 5.8: Error message appearing when removing some details from field of ballot information."

ror feedback is needed to make sure the voter does not give up the casting because of a mistake or an error, like in the Waterloo scenario.

For this use of trying out a test election, and not actually being a high risk election, the users were mostly positive, but had this been a real election scenario I would assume the voters would have been be a lot more critical to the system. Even though the participants were concerned with security issues, they felt that the benefits of electronic voting outweighed the risks.

Both tests emphasized the fact that it is an issue regarding if the target voters understand the subtle difference of having an open-audit voting system, or if voters would be able to use a system like this to its full potential in an real election scenario. Means to encourage more of the voters to actually audit their ballot are needed. If the voters are not helped to understand why such an open-audit system is different, then the goal of the system is also lost.

Chapter 6

Discussion and Conclusion

In this chapter we have discussed about the two modern voting solution, and how they are based on the same basic ideas. Some issues with end-to-end verifiable systems are discussed, including the tension between

6.1 Two modern voting systems

The Helios voting system and the Norwegian "E-Vote 2011" are two very different voting systems, created for different types of elections. Helios is mainly a system created for low-coercion elections and to teach about the benefits of open-audit voting systems. This system can be adapted to fit elections of different organizations, and have already been deployed in a few elections [52, 79, 77].

The Norwegian e-voting system is a political election system created by the Government. This system is created according to the Norwegian electoral system, to improve the process of political elections. While Helios is created to be as simple as possible for understanding and verification purposes, a political voting system is a complex scheme, with interactions between many components and processes.

Helios is a system for Internet voting only, while the Norwegian system are developed to mainly be an additional offer to the traditional voting scenario, and have to interact with other means of casting ballots. The mean of casting multiple electronic ballots is not sufficient for the threat of coercion in a Governmental election, so the possibility of casting an over-writing paper ballot at a polling station would still be available. Helios is an Internet voting

system only, and is running on Google App engine or possibly on a organization's server. The Norwegian system, requires its own infrastructure with secure hardware modules, ensuring providing a secure basis, and reliability and robustness for the system.

Helios is focusing on providing integrity throughout the entire election, while privacy is left second with weaker guarantees. Privacy is also important, but both mechanisms to ensure privacy as well as authentication mechanisms in Helios is definitely less developed than means to verify integrity.

The Norwegian system requires strong focus on integrity, but privacy and accuracy are of equal importance. The system has a strong authentication mechanism, avoiding voter impersonation, and security failures affecting the election outcome or compromise the voters' privacy are not acceptable.

A political election system, obviously has stronger requirements related to secrecy, accuracy, reliability etc, and such systems are also targets of more attacks. In addition the system has to deal with the threat of coercion, something Helios does not take into consideration. Helios are vulnerable to attacks like ballot stuffing, voter impersonation and if the Helios itself being corrupt the ballots of all voters could be revealed. These are threats the Norwegian system counters with strong authentication mechanisms and integrity verified throughout the entire election process, never letting the content of a ballot be associated with a specific voter. The Norwegian system communicating over secure authenticated channels and ballot box ensures only one vote is counted per voter, making ballot stuffing "impossible".

These are just some of the differences of the two voting solutions, but we understand they are significantly different. But one thing the two systems clearly have in common is the fact that they are both cryptographic voting systems casting ballots over the Internet, and they are both universally verifiable voting systems.

So, despite these differences there are several similarities to point out. Some of these similarities and how they are provided are listed in table 6.1. The Norwegian e-voting system is based on much of the same technology and methods as the Helios system.

Both systems are based on the El-Gamal cryptosystem, and anonymization mechanisms to provide voter secrecy. The "E-vote 2011" is planning to use the mean of a verifiable mix-net generating proofs of operation, while Helios has changed to homomorphic tallying to further simplify the mechanisms of verification. Helios, aiming at simple elections, can efficiently use homomorphic tallying, but for Norwegian elections or elections of even larger

	"E-Vote 2011"	*Helios*
Ballot options	Options and write ins	Mark options
Encryption	El-Gamal	El-Gamal
Verification of processes	Cryptographic proofs	Cryptographic proofs
Authentication mechanism	eID (MinID)	username/password
Voter confirmation	SMS with receipt code	E-mail with ballot fingerprint
Decryption and Tallying	Mix Net decryption Threshold decryption	Mix-Net → Homomorphic Tallying Threshold decryption

Table 6.1: Some similarities of the two voting systems

scale, this becomes unwieldy maybe impossible [1]. Both systems uses cryptographic proofs for verification and support the extra security feature of multiple trustees. Both systems depends on the voter auditing his cast vote to protect against compromised computers tampering with ballots.

For a low-coercion the advantages of an efficient voting process may outweigh the risks, but in a political election scenario a greater risk is involved. Helios is like a basis that can be adapted for elections, while a Governmental election system has to be created as a specific project.

6.2 Issues with E2E systems

There is a lot of research and development going on in the field of Internet voting, but not many countries have yet implemented such systems. Internet voting is a criticized and complex topic, and there is clearly issues with such election systems. The possibility of voting in a governmental election via Internet, is a desirable functionality, but with the use of Internet follows also new threats. The need for improved security mechanisms led to the development of E2E verifiable voting systems.

E2E systems are fantastic at detecting fraud. Such systems can provide greater assurance that the election outcome is correct than traditional systems. There is no need to trust a computer or any election officials for integrity of election outcome. But with the advantages of E2E new issues arise, and sacrifices have to be made. When introducing E2E verifiability, there is an increased potential for voter privacy violations [4], issues with coercion, complexity and usability arise.

Focusing on attempts to provide verifiable information for auditing purposes,

there might be some weak points of protecting privacy. What if somehow information about votes leak? While any voting protocol for a precinct voting system, for instance an electronic voting booth with a DRE machine, has to provide measures to be both coercion-resistant and voter verifiable, a remote Internet voting system would never be 100% coercion resistant [1]. Introduced mechanisms for verification, giving the voter a possibility of veriy who i voted for could potentially lead to the voter being coerced to vote a certain way. It is probably impossible to achieve both true voter verifiability and coercion-resistance. [1]

The introduction of universal voter verifiability also requires more system components (for instance a receipt generator), and more communication between client and server. Election systems are complex systems, and when then changing to use the Internet, the complexity of these Internet voting systems are a drawback itself. The complexity of these election systems makes them more vulnerable [82]. A larger number of elements, provides the attacker with more exploitation possibilities. All levels of the system can be attacked, from the browser, at cryptographic level or hardware level.

There definitely is a tension between verifiability and usability. Verification in Internet voting systems requires more steps for the voter to carry out in the voting process, and involves cryptographic mechanisms. It is a challenge to provide these means of technical verification possibilities, and still provide a user-friendly voting system. The survey of the usability test carried out in chapter 5.2, emphasizes this tension between verifiability and usability. The ordinary voter needs extra guidance to understand the steps of verification and especially the results. With remote voting systems there is a strong requirement to provide intuitive systems. There is no help available, other than information given (and a help desk?), to guide the voter to a voting process. This requires an good user interface, and especially for disabled voters.

The success of a Governmental voting system is strongly based on the public's trust in the system. The introduction of verifiable elections, were the voter himself or trusted experts can audit the system, increase the trust and acceptance of these systems. But the other way around the mean of verification could also be used to create doubt. What if a person, or a group or people get some other fellow conspirators to get their votes tampered with. And then the person, or people, show their receipts, which now would be erroneous? This could create doubt and bring the election or system into infamy.

6.3 Conclusion

The area of electronic voting is an enormous field. Electronic voting has developed since the early 1900s. Several techniques, with their advantages and drawbacks has evolved. Electronic means of counting votes are widely used but the mean of casting electronic ballots keeps the discussion going.

There has been issues with all kinds of voting technologies; erroneous punch cards systems and criticized DRE voting machines. The latest development of Internet voting, are a topic of great concern.

In chapter 2.4 concerns about Internet voting system were mentioned. It was mention that the advantages and benefits of such systems, like the efficiency and accessibility, could be outweighed by the issues of the many potential security threats. There are critical security threats to electronic voting systems, but developed secure cryptographic methods can provide secrecy, integrity and proofs of correctness, countering many of these potential attacks. Other will argue the benefits of electronic voting outweigh the risk.

It is impossible to create a generic voting system for political elections. Countries are different, have different elections with different ballots and this requires different systems. Each election is a significant project on its own, and systems have to be created and adapted to the specific country's electoral system.

The Helios system offers a good scheme for low-coercion elections without the highest stakes at risk, introducing people to the possibilities of open-audit systems. This is a scheme easily adaptable to different organizational elections. The usability test and survey performed concludes that the system offered a good sense of security, improving the voter's trust in the secrecy and accuracy properties of the system. The results also emphasize the tension between verifiability and usability.

The Norwegian voting system created for political elections, deploys strong cryptographic mechanisms for secrecy, integrity and verification providing a promising scheme for Internet voting. To deploy Internet voting schemes sacrifices has to be made. Even if the Norwegian voting system has the strongest cryptographic mechanisms it cannot protect against coercion attacks, affecting the democratic purpose of elections. An Internet voting system will never be 100 % secure. The importance of sacrifices has to be considered, like trading verifiability against the potential exposion to coercion.

References

[1] Kristian Gjosteen. Analysis of an internet voting protocol. Technical report, NTNU, March 2010.

[2] Mary Bellis. The history of voting machines. *About.com:Inventors*, 2000.

[3] Website of federal voting assistance program, us. `http://www.fvap.gov/`.

[4] Ronald L. Rivest. Perspectives on end-to-end voting systems. NIST E2E Workshop, October 13, 2009.

[5] David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 1981.

[6] C.A. Neff. Practical high certainty intent verification for encrypted votes. *VoteHere document*, 2004.

[7] J.L. Weber and U. Hengartner. Usability study of the open audit voting system helios. *INCLUDE*, 2009.

[8] Ghassan Z. Qadah and Rani Taha. Electronic voting systems: Requirements, design and implementation. *Computer Standards & Interfaces*, March 2007.

[9] IACR website of test election. `http://www.iacr.org/elections/evoting`.

[10] `http://blog.heliosvoting.org/`.

[11] Estonian National Electoral Committee. Internet voting in estonia. `http://www.vvk.ee/public/dok/Internet\_Voting\_in\_Estonia.pdf`, 2008.

[12] "e-valg 2011" website. `http://www.regjeringen.no/nb/dep/krd/prosjekter/E-valg-2011-prosjektet.html?id=597658`.

[13] W. Aspray et al. *Computing before computers (Chapter 4)*. Iowa State University Press, 1990.

[14] Website of Museum of Computer and Communication Technology. Punc card computing. `http://www.technikum29.de/en/computer/punchcard.shtm`, 2003-2010.

[15] `http://www.math-cs.gordon.edu/courses/cs323/FORTRAN/fortran.html`.

[16] US Federal Election Commission. Punchcards. `http://www.fec.gov/pages/punchrd.htm`.

[17] Stuart Gorin. Presidental race statistical tie in swing state florida. *America.gov*, 2008.

[18] Randolp C. Hite. *Elections: electronic voting offers opportunities and presents challenges*. United States General Accounting Office, 2004.

[19] `http://www.cityofeastlansing.com/Home/Departments/CityClerk/ElectionInformation/VotingEquipment/`.

[20] Kathy Gill. An illustrated history of voting methods and systems. `http://uspolitics.about.com/od/elections/ig/History-of-Voting-Systems`, About.com.

[21] Michael Shamos. Optical scan systems. Lectures, Institute for Software Research International Carnegie Mellon University, 2004.

[22] `http://www.sequoiavote.com/`, Last visited Feb 2010.

[23] Matt Blaze, Arel Cordero, Sophie Engle, Chris Karlof, Naveen Sastry, Micah Sherr, Till Stegers, and Ka-Ping Yee. Source code review of the sequoia voting system. Technical report, University of California, Berkeley, July 2007.

[24] e-voting systems. `http://www.tiresias.org/research/guidelines/e\_voting.htm`.

[25] `http://www.verifiedvoting.org/article.php?id=5018\#s1q1`.

[26] US Federal Election Commission. Direct recording electronic. `http://www.fec.gov/pages/dre.htm`.

[27] `http://whatis.techtarget.com/glossary/e-voting-glossary.html`.

[28] California Internet Voting Task Force. A report on the feasibility of internet voting. http://www.sos.ca.gov/elections/ivote/final_report.htm\#final-3, 2000.

[29] The working committee. E-voting: Challenges and opportunites. *"E-valg 2011" website: http://www.regjeringen.no/en/dep/krd/kampanjer/election_portal/electronic-voting.html?id=437385*, 2006.

[30] Kevin Bonsor and Jonathan Strickland. How e-voting works: Voting over the internet. *How stuff works.com*, March 2007.

[31] Kristian Gjøsteen. Analyse av kryptoprotokoller for e-valg. presentation, 2009.

[32] Program and presentations at nist e2e workshop. http://csrc.nist.gov/groups/ST/e2evoting/program_E2E.html.

[33] Website of pret-a-porter. http://www.pretavoter.com/.

[34] Website of punchscan. http://www.punchscan.org/.

[35] Website of scantegrity. http://www.scantegrity.org/.

[36] Ben Adida and C. Andrew Neff. Ballot casting assurance. *EVT 06 / Usenix*, 2006.

[37] Svetlana Z. Lowry and Poorvi L. Vora. Desirable properties of voting systems. NIST E2E workshop, October 13, 2009.

[38] Internet policy institute. Report of the national workshop on internet voting: Issues and research agenda. *Internet policy institute*, 2009.

[39] Svetlana Z. Lowry and Poorvi L. Vora. Desirable properties of voting systems. presentation, E2E workshop NIST, 2009.

[40] http://www.unc.edu/courses/2008spring/law/357c/001/onlinevotingsite/technology.html.

[41] Ben Adida. Helios: Web-based open-audit voting. *17th USENIX Security Symposium*, 2008.

[42] Guido Schryen. How security problems can compromise remote internet voting systems. *Electronic Voting in Europe*, 2004.

[43] Avi Rubin. Security considerations for remote electronic voting over the internet. AT&T Labs, http://avirubin.com/e-voting.security.pdf.

[44] W. Stallings. *Cryptography and network security.* Prentice Hall, 2010.

[45] D. Richard Kuhn, Vincent C. Hu, W. Timothy Polk, and Shu-Jen Chang. Introduction to public key encryption and the federal pki infrastructure. NIST, 2001.

[46] FIPS PUB. Secure hash standard (shs). Technical report, NIST, 1995.

[47] RSA Laboratories. Public key cryptography standards, chapter 2.1.1. public key cryptography. `http://www.rsa.com/rsalabs/`, 2009.

[48] `http://www.sun.com/blueprints/0801/publickey.pdf`.

[49] INCLUDE. Itu-t recommendation x.509. Technical report, International Telecommunications Union, 2005.

[50] `http://msdn.microsoft.com/en-us/library/ff649801.aspx`.

[51] Taher ElGamal. A public-key cryptosystem and a signature scheme based on discrete logarithm. *Advances in Cryptology*, 1985.

[52] Ben Adida, Olivier de Marneffe, Olivier Pereira, and Jean-Jaques Quisquater. Electing a university president using open-audit voting: Analysis of real-world use of helios. *Electronic Voting Technology/Workshop on Trustworthy Elections*, 2009.

[53] `http://courses.csail.mit.edu/6.897/spring04/L18.pdf,L19.pdf,L20.pdf`.

[54] `http://dsns.csie.nctu.edu.tw/research/crypto/HTML/PDF/E95/393.PDF`.

[55] Mikhail J. Atallah and Marina Blanton. *Algorithm Design and Analysis Techniques and Applications.* Chapman and Hall, 2009.

[56] Jim Dray. Advanced authentication technology. NIST, Computer Systems Laboratory Bulletin, 1991.

[57] `http://www.smartcardbasics.com`.

[58] `http://www.smartcardalliance.org/`.

[59] Website of bankid. `http://www.bankid.no`.

[60] Kjell Hole, Thomas Tjstheim, Vebjrn Moen, Lars-Helge Netland, and Yngve Espelid. Next generation internet banking in norway. Technical report, University of Bergen, 2008.

[61] Difi lanserer minid og id porten. http://www.difi.no/artikkel/2009/11/difi-lanserer-minid-3.0-og-id-porten, 2009.

[62] Difi. Implementation guide for federation with minid. Technical report, Difi, 2009.

[63] http://www.novinite.com/view_news.php?id=112540.

[64] E vote 2011 Ministry of Local Government and Regional Development. E-vote 2011: System requirements specification. Technical report, Ministry of Local Government and Regional Development, 2009.

[65] http://www.evalgbloggen.no.

[66] Website of scytl. http://www.scytl.com.

[67] ErgoGroup/Project E-vote. Project e-vote 2011: Contractor solution specification. "E-vote 2011" web site, 2009.

[68] Representanter. Representantforslag fra stortingsrepresentantene erna solberg, michael tetzschner og trond helleland om stanse forsk med avgi elektronisk stemme utenfor valglokale. http://www.stortinget.no/no/Saker-og-publikasjoner/Publikasjoner/Representantforslag/2009-2010/dok8-200910-128/1/, 2010.

[69] Website of ben adida. http://adida.net/.

[70] http://code.google.com/appengine/docs/whatisgoogleappengine.html.

[71] Helios v3 verification specs. http://documentation.heliosvoting.org/verification-specs/helios-v3-verification-specs.

[72] Dan Morrell. Election by encryption - secret ballots, verifiable votes. *Harvard Magazine*, May2010.

[73] Josh Benaloh. Simple verifable elections. *Accurate Electronic Voting Technology Workshop 2006 on Electronic Voting Technology Workshop*, 2006.

[74] Ben Adida. Verifying elections with cryptography. http://www.youtube.com/watch?v=ZDnShu5V99s, 2007.

[75] Josh Benaloh. Ballot casting assurance via voter-initiated poll station auditing. *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology*, 2007.

[76] oauth 1.0. protocol. http://oauth.net/core/1.0a, 2009.

[77] (https://iacr-helios.appspot.com/helios/elections/3a40d55c-1396-11df-bd14-19dba45a8649/view).

[78] Stuart Haber, Josh Benaloh, and Shai Halevi. The helios e-voting demo for the iacr. *IACR*, May 2010.

[79] http://usg.princeton.edu/officers/elections-center/election-reform-blog.html.

[80] Helios election monitor. Website UCL Crypto group, `http://www.uclouvain.be/crypto/electionmonitor`.

[81] Kwiksurveys - online survey. `http://www.kwiksurveys.com`.

[82] Leif Martin Kirknes. "e-vote" is not safe. *Computer World*, 2009.