

TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology
Department of Informatics
Chair of Information Security

Practical Security Analysis of E-voting Systems

Master thesis

Student: Triinu Mägi
Student code: 030282 LAP M
Supervisor: Prof. Ahto Buldas

Tallinn
2007

Declaration

Hereby I declare that this master thesis, my original investigation and achievement, submitted for the master degree at Tallinn University of Technology has not been submitted before for any degree or examination.

.....
(Date)

.....
(Author's signature)

Table of contents

Introduction.....	9
1. Concept of e-voting.....	12
1.1. E-voting terms.....	12
1.2. Security properties of e-voting.....	12
2. State of the art.....	13
3. Descriptions of e-voting systems.....	16
3.1. General description of e-voting systems.....	16
3.2. Estonian e-voting system.....	20
3.3. SERVE e-voting system.....	25
3.4. Differences between two systems.....	29
4. Security analysis.....	31
4.1. Analysis method.....	31
4.2. E-voting models for SERVE and the Estonian e-voting system.....	34
4.3. Adversary model and threats.....	44
4.4. Security assumptions and justifications.....	47
4.5. Security analysis in the modeled environment.....	49
4.5.1. The security analysis of the Estonian e-voting system.....	49
4.5.2. The security analysis of the SERVE system.....	55
4.6. Attack game risk analysis in the modeled environment.....	61
4.6.1. General characteristic for the environment.....	61
4.6.2. Environment's characteristics.....	62
4.6.3. Attack game risk analysis.....	65
4.6.3.1. The justification of Assumption IV.....	65
4.6.3.2. The justification of Assumption IX.....	66
4.6.3.3. Attack against voting privacy: Attack tree analysis.....	66
4.6.3.4. Large-scale votes' buying: Attack tree analysis.....	71
4.6.3.5. Large-scale votes' theft: attack tree analysis.....	74
4.6.3.6. Large-scale disfranchisement before receiving votes.....	75
4.6.3.7. Large-scale disfranchisement of votes in two servers.....	76
4.6.3.8. Large-scale modification of ballots in the connection between Voter Application and Network Server of SERVE.....	76
4.6.3.9. Control over processes of Votes Storing Server of SERVE.....	77
4.6.3.10. Large-scale votes' adding in Votes Counting Server of SERVE.....	78
4.6.3.11. Attack tree analysis with alternative environment characteristics....	79
Summary.....	83
Kokkuvõte.....	84
Appendix 1.....	86
Appendix 2.....	88
Appendix 3.....	89
Appendix 4.....	90
Appendix 5.....	90
Appendix 6.....	91
Appendix 7.....	92
Appendix 8.....	93

Appendix 9.....	93
Appendix 10.....	94
Appendix 11.....	95
Appendix 12.....	97
Appendix 13.....	98
Appendix 14.....	99
Appendix 15.....	100
Appendix 16.....	100
Appendix 17.....	101
Appendix 18.....	102
Appendix 19.....	103
Appendix 20.....	104
References.....	105

Figures

Figure 1. Phases of e-voting.....	16
Figure 2. The components of e-voting.....	17
Figure 3. The description of e-voting.	19
Figure 4. The components of the Estonian e-voting system.....	21
Figure 5. The description of the Estonian e-voting system.	24
Figure 6. The infrastructure of the SERVE project.	26
Figure 7. The description of the SERVE system.	28
Figure 8. Attack Tree.	32
Figure 9. Diagram of the attack game from the attacker’s point of view.	33
Figure 10. Voter Application of the Estonian e-voting system.	36
Figure 11. Voter Application of SERVE.	36
Figure 12. The processes of Network Server of the Estonian e-voting system.	38
Figure 13. The processes of Network Server of SERVE.....	39
Figure 14. The processes of Votes Storing Server of the Estonian e-voting system.	41
Figure 15. The processes of Votes Storing Server of the SERVE e-voting project.	42
Figure 16. The processes of Votes Counting Server of the Estonian e-voting system.	43
Figure 17. The processes of Votes Counting Server of SERVE.....	43
Figure 18. Possible ways for large-scale votes’ theft in the Estonian e-voting system. ...	50
Figure 19. Possible ways of disfranchisement in the Estonian e-voting system.	52
Figure 20. Possible ways of votes’ theft in SERVE.	56
Figure 21. Possible ways of disfranchisement in SERVE.	58

Tables

Table 1. The function of general e-voting model.	18
Table 2. Data items and their abridgements of general e-voting model.	18
Table 3. The functions of the Estonian e-voting system.....	23
Table 4. The data items of the functions of the Estonian e-voting system.	23
Table 5. The functions of the SERVE system.	27
Table 6. The data items of the functions of SERVE.....	27
Table 7. The differences of the two e-voting systems.	29

Abstract

We adapt game theoretic methods for studying the security of two e-voting systems: the Estonian e-voting system and Secure Electronic Registration and Voting Experiment (SERVE) performed in the United States of America. Our security analysis does not give absolute security proofs; it analyzes practical security against large-scale attacks performed by rationally thinking attackers. We define a model for describing the real life in environment in which voting takes place and analyze the behavior of rational adversaries based on game theory. Some of the assumptions in the model are justified by using multi-parameter attack trees. We show that in our model the Estonian system is secure while the American one is not. We tried to choose the parameters of the model as close as possible to the real world characteristics. The reliability of the results is still somewhat questionable because of our limited knowledge about many of these parameters. For having some more justifications we analyze the robustness of our choices of parameters and show that our main results do not change, if the parameters are reasonably modified.

Annotatsioon

Käesolevas töös kohaldame mänguteoreetilisi meetodeid analüüsimaks e-valimiste süsteemi turvalisust. Vaatluse all on Eesti e-hääletamise süsteem ja Ameerika e-valimiste projekt Secure Electronic Registration and Voting Experiment (SERVE). Meie turvaanalüüs ei käsitle mitte absoluutset turvalisust, vaid praktilist turvalisust ratsionaalse ründaja ulatuslike tagajärgedega rünnete vastu. Töös defineerime keskkonnamudeli, kirjeldamaks valimiste reaalsel keskkonda ja analüüsime mänguteooria abil ratsionaalse vastase käitumismudelit. Kasutades mitme parameetriga ründepuid, analüüsime keskkonnamudeli neid eeldusi, mis vajavad empiirilist põhjendust. Töö tulemusena näitame, et meie keskkonnamudel on Eesti e-valimiste süsteem turvaline, Ameerika e-valimiste projekt aga mitte. Keskkonnamudeli parameetrid on valitud võimalikult lähedaselt reaalse keskkonna omadustega. Tulenevalt meie piiratud teadmistest mitmete keskkonna parameetrite suhtes, on analüüsi tulemused vaieldavad. Analüüsides põgusalt ründemängu väärtuste tundlikust keskkonnaparameetrite suhtes näeme siiski, et parameetrite mõistlikul tasemel muutused ei mõjuta turvaanalüüsi peamisi tulemusi. Seega, kui meie keskkonnamudel on vastav reaalsele keskkonnale, siis e-valimiste süsteemide turvaanalüüsi tulemused on tõesed.

Introduction

Many of us have dealt with electronic commerce transactions. This is already a part of everyday life. However, e-voting is not yet an obvious method for voting. The construction of electronic voting system is one of the most challenging security-critical tasks, because of the need for finding a trade-off between many seemingly contradictory security requirements like privacy vs. auditability. Thereby it is difficult to adopt ordinary mechanisms of e-commerce. For example, in e-commerce there is always a possibility to dispute about the content of transactions. Buyers get receipts to prove their participation in transactions. E-voters, in turn, must not get any receipts, because this would enable voters to sell their votes.

In 2003, Estonia initiated the project of e-voting. The aim was to implement e-voting in the elections of the local government councils in 2005. In January 2004, a group of American security experts revealed the security report of Secure Electronic Registration and Voting Experiment (SERVE) [1]. The SERVE system was planned for deployment in the 2004 primary and general elections and allows eligible voters to vote electronically via Internet. After examining the security of SERVE, the group of security experts recommended that SERVE should be shut down. They also declared that they do not believe that differently constituted projects could be more secure than SERVE. Their conclusion was that the real barriers to success in e-voting are not skills, resources, etc; it is the fact that given the current Internet and PC security technology, e-voting is an essentially impossible task.

The SERVE project was terminated indeed in January 2004. At the same time, Estonia continued to develop an e-voting system and implemented it according to the plans. The Estonian security experts published their security analysis [2] at the end of 2003. They declared that in *practical sense* the Estonian e-voting system is secure enough for implementation.

This contradicting situation was the main initiator of this work. By closer view, both security reports are consistent and contain truthful and convincing arguments. One of the main reasons for two totally different results was the lack of unified rational security analysis in both reports. Some of the arguments were quite emotional, being based on experts' subjective opinions and "common wisdom".

The aim of the work is to adapt rational security analysis methods for studying the two e-voting systems. It gives us the possibility to compare the practical security of these systems.

In absolutely secure systems unexpected events are not possible. We may dream about such systems, but they can never be achieved in practice. This applies particularly to e-voting systems. Considering the security level of personal computers, it is impossible to design e-voting systems, which are absolutely secure for every user. The most important security goal of voting is not to affect the final results and not to abuse the principles of

democracy. The single incidents with users are still important but they do not have influence to the final result. Moreover, even in traditional voting systems small-scale incidents are acceptable. Therefore, in practical security analysis of e-voting we should concentrate on large-scale threats.

One of the rational approaches of security is known from theoretical cryptography: *security reductions*, which are proofs that security conditions held under certain combinatorial assumptions, such as hardness of factoring or Diffie-Hellman problem. For proving practical security, we also need empirical assumptions about the real world. Moreover, in theoretical cryptography the adversaries are considered to be Turing machines, which are well-defined and relatively easy to study. The real world adversaries are human beings with unpredictable behavior and different motives. Hence, for analyzing practical security, we need real world adversary models. There are works, which attempt to model real world adversaries. In 2006 Buldas *et al* [3] presented a risk analysis method against rational attacks, which used assumptions about real world adversaries. In this work, we are going to adapt their method for analyzing the security of e-voting systems, in particular, for comparing the two systems.

In Chapters 1 and 2, we give the general background of e-voting. In Chapter 3, we describe the Estonian and the SERVE e-voting systems and emphasize the differences of the two systems by paying attention to the points, which could affect the systems' security. However, just pointing out the differences is clearly not enough to claim that one of the systems is secure and the other one not.

In Chapter 4, we give the practical security analysis for the two systems. First, we describe the security analysis method. In Section 4.2, we create the e-voting process models for SERVE and for the Estonian e-voting system. Adversaries are part of the environment and their actions are undesired events. For measuring the security we create an adversarial model in Section 4.3. In our analysis adversaries are rationally thinking persons who attack only, if this is profitable for them. Hence, adversaries estimate the gains and the costs of attacks. In Section 4.4 we define the security assumptions and give their justifications. Security assumptions are certain widely believed conditions, which give the basis of provable security. Section 4.5 gives the security analysis of SERVE and of the Estonian e-voting system based on the security assumptions by using the provable security approach. In this work, we do not completely formalize the security arguments, but in principle they can be formalized. We justify not widely believed assumptions in Subsection 4.6.3. In this justification we also study the influence of society to e-voting security.

In Section 4.6 we justify less obvious assumptions by using attack trees risk analysis. In Subsections 4.6.1 and 4.6.2, we create a hypothetical environment model. First we present the need of environment parameters for analyzing the practical security of e-voting systems. Next, we define the society characteristics, which can affect to success attacks against e-voting systems. For example, we assume that some users notice, if their computers are infected and inform Electoral Committee about that. On the other hand, all voters are not honest; some of them are agree to sell their votes to interest groups who

have purpose to affect the result of voting. Additionally, we consider that some members of the development team of e-voting system can be corrupted. Obviously, it is a serious threat in e-voting systems. Large-scale attacks involve many people and therefore there is always possibility that somebody leaks the information, which could cause the attackers to be caught. We present all these hypothetical characteristics in Subsection 4.6.2. This environment model is not perfect, but can be considered as the first step to formally analyze the influence of society to the security of e-voting systems.

In Subsection 4.6.3., we analyze adversaries' activities in defined environment model for abusing e-voting systems. This empirical analysis uses multi-parameter attack trees [3]. For example, the cost and the success probability are considered as parameters of attack. We justify some of the security assumptions, which were used in previous subsections.

We show that the Estonian e-voting system is practically secure in the defined environment model. The SERVE project has vulnerabilities in the system design, which makes it possible to perform voting-specific attacks. Additionally, we show that reasonable changes in our environment model will not change the results of this analysis. This means that if the defined environment model indeed reflects the reality, then the Estonian e-voting system is more secure than SERVE and the security experts' opinions were reasonable.

It turns out that the main technical disadvantages of SERVE, which make it less secure than the Estonian system, are:

- non-encrypted ballots in an e-voting server;
- no independent log file system to check the correctness of processes of e-voting servers;
- votes counting server is online and contains, besides votes, also the names of voters;
- ballots are not signed by voters.

For defining the environment model, we have tried to estimate the characteristics of environment as close as possible to real society. We have used information from Internet, from research papers, interviews with public prosecutors and studied well-known attacking scenarios. This environment model is not perfect; the estimation of environment characteristics is subjective. However, it defines the need of environment characteristics for analyzing a practical security in e-voting systems. Future works towards refinement of the environment model's characteristics definitely would improve this security analysis.

As far as we know, there are no analogous security analyses published for e-voting systems. Therefore, this work can be considered as one of the first steps in this area.

1. Concept of e-voting

1.1. *E-voting terms*

This chapter gives the explanations of the term “e-voting”. The term “e-voting” is used, in variety of different ways mainly and it encompasses all voting techniques involving electronic voting equipment, including voting over the internet, using booths in polling stations and sometimes even counting of paper ballots.

Electronic voting (e-voting) is any voting method where the voter’s intention is expressed or collected by electronic means. There are considered the following electronic voting ways.

Kiosk voting means the use of dedicated voting machines in polling stations or other controlled locations. Voters mark their choice electronically (perhaps on touch sensitive screen) rather than on paper ballot. The votes are counted on individual machines, known as Direct Recording Electronic (DRE) machines, and the votes cast are transferred to the central tallying point by unspecified means. A ballot paper can be printed and retained in confidence in a ballot box as an additional check.

Remote electronic voting is the preferred term for voting that takes place by electronic means from any location. This could include the use of the Internet, text message, interactive digital TV or touch tone telephone.

Internet voting (i-voting) is a specific case of remote electronic voting, whereby the vote takes place over the Internet such as via a web site or voting applet. Sometimes also used synonymously with Remote Electronic Voting. That usage is however deprecated and it will be used instead as a strict subset of remote electronic voting.

In this work, we use the term e-voting with the specific meaning of Internet voting. If we use it as a general term, then we specify the meaning.

1.2. *Security properties of e-voting*

High security is essential to elections. Democracy relies on broad confidence in the integrity of elections. There has been a lot of attention to an electronic voting by cryptographers. Many scientific researches have been done in order to achieve security, privacy and correctness in electronic voting systems by improving cryptographic protocols of e-voting systems. Currently, the cryptographic schemes are not the main problem. The main interest is the practical security in e-voting systems. Which properties must be justified in order we could say that the system is secure for implementing? One of the main interests is seemingly contradicting security properties. On the one hand, voting must be private and the votes anonymous. On the other hand, voters must be identified in order to guarantee that only the eligible voters are capable to vote. Hence, e-

voting should be uniform, confidential, secure and verifiable. In the following, we define the most important requirements of e-voting.

- 1. Eligible voters are capable to cast ballots that participate in the computation of the final tally.**
- 2. Non-eligible voters are disfranchised.**
- 3. Eligible voters are not capable to cast two ballots that both participate in the computation of the final tally.**
- 4. Votes are secret.**

This is the property of privacy. This property is apparently contradicting property with correctness. On the one hand voting must be private and the votes that are counted anonymous. On the other hand, voters must be identified in order to guarantee that only the eligible voters are capable to vote.

- 5. It is possible for auditors to check whether all correct cast ballots participated in the computation of the final tally.**

This requirement says that a group of dedicated auditors or Electoral Committee can check the correctness of voting.

- 6. The result of an election must be secret until the end of an election.**

The third party must not be capable to reveal the results of the election. Additionally, the system should guarantee that official votes' counting office cannot reveal the final tally before the end of voting. Otherwise, the result of voting could affect voters' decisions during the voting.

- 7. All valid votes are counted correctly and the system outputs the final tally.**
- 8. It must be possible to repeat the computation of the final tally.**

2. State of the art

In this chapter, we give a brief overview of different kinds of electronic voting systems. This list is not perfect; however it gives us a glance of how electronic voting is implemented in Europe and in the United States.

The main reasons for a government to use electronic elections are:

- to increase elections' activity by facilitating the casting of votes by voters;
- to reduce elections' and referendums' expenses;
- to accelerate vote counting and the delivery of voting results;
- to enable voters to cast their votes from different places, not from only a particular polling station.

The Internet voting system [22] was used in the national referendum in Geneva canton of Switzerland in 2004. In Switzerland, elections or referendums are held four or five times a year. There are 580.000 Swiss citizens living abroad, to compare with 7 million inhabitants in the country. It is important to provide them with an efficient and simple voting system. Approximately 52% of the Swiss population has Internet access, both at home and at the workplace. For all these reasons, the governments, both in Geneva and at the Federal level have decided to develop Internet-voting solutions.

The voting cards were sent to voters a few weeks before the voting day. The voting cards were smartcards with private keys validated by a local Public Key Infrastructure service provider. The voting cards were valid for voting operation only. Voters made their choices and confirmed these with the private keys and personal data (date of birth and place of birth). The votes were encrypted in the voting servers by using special public voting keys. The voting system separated voters' personal data and ballots to guarantee the principle of voting privacy. The political parties, in order to check democracy of the votes delivering process, share the keys for triggering votes' counting process.

By the polling of 2003, the 73% of the Swiss population support online Internet voting. However, the Internet voting system has been applied only in referendums. More than 80% of the voters want the system to be implemented for the elections too [22].

The remote voting system was applied in the European Parliamentary elections in the Netherlands in 2004. The target group consisted of the Dutch electors' resident abroad and electors resident in the Netherlands who are temporarily abroad on business on the Election Day and members of their family who accompany them. There was a registration procedure before the elections where eligible voters had to choose the way of elections: by post, by proxy holder, by Internet or by telephone. 41% of the eligible voters preferred the Internet voting system [18]. Nevertheless, the activity of Internet voting was not so high. The main reason why eligible voters did not vote electronically was that they did not receive the voting documents in time.

In the United States of America, there were many attempts made to use electronic voting systems. The project named Voting over the Internet (VOI) was one of them. VOI was used in the general elections of 2000 in four states (Florida, South Carolina, Texas and Utah). The votes given via Internet were legally accepted, but their amount was small (84 votes) [17]. VOI's experiment was so small that it was not a likely target of attacks.

Another Internet voting project named Secure Electronic Registration and Voting Experiment (hereafter SERVE) was developed for primary and general elections in 2004. The SERVE system would have allowed the eligible voters to vote via Internet [1]. The eligible voters of SERVE were mainly overseas voters and military personnel. The target group was 6 million voters. The US Department of Defense terminated the SERVE project in the beginning of year 2004 because a group of security experts had found that the SERVE system was not sufficiently secure.

The projects of the kiosk voting systems have been more successful in the USA. In these systems, like in the paper-based elections, a voter goes to one's home precinct and proves that he/she has a permission to vote there by presenting one's identity card. After that, PINs, smartcards, or some other tokens for authentication are given to voters. Having a token, a voter is able to cast a vote by using a direct recording electronic machine [19].

A public opinion poll held in 2004 showed that 68% of American voters had supported kiosk voting systems while 15% were against it. On the other hand, the positive trust in relation to remote voting systems was 32% and negative attitude was 47% [21].

In Great Britain, many different electronic voting methods have been experimented since 2002, for example, polling booth, telephone, SMS, remote electronic voting via Internet and digital television. Remote electronic voting systems were used in the local election in 30 municipalities in 2003. There were 27% of the voters who voted electronically (146 000 votes) [20]. The majority of all the voters are in favor of Internet voting while only a small group of the voters is against it. Many non-voters are against it too. Even though many eligible voters would not use e-voting methods by themselves, there was a widespread support for making it available to the others.

In 2004, there was an intention to develop the e-voting systems for the European Parliamentary elections and local elections. However, in spring 2004 the decision was made to terminate the development of e-voting systems and concentrate on the voting system via post. The decision was influenced by recommendations of the American security experts, which caused the termination of the Secure Electronic Registration and Voting Experiment project (SERVE).

Estonia has been developing an online Internet voting system since 2003. There were many political discussions whether to allow the implementation of an e-voting system. The Estonian e-voting system was involved in the municipal elections in autumn 2005. On the other hand, a public opinion poll said that general support to e-voting is 73% of voting age inhabitants [13], but the real result was 1.8% e-votes of all votes. There were not successful attacks against the e-voting system. The target group of the e-voting system was 1 million voters.

The security experts are more skeptical about e-voting than the public. Their greatest worries are not related to malicious attacks against e-voting servers, but the system and programming errors and the security of private computers. Another complicated problem seem to be the contradicting properties of correctness and privacy harmony. Additionally, a majority of countries does not apply e-voting to all citizens, but solely to electors' resident abroad. This property expresses also some kind of unreliability.

3. Description of e-voting systems

This chapter presents the detailed descriptions of an e-voting system. In the beginning, we describe how e-voting systems work. Next, we give the descriptions of the Estonian e-voting system and the Internet voting project Secure Electronic Registration and Voting Experiment (SERVE) in the United States of America. Finally, we point out the main differences between the two e-voting systems.

3.1. General description of e-voting systems

Generally, e-voting systems consist of six main phases:

- voters' registration;
- authentication;
- voting and votes' saving;
- votes' managing;
- votes' counting;
- auditing.

The voters' registration is a phase to define voters for the e-voting system and give them authentication data to log into the e-voting system.

The authentication is a phase to verify that the voters have access rights and franchise.

The voting and vote's saving is a phase where eligible voters cast votes and e-voting system saves the received votes from voters.

The votes' managing is a phase in which votes are managed, sorted and prepared for counting.

The votes' counting is the phase to decrypt and count the votes and to output the final tally.

The auditing is a phase to check that eligible voters were capable to vote and their votes participate in the computation of final tally. Additionally there are some other e-voting specific rules verified in this phase.

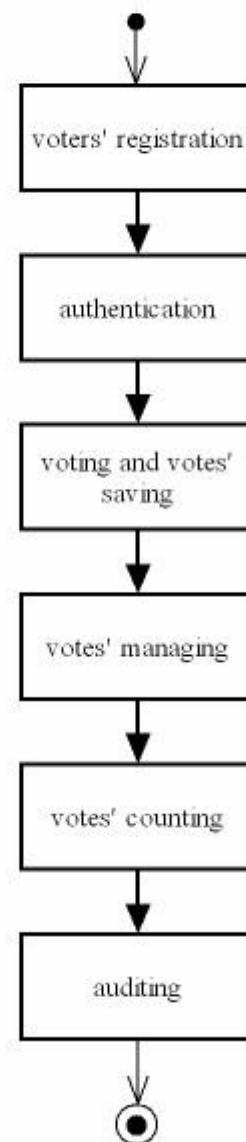


Figure 1. Phases of e-voting.

There are many other relating phases, which were not mentioned. To list some: storing and managing the list of candidates, key generation and management, storing and managing the list of eligible voters, the installation of system initial position, taking down and archiving the system. For the sake of simplicity, we assume that all these phases are secure, and work properly.

Generally, it is possible to divide the e-voting system into three main components of infrastructure:

- Voter Applications;
- Network Server;
- Back-office.

Figure 2 depicts communications between these components.

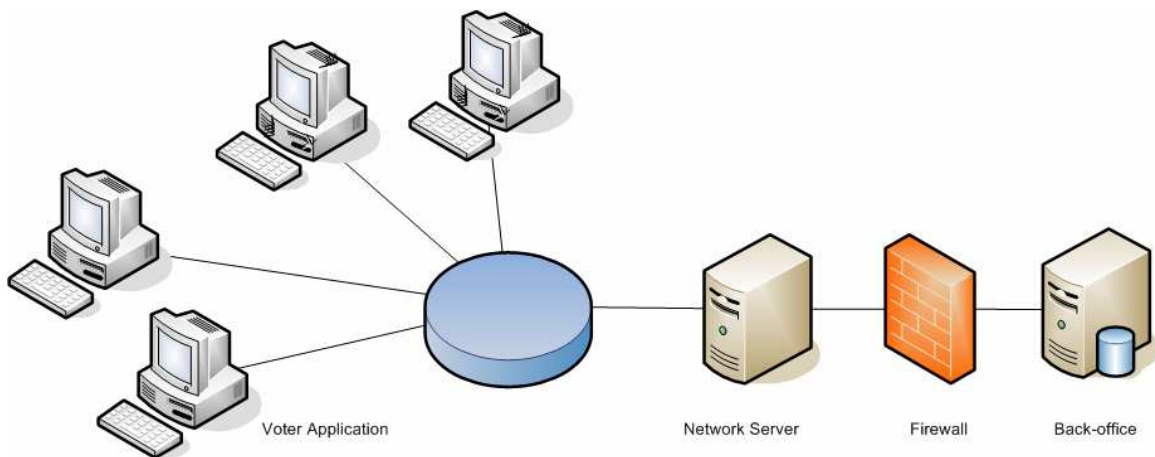


Figure 2. The components of e-voting.

Voter Application is a web application or an application in voters' personal computers for casting votes. Voter application connects to Network Server. Usually, encryptions and authentications methods secure the communication between these components.

Network Server is an online server that provides voters a necessary interface for casting votes. Network Server connects to Back-office server and transfers the received votes to it.

Back-office consists of servers to save and maintain votes and to count a final tally.

In e-voting systems there are many Voter Applications, Network Servers and Back-office servers, but for the sake of simplicity and generalization we consider only one.

In the following, we describe the process of e-voting systems. It starts with a voter connection to Network Server. Next, the voter provides his personal data for authentication. An authenticated voter makes one's choice by using the list of candidates transferred from Network Server. Next, the voter generates a random number r , concatenates it to the vote and encrypts created ballot by using a public key PK of the e-

voting system. It guarantees that without knowing r the voter's choice is hidden. Without a randomized component in the plaintext, it would be possible for an adversary to create ballots for all possible votes, because the encryption key PK is public. It depends on the specific system, if it uses the voter's signature technology or not. Therefore, the voter's signature on the cipher text of ballot $Enc(v, r, PK)$, is optional in the model of the system. In e-voting general model we consider that voters sign encrypted ballots by using their signature private keys.

Network Server receives signed encrypted ballot $Sign(Enc(v, r, PK), SK[i])$ and transfers the accepted signature to Back-office. In order to guarantee that only eligible voters can vote, the processes of Back-office checks the signatures of the ballots and verifies whether voters already voted. If a voter had already voted the systems sends to the voter a signed receipt of voting $Sign(ID, SK[0])$. Votes' managing process saves every cast vote v and voter's personal data ID in Back-office servers. Back-office process replies to each correctly cast vote with a signed receipt $Sign(ID, SK[0])$, which is a confirmation of the voting system that the vote of the i -th voter has been correctly cast. Receipts do not contain any information about the corresponding votes. The voter can verify the signature $Sign(ID, SK[0])$ with public key $PK[0]$ that corresponds to $SK[0]$.

When the voting period is ended, Back-office's votes' counting process computes the final tally. Back-office outputs the signed final tally and the signed list of voters.

Figure 3 depicts the process of e-voting systems. The activities and their abridgements in the model are given in Table 1 and Table 2:

Table 1. The function of general e-voting model.

Authentication	process for authentication
Cast	process to cast a vote
Random	function for generate random number
Enc	function for encrypting
Sign	function for digitally signing encrypted ballots
Save	function for saving data to following transmission
Count	function for counting the final tally

Table 2. Data items and their abridgements of general e-voting model.

PK	the public key of the e-voting system which is used to encrypt ballot
SK	the secret key of the e-voting system, which is used to decrypt encrypted ballot in the back-office server
SK[i]	the private signature key of eligible voter
PK[i]	the public signature key of eligible voter
SK[0]	the private key of back-office for signing the voting confirmation
PK[0]	the public key of back – office for verifying the signature of the voting confirmation
v	a voter's choice, vote
r	randomly generated number
ID	voter's personal data file

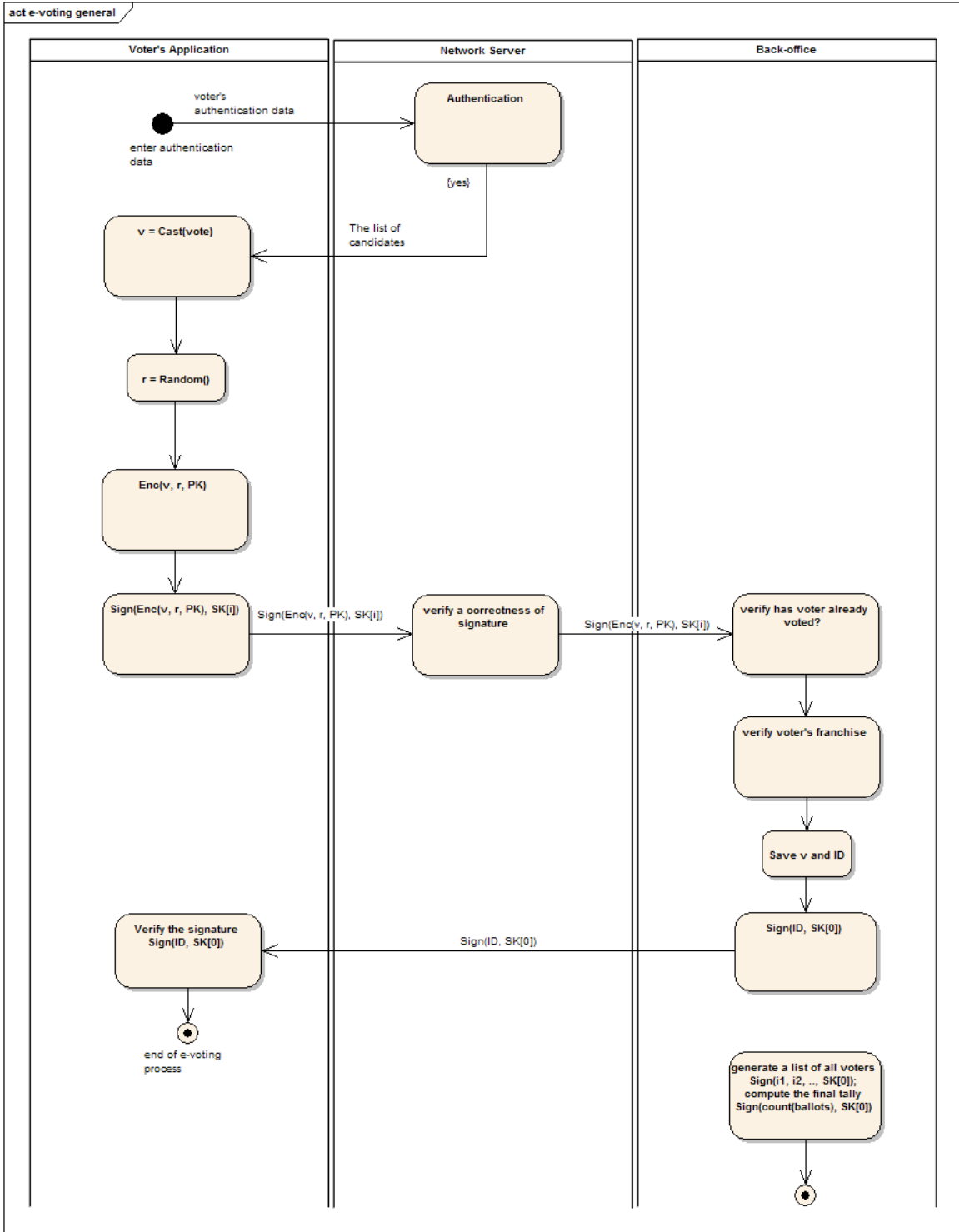


Figure 3. The description of e-voting.

3.2. Estonian e-voting system

The Estonian e-voting system is implemented from the sixth day up to the fourth day before the Election Day. There are following principles in the Estonian e-voting system:

- each eligible voter is able to revoke. In this case the older votes are deleted.
- classical voting in polling box cancels the voters' electronic votes.
- if considerable attacks against e-voting have been detected, Electoral Committee might stop e-voting and cancel the result of voting.

The main components of the Estonian e-voting systems are Voter Application, Network Server; and Back-office is divided into two Votes Storing Server and Votes Counting Server. These components have following described processes [5].

Voter Application is a web application. The encryption and authentication built into the Secure Socket Layer (SSL) protocol protect the communication between voters and Network Server. The Estonian e-voting system is able to run on Windows, Linux and MacOS operation systems. In the Windows operation system it is required to use Microsoft Internet Explorer. The public key *PK* of e-voting is integrated into Voting Application. Voter Application uses signed ActiveX application.

The processes of Network Server are authentication, the checking of franchise, sending a candidates' list to voters, receiving signed and encrypted ballots. Network Server immediately transfers the received encrypted ballots to Votes Storing Server and transposes the acknowledgements of receipt from Votes Storing Server to voters. Network Server completes the work at the moment when the period of e-voting finishes.

Votes Storing Server receives encrypted ballots from Network Server and stores them until the end of voting period. One of the specific properties of the Estonian e-voting system is an option to cast a vote more than once. The last vote is taken into account. Votes Storing Server has a responsibility of votes' managing and canceling.

Votes Counting Server is an offline server, which summarizes all encrypted ballots. The encrypted ballots are transferred from Votes Storing Server to Votes Counting Server by using data carriers. Votes Counting Server does not get voters' digital signatures and it does not know voters' personal data.

Additionally, e-voting system delivers independent log files, which consist of trace of the received encrypted ballots from Network Server, all annulled encrypted ballots, all encrypted ballots sent to Votes Counting Server and all counted encrypted ballots. All the records in the log files are linked by using cryptographic protocol. The electoral committee has the right to use the log files for resolving disputes.

The process of auditing uses the independent log files of e-voting system. There are the following logs:

LOG1: received signed encrypted ballots;

LOG2: cancelled signed encrypted ballots with reasons of cancellation;
LOG3: signed encrypted ballots which are transferred to Votes Counting Server;
LOG4: invalid encrypted ballots;
LOG5: counted encrypted ballots.

In the process of auditing, there is possibility to verify the integrity of log files. The intersection of LOG2 and LOG3 must give LOG1. The intersection of LOG4 and LOG5 must give the content of LOG3.

Hence, there is the independent audit trail to verify e-voting process and to help solve problems in the Estonian e-voting system.

Figure 4 depicts the components of e-voting system.

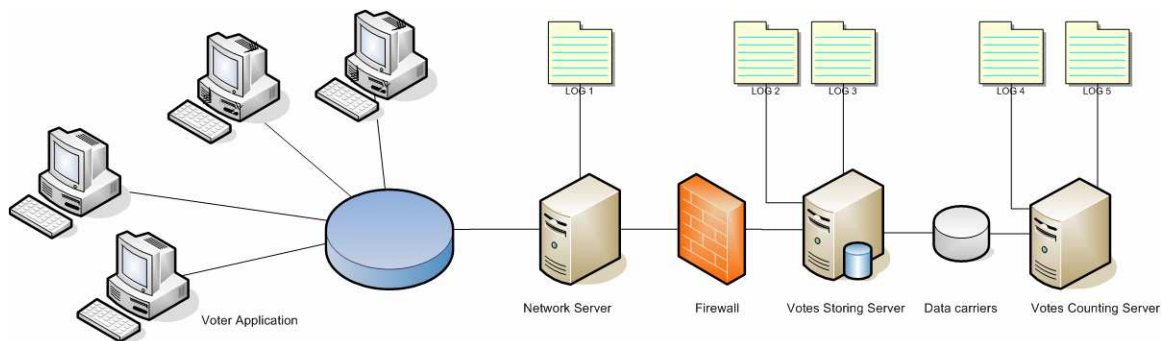


Figure 4. The components of the Estonian e-voting system.

RSA based national Public Key Infrastructure (PKI) is applied in the Estonian e-voting system. Hence, 74% of citizens [23] have identity card (hereafter ID-card) as a passport with authentication and digital signatures certificate. Every voter who prefers to vote electronically must have an ID-card. The authentication of e-voting system and voter's digital signature are applied by using the Estonia's Public Key Infrastructure. Therefore a Certification Authority is one of the concerned parties in the Estonian e-voting system. For the sake of simplicity, it is not concentrated to PKI system and it is assumed to be secure.

There is no classical voters' registration process in the Estonian e-voting system. Data of eligible voters come from the State Population Register and from previously mentioned information of authentication based on ID-card technology.

The processes of e-voting are described by using public information from the web page of the Estonian National Electoral Committee [4]. For authentication, a voter connects to Network Server by using https protocol and authenticates oneself by using the ID-card with authentication key. When the connection is established, then a signed ActiveX

control is downloaded to voter's computer. Network Server verifies voter's franchise by sending a query to State Population Register. If the answer is negative, then Network Server delivers message to voter for that purpose. Otherwise, Network Server makes a query to Votes Storing Server to ask whether the voter had already voted. The query result is transmitted to the voter and in the case of positive answer; the voter can choose to cast a vote again.

In the phase of voting, Network Server reveals a list of candidates and the voter makes one's choice. Next, the application generates a random number to guarantee non-deterministic cryptogram and encrypts the vote and the random number by using the public key of Votes Counting Server. Hereafter, the cipher text $Enc_bal = Enc(v, r, PK)$ is named encrypted ballot. Next, the voter signs the encrypted ballot by using the personal private key $SK[i]$. The voter's application sends the signed and encrypted ballot $Sign_Enc_bal = Sign(Enc_bal, SK[i])$ to Network Server. Network Server compares whether the session owner is the same person who had signed the encrypted ballot and in case of positive acknowledgment, transfers the signed and encrypted ballot to Votes Storing Server. Votes Storing Server connects to the Certification Authority and provides the attestation of digital signature validation. The system replies to each correctly cast vote with a receipt *Response*. *Response* is a text type file and consists of the information about reception of ballots. Additionally, the record of received signed encrypted ballot and voter's personal data is attached into the log file named LOG1.

The votes' managing phase starts after the end of the e-voting period. Votes Storing Server cancels all the voters' multiple signed encrypted ballots and saves the trace of cancelled signed encrypted ballots together with the reasons of canceling into the log file named LOG2. Votes Storing Server verifies voters' franchise by making query to the list of voters by using personal data from digital signatures. Next, the server separates digital signatures and encrypted ballots.

The encrypted ballots Enc_bal are transferred to offline Votes Counting Server by using data carriers. The encrypted ballots are decrypted by using the private key SK of Votes Counting Server. Next, it is verified that the format of the decrypted ballots corresponds to the fixed rules. Non-accepted encrypted ballots are saved into the log file named LOG4. Votes Counting Server counts only positively verified votes. All counted encrypted ballots are saved into the log file named LOG5. The phase of votes' counting is repeatable.

The activities and their abridgements in the system are following:

Table 3. The functions of the Estonian e-voting system.

Authentication	process for authentication
Enc	function for encrypting
Dec	function for decrypting ballots
Sign	function for digitally signing encrypted ballots
Random	function for generate random number
Cast	process to cast a vote
Send	process to send data to other participant
Verify_1	function to verify voter's franchise
Verify_2	function to check if voter has already cast a vote
Verify_3	function to verify the correctness of digital signature and session owner correspondence to digital signer
Verify_4	function to verify validation of digital signature
Verify_5	function to cancel voter's multiple votes
Verify_6	function to verify the correctness of vote's format
Separate	function for separating voter's personal data and encrypted ballot
Save	function for saving data to following transmission
Count	function for counting the final tally

The relevant data items and their abridgements:

Table 4. The data items of the functions of the Estonian e-voting system.

PK	the public key of the e-voting system which is used to encrypt ballot
SK	the secret key of the e-voting system, which is used to decrypt encrypted ballot in Votes Counting Server
SK[i]	the private signature key of eligible voter
PK[i]	the public signature key of eligible voter
v	a voter's choice
r	a randomly generated number
bal	a ballot which is formed voter's choice v and random number r ;
Enc_bal	encrypted ballot
Sign_Enc_bal	signed encrypted ballot
ID	voter's personal data file
Response	the plain text file with message

Figure 4 depicts the process of Estonian e-voting system.

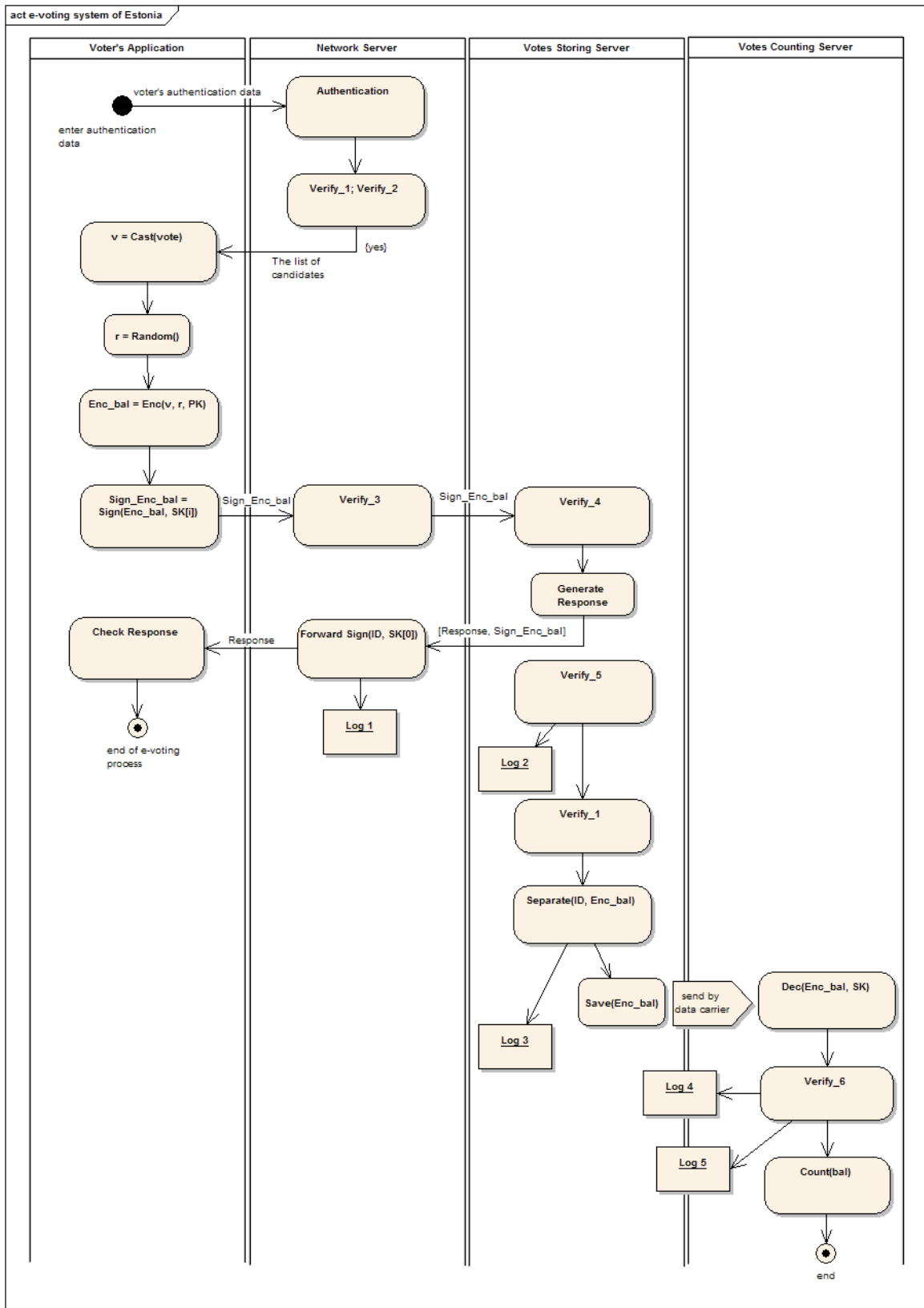


Figure 5. The description of the Estonian e-voting system.

3.3. SERVE e-voting system

We refer to the SERVE e-voting system as the SERVE system and as the SERVE project. On the one hand, it is an e-voting system, but due to the case that the United States didn't use it in elections, we may appoint it like a project.

In the SERVE system, it is possible to vote any time within 30 days before the Election Day until the closing time of polls on the Election Day. Every eligible voter can cast a vote only once. There are no Public Key Infrastructure and ID-cards used in SERVE. If Electoral Committee is informed of considerable attacks against the e-voting system, the e-voting might be terminated and the result cancelled.

The main participants in SERVE are Voter Application, Network Server and Back-office that is divided into Votes Storing Server and Votes Counting Server.

Voter Application is a web application. Voters' computers must run a Microsoft Windows operating system and either the Internet Explorer or the Netscape web browser. The browser must be configured to enable JavaScript and either Java or ActiveX scripting and it must permit session cookies. Like the Estonian e-voting system, the encryption and authentication built into the SSL protocol, protects the communication between the user's web browser and the voting application on Network Server.

Network Server is an online server. Network Server receives ballots and personal information from voters, encrypts this data, and transmits cipher texts to Votes Storing Server.

Votes Storing Server verifies the voters' rights of access and franchise. But the most important is that the server decrypts the received cipher texts. Hence, the ballots are in the server in non-encrypted way within a little period until the server encrypts the ballots again. The server retains encrypted ballots and the list of voters' personal data, even after a copy has been sent to Votes Counting Server. The aim to store the list of voters is to verify whether the voter has already cast a vote.

The SERVE system has many Votes Counting Servers named Local Election Office. Every voting district that participates in SERVE has online Votes Counting Server. Every Votes Counting Server generates an e-voting key pair. The Votes Counting Server's public key is used to encrypt the ballots voters from that certain district. Encrypted ballots can be read exclusively by using the private key known only to Votes Counting Server. There are direct interactions between Votes Storing Server and Votes Counting Servers for downloading the list of voters' personal data and encrypted ballots. To model the activities of system we focus on one Votes Counting Server only.

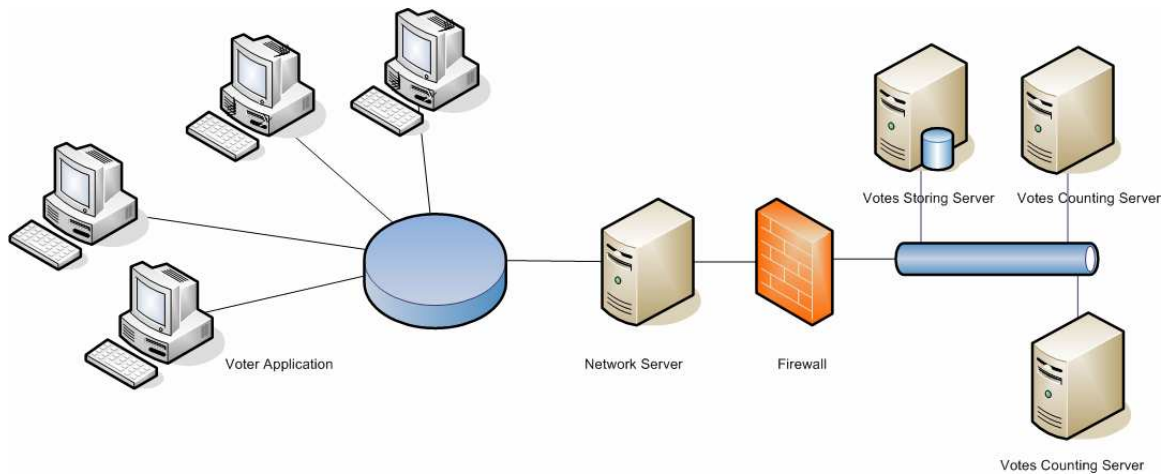


Figure 6. The infrastructure of the SERVE project.

We describe the processes of the SERVE e-voting system by using the information from D. Jefferson, A.D. Rubin, B. Simons, D. Wagner paper “A Security Analysis of the Secure Electronic Registration and Voting Experiment”.

To participate in the e-voting process, an eligible voter must firstly enroll for the SERVE program. After the enrollment, the voter will be able to register oneself as a voter. The authentication of the SERVE system uses password-based service. To authenticate oneself, the voter connects over https protocol to Network Server and inserts the login data. Additionally, Network Server verifies the voter’s franchise. Once the connection is established, a trusted ActiveX control is downloaded into the voter’s computer when voter uses Internet Explorer browser, or for Netscape users-a Java applet runs are downloaded.

Next, Network Server reveals the list of candidates and the voter makes one’s choice. The application encrypts a vote v and a generated random number r by using SERVE’s public key $PK[S]$. Voter Application casts the encrypted ballot $Enc_bal_1 = Enc(v, r, PK[S])$ and voter’s personal data ID to Network Server. After the verification of correctness of message the encrypted ballot and voter’s personal data are transferred from Network Server to Votes Storing Server.

In the votes’ managing phase, the Votes Storing Server verifies that a voter is registered and has not yet voted. The server generates a *Response* for each accepted vote. In case the voter has already voted the receipt *Response* with corresponding answer is generated and voting process terminates. Accepted encrypted ballots follow the process in Votes Storing Server. The server decrypts cipher text of votes using the private key of SERVE and separates the ballots and the voters’ personal data. Afterwards, Votes Storing Server encrypts ballots without the voters’ personal data using the public key of Votes Counting Server. Votes Storing Server retains the encrypted ballots $Enc_bal_2 = Enc(v, r, PK) = Enc(bal, PK)$ and the list of voters who have already cast a vote. Votes Counting Server downloads the list of voters and the encrypted ballots from Votes Storing Server when Votes Counting Server updates its database. For counting votes, Votes Counting Server

decrypts the encrypted ballots by using the private key SK of Votes Counting Server. Only accepted format of votes are counted to the final tally. The computation of the votes is repeatable.

In SERVE there are two possibilities to verify which components of the system have received the information about the voters or the ballots. The first option is to check the list of voters in Votes Counting Server and another option is to make a query to the list of voters in Votes Storing Server. Both servers retain the voters' list and the encrypted ballots separately. It is not possible to verify whether a voter's vote participated in the calculation of the final tally. To summarize, there is no independent audit trail of votes to verify the e-voting process.

The activities and their abridgements in the system are following:

Table 5. The functions of the SERVE system.

Authentication	process for authentication
Enc	function for encrypting
Dec	function for decrypting ballots
Random	function for generate random number
Cast	process to cast a vote
Send	process to send data to other participant
Verify_1	function to verify voter's franchise
Verify_2	function to check if voter has already cast a vote
Verify_6	function to verify the correctness of vote's format
Download	function to download the list of voters' data and encrypted ballots
Separate	function for separating voter's personal data and ballot
Save	function for saving data to following transmission
Count	function for counting the final tally

The relevant data items and their abridgements:

Table 6. The data items of the functions of SERVE.

PK	the public key of the e-voting system which is used to encrypt ballot
SK	the secret key of the e-voting system, which is used to decrypt encrypted ballot in the Votes Counting Server
SK[S]	the private signature key of SERVE
PK[S]	the public signature key of SERVE
v	a voter's choice
r	a randomly generated number
Bal	a ballot which is formed voter's choice v and random number r ;
Enc_bal	encrypted ballot
ID	voter's personal data file
Response	the plain text file with message

Figure 7 depicts an e-voting model of SERVE.

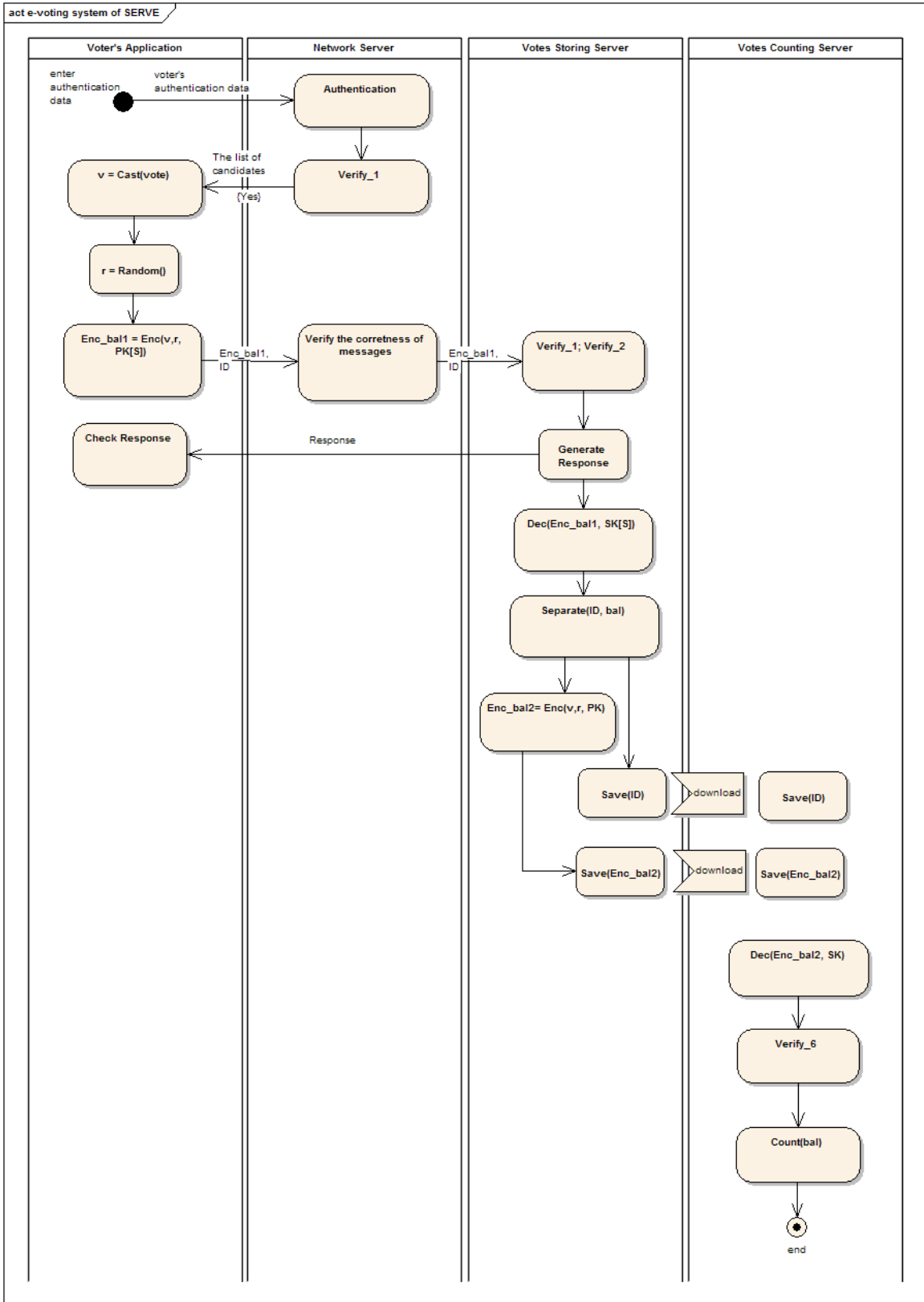


Figure 7. The description of the SERVE system.

3.4. Differences between two systems

This section brings out the main differences between the Estonian e-voting system and SERVE. It does not define which system is more secure. Comparison of two e-voting systems is not sufficient to analyze the systems security in a rational way. Table 7 points out briefly the main differences between the two e-voting systems.

Table 7. The differences of the two e-voting systems.

	Characteristic	The Estonian e-voting system	SERVE
1)	The period of e-voting	The e-voting is implemented within a period before the Election Day	The e-voting is implemented within a period before the Election Day and on the Election Day
2)	Revoting process in the polling station	Yes	No
3)	National Public Key Infrastructure	Yes	No
4)	A voter signs the encrypted ballot	Yes	No
5)	The state of votes in Votes Storing Server	Encrypted ballot	No encrypted ballot
6)	The state of Votes Counting Server	Offline	Online
7)	Log files system	Yes	No

The first reviewed differences between two systems are the period of e-voting and revoting in polling stations. The Estonian e-voting system is implemented from the sixth day to the fourth day before the Election Day. On the Election Day voters can vote in polling stations by classical way, which cancels voters' e-votes. In the SERVE system, it is possible to vote any time within 30 days before the Election Day until the closing time of polls on the Election Day. Every eligible voter can cast a vote only once. Hence voters are not able to revote by using classical voting process.

The Estonian e-voting system uses national Public Key Infrastructure with authentication and digital signatures certificate. Therefore, there is no specific e-voters registration process in the Estonian e-voting system. The Certification Authority gives the certification of authentication and the State Population register gives the data of eligible voters. There is no Public Key Infrastructure used in SERVE. In SERVE there are password based authentication and voters' registration process.

Additionally, Estonian e-voters sign encrypted ballots by using digital signature certificate of national Public Key Infrastructure. Estonian Voter Application sends signed

and encrypted ballots to Network Server. In SERVE the digital signature technology of voters is not used and encrypted ballots are transferred to Network Server.

One of the main differences between the two systems is that in SERVE the received ballots are held non-encrypted within a short period of time in Votes Storing Server. In the votes' managing process in SERVE the encrypted ballots are decrypted by using the public key of SERVE and then encrypted again by using the public key of Votes Counting Server. In the Estonian e-voting system Voter Application encrypts ballots and only Votes Counting Server is able to decrypt them. Hence, ballots are in encrypted way in Votes Storing Server of the Estonian e-voting system.

Votes Counting Server in the SERVE system is online. Votes Counting Server updates its database in every certain moment by downloading the list of voters and encrypted ballots from Votes Storing Server. Votes Counting Server of the Estonian system is offline. Encrypted ballots are transferred to Votes Counting Server by using data carriers. The list of voters is not saved in Votes Counting Server.

The Estonian e-voting system has the system of independent log files for auditing process. The log files have information of received encrypted ballots and encrypted ballots, which are transferred to Votes Counting Server. SERVE does not deliver any independent log files. For the counting process, prepared encrypted ballots and the list of voters are saved in Votes Storing Server.

4. Security analysis

To measure the security of security-critical systems like e-voting, we should analyze the security in an objective way. We cannot use informal risk analysis based on security experts' experience, knowledge and opinion. We need a structured method to determine whether the system is secure. In this chapter, we give the practical security analyzes method for the e-voting systems.

4.1. Analysis method

E-voting is one possible way to vote; therefore, mainly it has the same security requirements as classical voting. If e-voting has the same security properties as the classical voting, then we may consider that also e-voting is secure. Obviously, classical voting systems are not absolutely secure. There might be small-scale misbehaviors in paper based voting systems. Hence, e-voting provides more risk of having large-scale attacks against voting properties. We describe them as voting-specific attacks and give also the justification for these specific attacks.

For analyzing the security of e-voting, we have to define the desired properties of the system. Desired properties represent the intended structure and functioning of an e-voting system. In this work, desired properties are pointed out in Section 1.2. Secondly, we create models for e-voting systems in Section 4.2. The model of system consists of processes and communication between components of the system. We are going to analyze the security of e-voting systems in an environment model. Therefore, we create the environment model as similar as possible to the real world. We defined the environment model by using security assumptions, the properties of adversaries, and society characteristics. We define the security assumptions as believed and proved security conditions in the environment. For proving practical security, we also need empirical assumptions about the real world. Therefore, we define the environment characteristics by using information from Internet, from research papers and interviews with public prosecutors. We describe the model of adversarial based on the threats of the system and voting-specific attacks. We assume that adversaries are rationally thinking persons and attack with purpose to affect the result of elections. We analyze adversarial behavior by using attack tree method and a game-theoretic risk analysis.

Attack trees [10] provide a formal method of describing the security of systems, based on varying attacks. Figure 8 depicts the example of attack tree. Basically attack trees represent attacks against a system in a tree structure. The root node represents the goal of attack and sub nodes represent different ways how to achieve the goal. Nodes are divided into child nodes and parent nodes. A parent node may be a child of another parent. Child nodes are conditions, which must be satisfied to make the direct parent node's condition true. There are two types of conditions: AND and OR. They represent logical operations. To satisfy the condition of an OR node, it is sufficient to satisfy at least one of his child nodes. The node of AND conditions is true if every child node is satisfied. When the condition of root node is satisfied, the attack is complete.

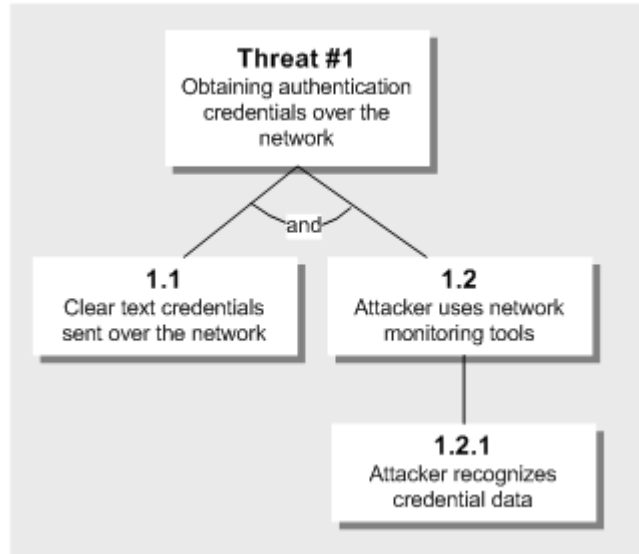


Figure 8. Attack Tree.

Therefore, attack tree can also be viewed as attack planning tree. When attackers plan to attack, they would probably consider the gains and the costs of the attack. Hence, we may assume that attackers behave in a rational way. It means that attackers do not attack, if the attack is unprofitable and attackers always choose the most profitable ways for attacking. Therefore, beside the cost and the gain of attack attackers also consider the probability of success, the probability of getting caught and the penalties. These parameters are all involved in the decision-making process of a rational attacker.

Buldas *et al* [3] present multi-parameter attack trees that use aforementioned parameters in the calculation of attack trees. They view attack as a game played by attacker. The parameters of the game are:

- *Gains* – the gains of the attacker, in case the attack succeeds;
- *Costs* – the cost of the attack;
- p – the success probability of the attack;
- q - the probability of getting caught (in case the attack was successful);
- *Penalties* – the penalties in case the attacker is caught (assuming that the attack was successful);
- q_- - the probability of getting caught (in case the attack was not successful);
- *Penalties₋* - the penalties in case the attacker is caught (assuming that the attack was not successful).

The model of the attack game presents attacker’s rational thinking in Figure 9. The attack game starts with preparations and the attacker calculates a cost of attack. Next, the attacker considers with the probability p to succeed the attack and to get gains from the attack. After the attack, it is possible that the attacker will be detected and will be caught. Hence, the rational attacker estimates this possibility and penalties so that an outcome ratio will be $- Costs + Gains - Penalties$. The attacker may also get caught if the attack

will be unsuccessful; therefore it is necessary to estimate also the probability to get caught and penalties.

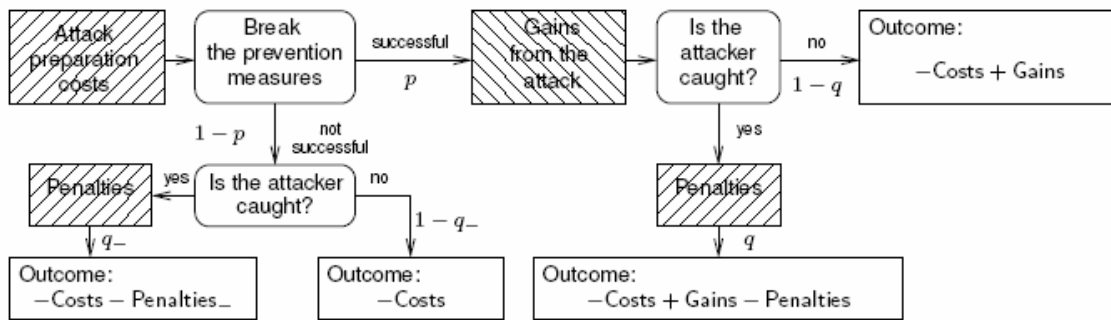


Figure 9. Diagram of the attack game from the attacker's point of view [3].

Considering all these parameters attacker calculates the expected outcome of the attack game, which gives also the rational answer for attacking.

$$\begin{aligned}
 Outcome &= (1 - p) \cdot [q_- \cdot (-Cost - Penalties_-) + (1 - q_-) \cdot (-Costs)] + \\
 & p \cdot [q \cdot (-Costs + Gains - Penalties) + (1 - q) \cdot (-Costs + Gains)] = \\
 & = -Costs + p \cdot (Gains - q \cdot Penalties) - (1 - p) \cdot q_- \cdot Penalties_-.
 \end{aligned}$$

For the attacker the attack is unprofitable if $Outcome < 0$. Hence, in our security analysis we may consider attacks as serious attacks if $Outcome > 0$.

For the sake of simplicity we denote

- by π the average penalty of an attacker in case the attack was successful $\pi = q \cdot Penalties$ and
- by π_- the average penalty in case the attack was not successful $\pi_- = q_- \cdot Penalties_-$.

so we have

$$Outcome = -Costs + p \cdot (Gains - \pi) - (1 - p) \cdot \pi_- .$$

The multi-parameter attack tree method consists of two phases:

- 1) to identify primary attacks as ultimate goals for attackers;
- 2) to create attack tree for ultimate goal and computing the tree in order to determine the most profitable attack and to decide whether the attack game outcome is positive.

A primary attack is an event that directly causes loss. For example “the configuration vulnerabilities of e-voting’s server” is not a primary attack, however “non-eligible voter is able to cast a vote that participates in the computation of the final tally” causes serious damage in e-voting. In our analysis we consider the primary attacks as undesirable events against e-voting system security.

If the set of primary attacks is fixed, the second step of the analysis is to construct an attack tree for each primary attack. For computing attack trees we need for each node the values of parameters: $Costs$, p , $Gains$, π , π_- . It's hard to estimate the gain for each child node. A rational goal oriented attacker is interested in achieving the main goal of an attack. For these reasons, we assume that the value of $Gains$ is constant for every node of a tree. For child nodes the parameters are deduced from experts' assumptions and social researches. For leaf nodes, these parameters are computed based on the corresponding parameters of the child nodes. In addition to the parameters, the $Outcome$ value is computed for all nodes. The parameters of leaf nodes are computed as follows:

- for an OR-conditions with child nodes with parameters $(Costs_1, p_1, \pi_1, \pi_{1-})$ and $(Costs_2, p_2, \pi_2, \pi_{2-})$ the parameters $(Costs, p, \pi, \pi_-)$ are computed as follows:

$$(Costs, p, \pi, \pi_-) = \begin{cases} (Costs_1, p_1, \pi_1, \pi_{1-}), & \text{if } Outcome_1 > Outcome_2 \\ (Costs_2, p_2, \pi_2, \pi_{2-}), & \text{if } Outcome_1 \leq Outcome_2 \end{cases},$$

where

$$Outcome_1 = -Costs_1 + p_1 \cdot (Gains - \pi_1) - (1 - p_1) \cdot \pi_{1-}$$

$$Outcome_2 = -Costs_2 + p_2 \cdot (Gains - \pi_2) - (1 - p_2) \cdot \pi_{2-}.$$

- for an AND-conditions with child nodes with parameters $(Costs_1, p_1, \pi_1, \pi_{1-})$ and $(Costs_2, p_2, \pi_2, \pi_{2-})$ the parameters $(Costs, p, \pi, \pi_-)$ are computed as follows:

$$Costs = Costs_1 + Costs_2$$

$$p = p_1 \cdot p_2$$

$$\pi = \pi_1 + \pi_2$$

$$\pi_- = \frac{p_1(1 - p_2)(\pi_1 + \pi_{2-}) + (1 - p_1)p_2(\pi_{1-} + \pi_2) + (1 - p_1)(1 - p_2)(\pi_{1-} + \pi_{2-})}{1 - p_1p_2}.$$

The formula π_- represents the average penalty of an attacker, assuming that at least one of the two child-attacks was not successful. For example, if the first attack was successful and the second one unsuccessful (which is an event with probability $p(1 - p_2)$), then the average penalty of the attacker is $\pi_1 + \pi_{2-}$.

The opportunity to measure and compare the security of e-voting systems with multi parameter attack tree gives us a rational risk analysis method to estimate the importance of threats.

4.2. E-voting models for SERVE and the Estonian e-voting system

In the following, we give the models of the SERVE project and the Estonian e-voting system. For the sake of simplicity, the processes of an e-voting system are described by using communicating synchronous abstract automata. In a synchronous automaton the changes of states happen in fixed moments. Such approach avoids the synchronization problems of real-time systems. Automata theory provides a sufficiently universal model for analyzing the security of voting system.

For constructing the models of systems we focus on four components:

- Voter Application,
- Network Server,
- Votes Storing Server
- Votes Counting Server.

We describe all components in general level of abstraction as black boxes. We do not describe how the processes are constructed, for example how the votes are saved in Votes Storing Server.

The processes in Voter Application of SERVE and the Estonian e-voting system are quite similar. Figure 10 depicts Voter Application processes of the Estonian e-voting system and Figure 11 does Voter Application of SERVE.

Voter Applications of both systems send a request to Network Server for establishing a secure connection. If Voter Application receives the certificate of Network Server, he has to make a decision whether to verify it or not. In our model, we consider this as a probabilistic event. Voter Application verifies the server's certificate with probability P and if the certificate is not accepted Voter Application terminates the establishing of connection. Otherwise, an SSL connection is created. Voter Application sends to Network Server the authentication data. Next, both e-voting systems ask if Voter Application accepts the signed ActiveX component. We do not consider the case if a voter does not do that, because in this case the voting process is terminated. The voter decides whether to verify the signature on the ActiveX component. With probability $1-X$ the voter downloads the ActiveX component without verifying it. After that, Voter Application receives a list of candidates.

Starting from this point, the two systems act differently. In the Estonian e-voting system Voter Application creates a vote, generates a random number and encrypts the ballot by using a public key PK of the e-voting system. Voter Application signs the encrypted ballot with voter's private signature key $SK[i]$ and sends the signed encrypted ballot to Network Server. In SERVE Voter Application casts a vote, generates a random number and creates an encrypted ballot by using the public key $PK[S]$ of SERVE. Voter Application sends the encrypted ballot with voter's personal data to Network Server.

Finally, if the Estonian e-voting system and SERVE accept the received ballots, Voter Application receives a confirmation *Response*, which confirms that voter's vote reached to Votes Storing Server. We assume that with probability Q , Voter Application waits for the confirmation and checks it.

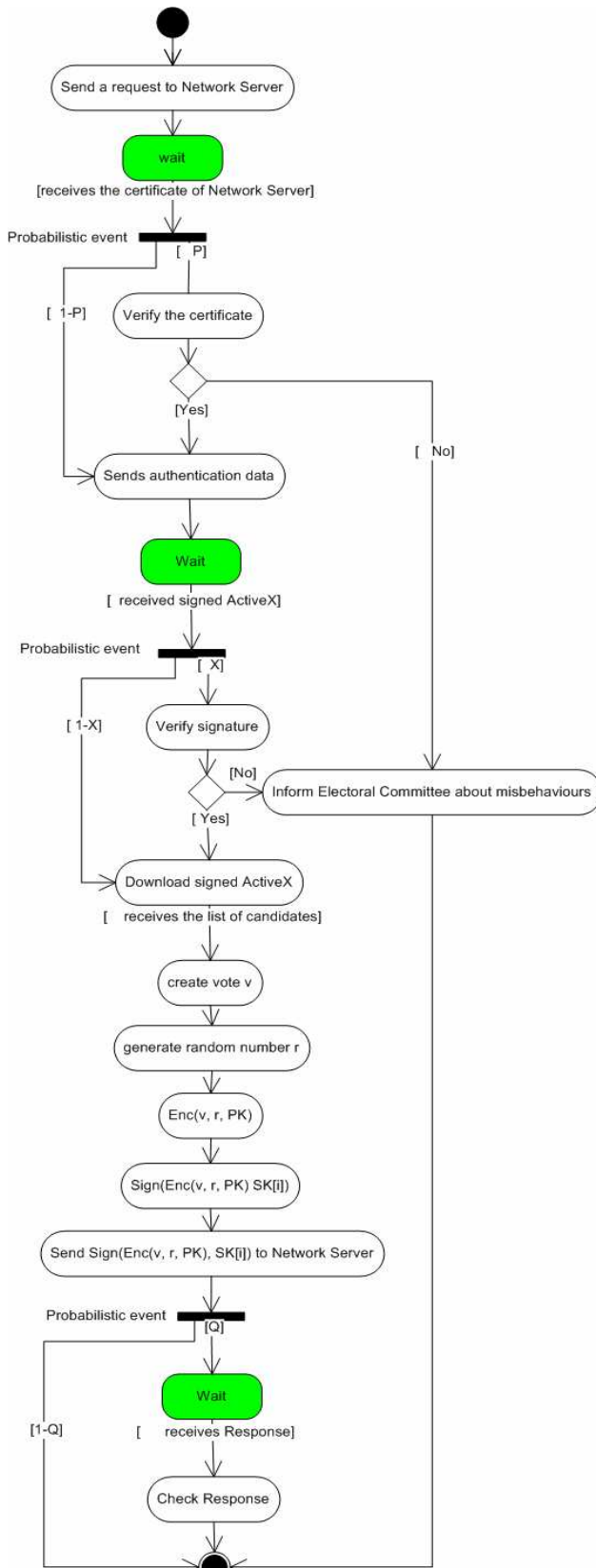


Figure 10. Voter Application of the Estonian e-voting system.

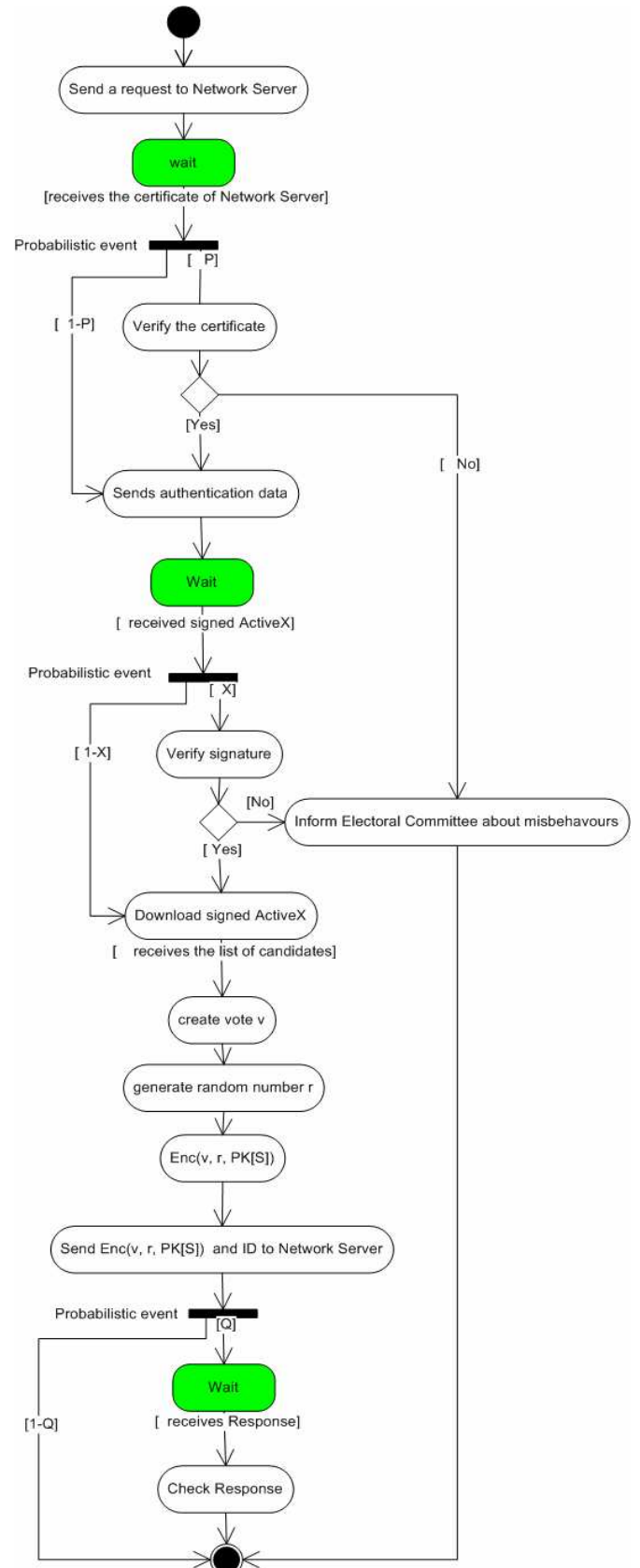


Figure 11. Voter Application of SERVE.

Network Servers of the two systems are different. First, we describe corresponding server of the Estonian e-voting system. Figure 12 depicts the processes of Network Server of the Estonian e-voting system. If Network Server receives Voter's Application request it replies with server's certificate for establishing an SSL connection. If Voter Application accepts the certificate, the SSL connection is created. On an authentication data of a voter, Network Server verifies that the voter has access rights and franchise. In the positive case, server sends a query to Votes Storing Server in order to check whether the voter has already cast a vote. If the voter has voted, Network Server forwards a receipt to Voter Application. Anyway, the voter is able to cast a vote again. Network Server sends the list of candidates to Voter Application. On the received signed encrypted ballot Network Server checks, if the signer of the encrypted ballot is the same voter who initiated the SSL connection. Network Server forwards correct ballots with voters' signatures to Votes Storing Server. Network Server waits for an answer from Votes Storing Server. If the answer is received, Network Server forwards it to Voter Application and saves voter's personal data *ID* and encrypted ballot to log file LOG1.

Network Server of SERVE sends server's certificate to Voter Application immediately after receiving a request. When Network Server receives voter's authentication data, it verifies has the voter a right to vote. Upon positive answer, Network Server sends a list of candidates to Voter Application. Otherwise it sends an error message to Voter Application. On received signed encrypted ballot Network Server verifies the correctness of message and forwards it to Votes Storing Server. Network Server waits for an answer from Votes Storing Server and when the answer is received, the server forwards it to Voter Application. Figure 13 depicts the processes of Network Server of SERVE.

The Estonian e-voting system and the SERVE system have principal differences in Votes Storing Servers. Figure 14 depicts Votes Storing Server of the Estonian system. On Network Server's request "voter has already voted", Votes Storing Server compares the signatures of voters and replies with a receipt. On received signed encrypted ballots, Votes Storing Server verifies a validation of signature. Voter Application replies to each correctly cast vote with a plain text type *Response*, which is a confirmation of the voting system that the vote has been correctly cast. Receipts do not contain any information about the corresponding votes. Additionally, Network Server saves signed encrypted ballots and voter's personal data into log file named LOG1. If the voting period is over, Votes Storing Server eliminates multiple votes—only the most recent vote counts. The log file LOG2 is delivered with results of votes' managing. Before the separation of encrypted ballots and voters' signatures, server checks, if the owners of the encrypted ballots are eligible to vote. Votes Storing Server delivers the list of voters and the file LOG3 with encrypted ballots, which are transferred to Votes Counting Server.

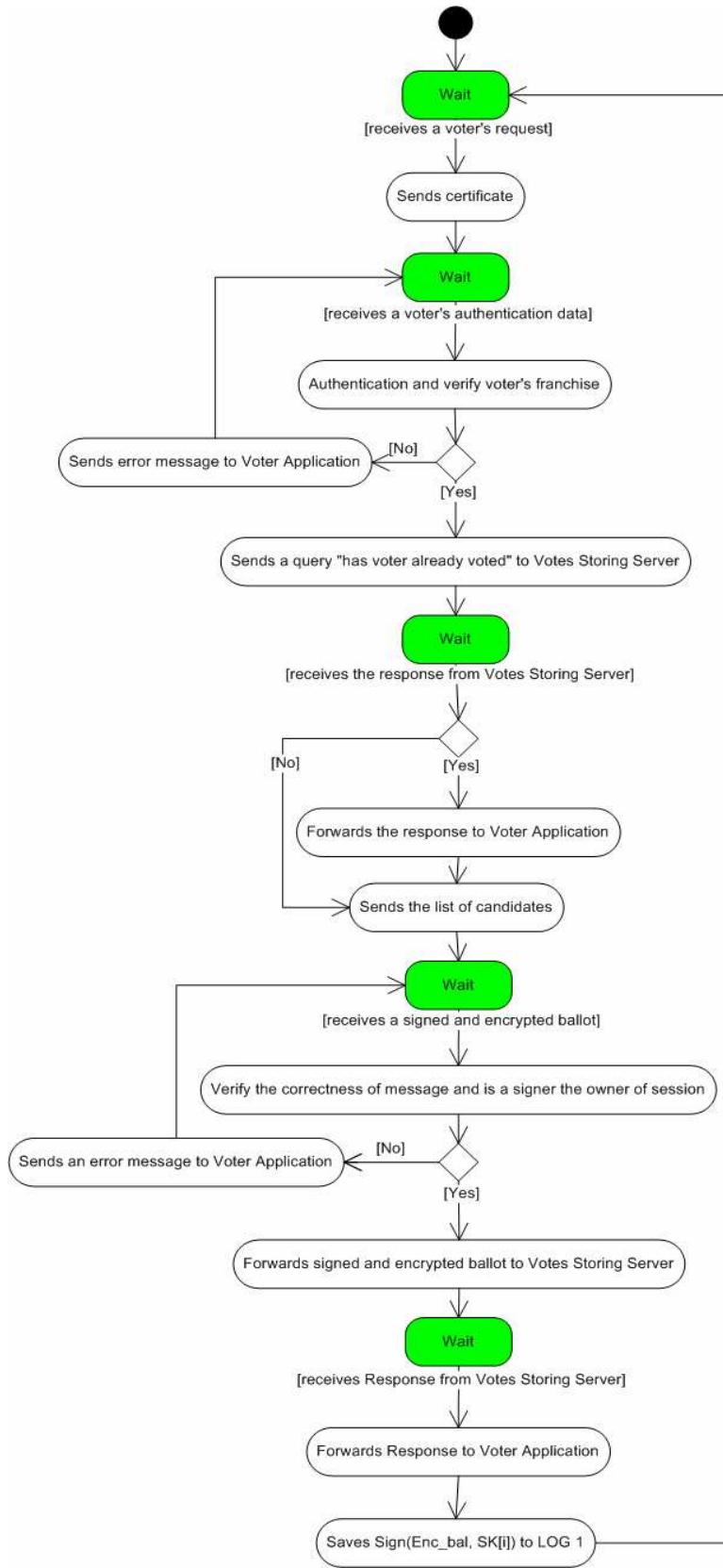


Figure 12. The processes of Network Server of the Estonian e-voting system.

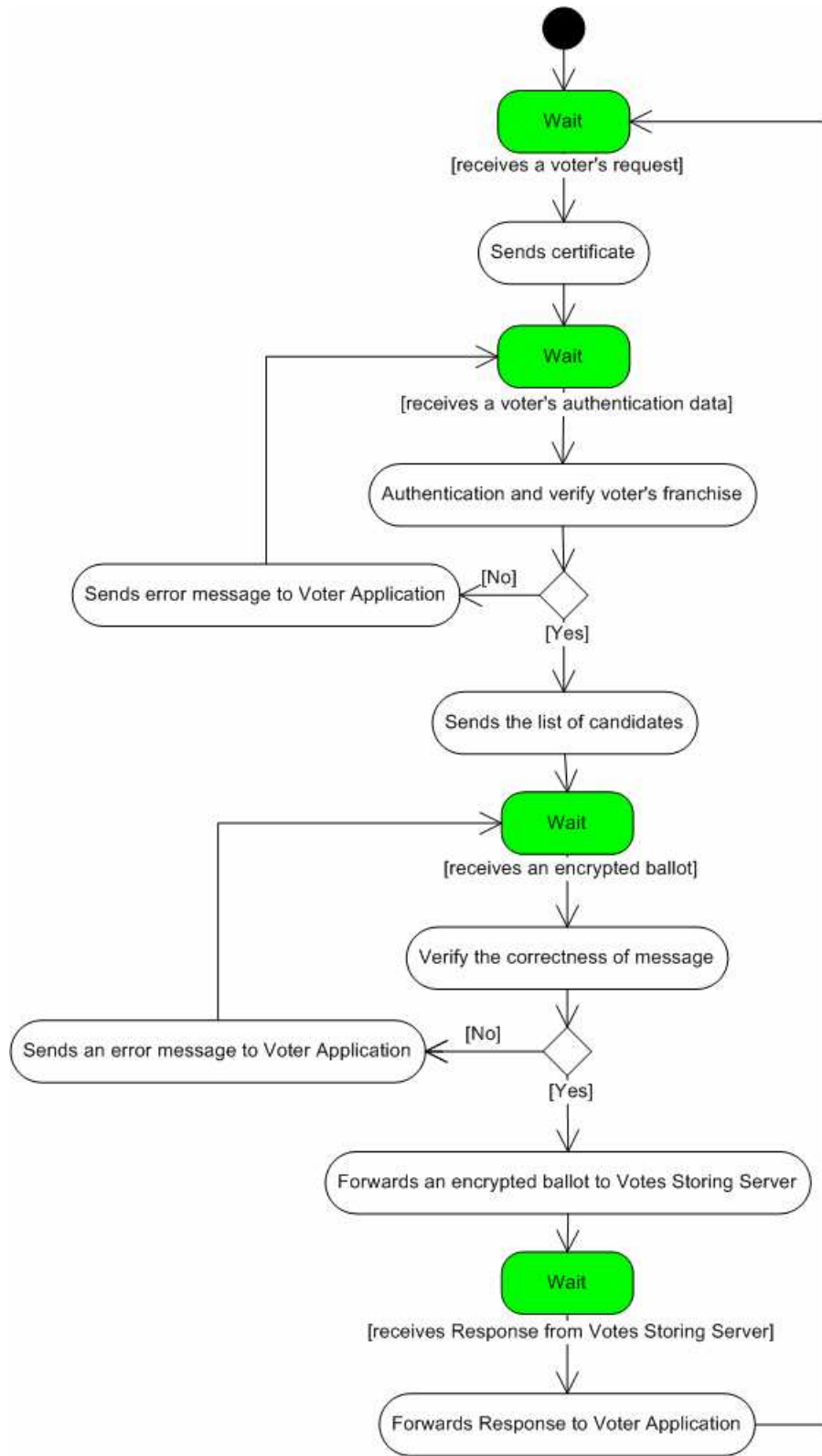


Figure 13. The processes of Network Server of SERVE.

Figure 15 depicts Votes Storing Server's processes in SERVE. Votes Storing Server receives voter's encrypted ballot from Network Server. In order to guarantee that only an eligible voter can vote the function of Votes Storing Server checks, if the voter has been registered and whether the voter has already voted. If the voter has not been registered, the server sends an error message to Network Server. If the voter has already voted, Votes Storing Server sends Voter Application an error message and cancels the received vote. Otherwise, Votes Storing Server saves the encrypted ballot and the personal data of the voter. The server replies each voter with a confirmation that the encrypted ballot was received. Next, Votes Storing Server decrypts the encrypted ballots by using the private key $SK[S]$ of SERVE. After that, server separates ballots and voters' personal data. Next, Votes Storing Server encrypts ballots with a public key of Votes Counting Server. Finally, Votes Storing Server saves the encrypted ballots and the list of voters in separate files, which are later transferred to Votes Counting Server.

Votes Counting Server of the Estonian e-voting system is offline. If the e-voting period is over, the server decrypts the encrypted ballots, verifies the format of ballots and counts the final tally. Votes Counting Server outputs the log file of incorrect ballots, the log file with counted ballots, and the final tally. The phase of votes' counting is repeatable. Figure 16 depicts the votes' counting processes of the Estonian e-voting system.

Votes Counting Server of SERVE downloads the list of voters and the encrypted ballots from Votes Storing Server periodically. After the end of the e-voting period, Votes Counting Server decrypts the encrypted ballots and counts the final tally. Votes Counting Server outputs the list of voters and the final tally. The output contains no information about the individual votes. The phase of votes' counting is repeatable. Figure 17 depicts the procedures of Votes Counting Server of the SERVE system.

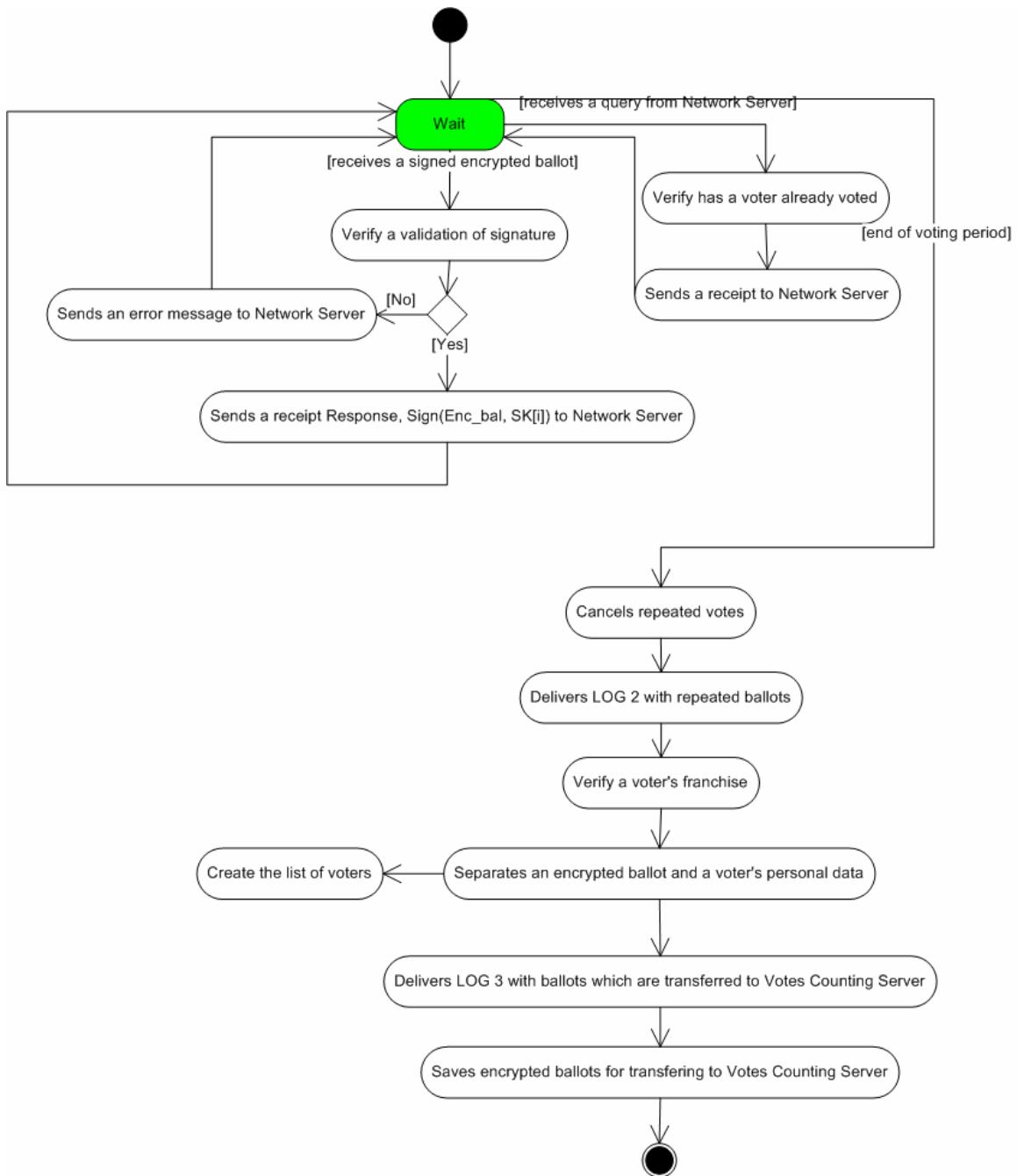


Figure 14. The processes of Votes Storing Server of the Estonian e-voting system.

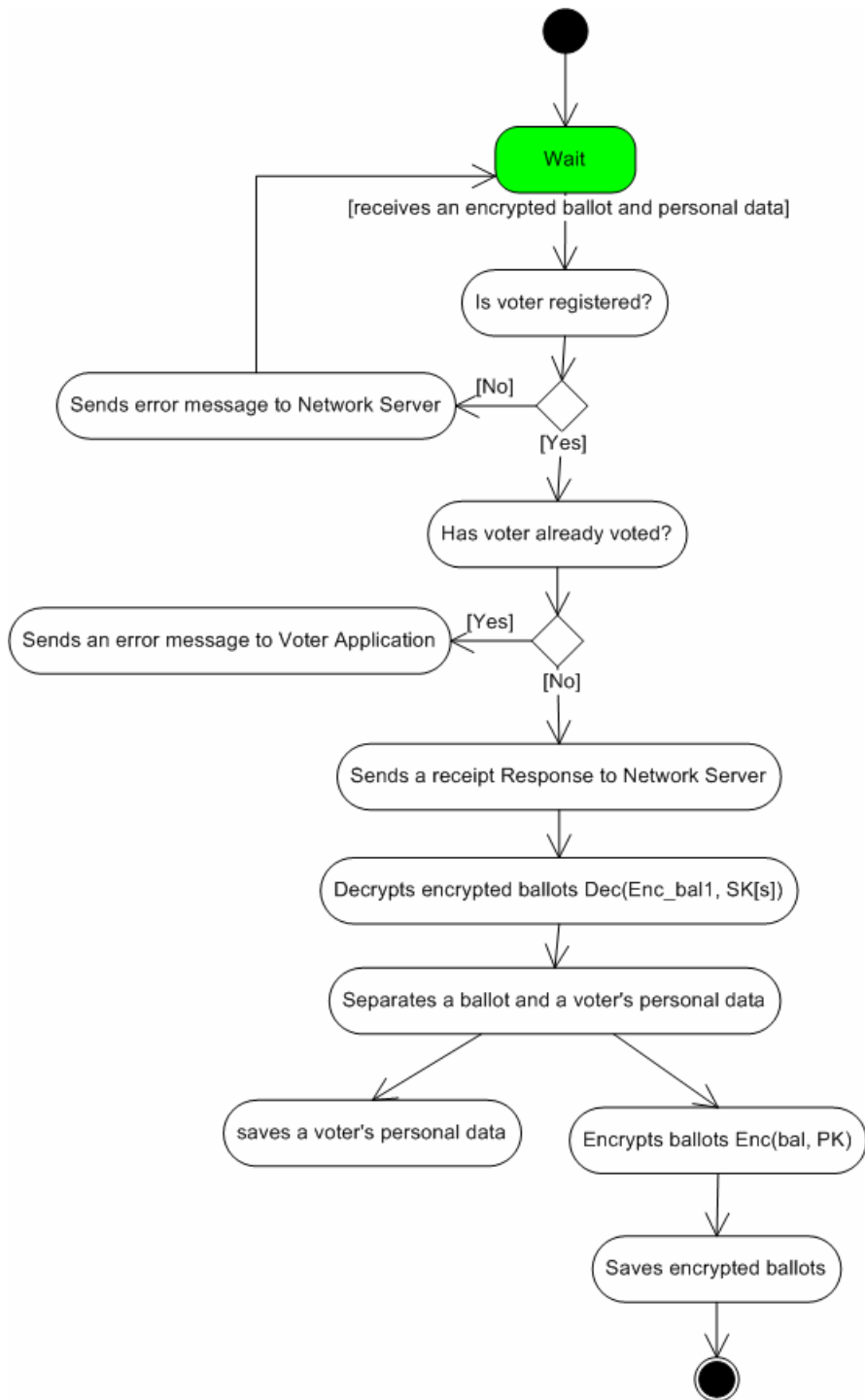


Figure 15. The processes of Votes Storing Server of the SERVE e-voting project.

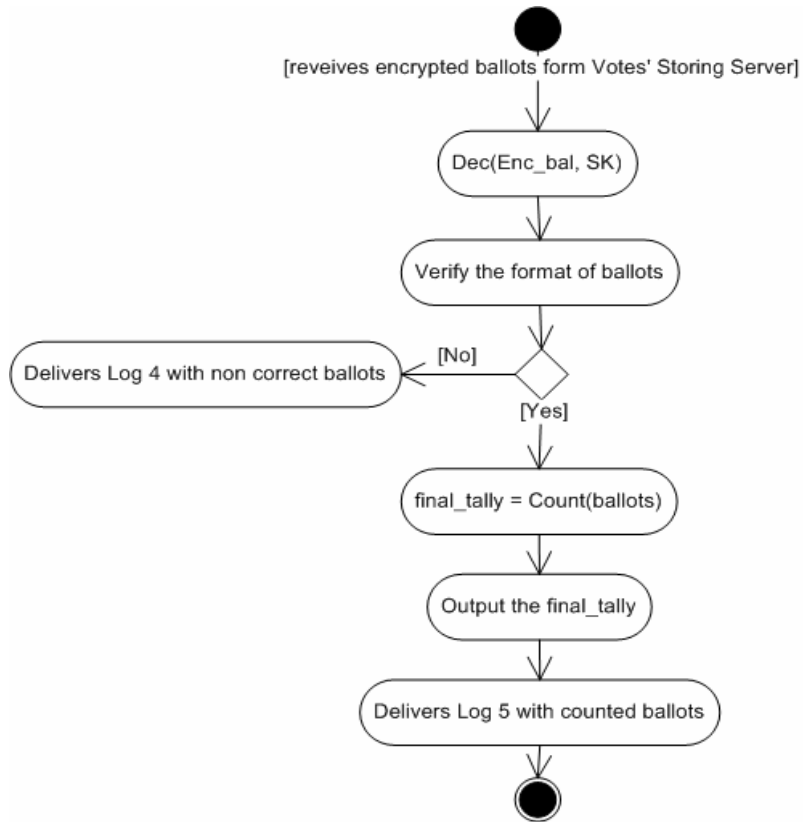


Figure 16. The processes of Votes Counting Server of the Estonian e-voting system.

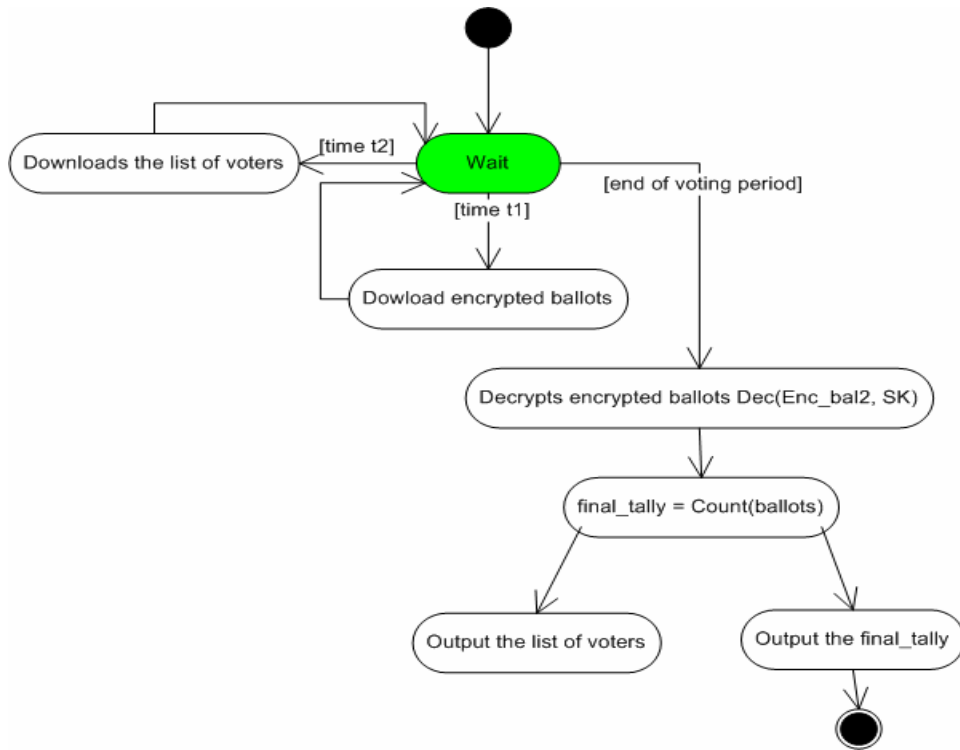


Figure 17. The processes of Votes Counting Server of SERVE.

4.3. Adversarial model and threats

In this section, we describe the activities of attackers through threats related to the components of e-voting system and specific e-voting attacks, which directly affect the desired properties of e-voting systems.

Small-scale attacks, which affect a small number of votes (for example 50 votes) are always possible in e-voting systems. Small-scale attacks do not affect the overall result of voting. Even if small-scale attacks take place, the e-voting is practically secure, considering the fact that the result of voting is not affected. Moreover, even traditional voting does not exclude single misbehaviors in elections. One of the main demands to e-voting is that it should be at least as secure as classical paper based voting. E-voting has more threats than classical voting, because e-voting should resist large-scale computer aided attacks. A large-scale attack may cause considerable changes in the final tally or reveal large numbers of votes. E-voting systems can never guarantee absolute security. To use appropriate methods it is possible to guarantee only practical security against large-scale attacks.

The question is how many votes should be changed or revealed in e-voting systems so that we may talk about a large-scale attack. For estimating this parameter we analyzed elections in Estonia and in the United States. We saw that the minimum average per cent of votes to affect the result of voting could be 4% [4, 15]. There has been exception in the presidential election in the United States in 2004. The difference between the rate of parties was only 0.0246. The number of target voters of the Estonian e-voting system and of SERVE was 1 million and 6 millions, respectively. Obviously, 100 computers are not enough to affect the result of voting. If 1,000 computers are infected, it would be possible to affect 0.1 per cent of the Estonian votes and 0.016 per cent of the United States votes. To summarize, we consider that infecting 1,000 computers is enough to have a large-scale attack in e-voting systems.

The two e-voting systems have the following components:

- a voter with Voter Application;
- Network Server;
- Votes Counting Server;
- Votes Storing Server.

There are millions voters with Voter Applications, many Network Servers, Votes Storing Servers and Votes Counting Servers. An attacker can attack these components or connections between them for affecting the input data of server.

In this analysis, we do not model an adversary as inside attacker. We assume that the team of e-voting has been created carefully and the team members are benevolent by themselves. However, we assume that the team members can be influenced from outside (for example, bribing) in order to affect an e-voting system maliciously. In this work we

do not analyze the crimes against person. Therefore, we do not consider that anybody is involved in attack by coercion or violence.

Next, we describe the behavioral model of an adversary. An adversary has the following activities:

1) To attack Votes Counting Server in order to affect the phase of votes' counting.

An adversary smuggles malicious code into the server for changing the functionality of Votes Counting Server, for example, for affecting the votes' counting.

2) To attack the connection between Votes Storing Server and Votes Counting Server in order to change the input of votes' counting phase.

For example, adversaries take control over the connection between Votes Storing Server and Votes Counting Server for adding votes or for deleting votes.

3) To attack Votes Storing Server in order to injure the phase of votes' saving and managing.

For example, adversaries smuggle malicious code into server for getting information on voters' ballots, for adding ballots or for deleting undesired votes.

4) To attack the votes' transferring process between Network Server and Votes Storing Server in order to affect the input of Votes Storing Server.

Adversaries "eavesdrop" the connection between servers for getting voters' encrypted ballots, for adding encrypted ballots or for deleting undesired votes.

5) To attack Network Server in order to affect the votes' reception.

Adversaries smuggle malicious code into Network Server for getting received encrypted ballots or for disfranchising undesired voters. Another widely known attack is a denial of service attack against web servers. A denial of service attack is an attempt to make Network Server's resource unavailable to voters.

6) To attack the connection between Voter Application and Network Server in order to affect the votes before they are received by Network Server.

The attack against the connection between Voter Application and Network Server might be a large-scale attack and could affect many votes before votes are received by the Network Server. For example, a Man in the Middle Attack for disfranchising votes or to find out how the voters had voted. A Man in the Middle Attack is the one by which the adversary interposes itself between the legitimate communicating parties and simulates parties to each other. The adversary communicates using two SSL sessions, one between oneself and the Voter Application and the other between oneself and Network Server.

7) To attack Voter Application in order to affect the votes' casting process.

The attack against Voter Application could be a large-scale attack to affect the voters voting. For example, launching of malicious codes in order to detect how the voters had voted or to change voters' choice, etc.

By using the adversarial model activities, an adversary could cause voting-specific attacks. Voting-specific attack is a large-scale attack, which means considerable changes in the final tally or a large scale of votes become revealed. If an e-voting system is secure in relation to voting-specific attacks, then we could deduce that it is practically secure. There are the following voting-specific attacks.

1) Large-scale votes' theft

The aim of the attack is to change votes or to give more votes for favorite candidates. If the e-voting system is not secure against the large-scale votes' thefts, then the adversary is able to cast ballots that participate in the computation of the final tally. Another threat is that voters are able to cast more than one vote, so that all votes are accepted final tally.

2) Large-scale disfranchisement of votes

It means that a large number of correctly encrypted ballots from eligible voters never reach Back-office. Attack could also selectively disfranchise eligible votes. The aim of disfranchisement of votes is to eliminate undesirable votes. The aim is not to cause the e-voting be failed, because it is not profitable for attackers. If elections are cancelled, then new elections will be organized.

3) Large-scale votes' buying and selling

It means that a large number of votes are sold. The aim of the attack is to increase the amount of votes for certain supported candidates.

4) Large-scale privacy violation

One of the main rights of voting is voters' privacy. The aim of the attack is to reveal how voters have voted. It can cause the violence, persecution in the society. Hence, democracy and freedom of word could be jeopardized.

If an e-voting system is secure against a large-scale votes' theft, then the following two security properties are justified:

- *Non-eligible voters are disfranchised*
- *Eligible voters are not able to cast two ballots that both participate in the computation of the final tally.*

The security property

- *Eligible voters are able to cast ballots that participate in the computation of the final tally*

is justified if an e-voting system is secure against a large-scale disfranchisement of votes. The security against a large-scale vote buying and selling and a large-scale privacy violation gives the justification of security property called

- *Votes are secret.*

4.4. Security assumptions and justifications

In this section, we point out widely believed security conditions, which are used as assumptions in our security analysis. All the security assumptions are justified for SERVE and the Estonian e-voting system. For the sake of simplicity, we do not consider security of cryptographic scheme. Security specialists have paid a lot of attention to the security of signature scheme, blind signature schemes, linking algorithms and other cryptographic algorithms that are used in voting systems.

Assumption I.: Signature schemes are secure.

The probability that an adversary not having access to the private key, creates a forged voter's signature so that the verification of the signature is true—is negligible. Cryptographers have worked a lot with this subject. Therefore, in this work we assume that the signature schemes are secure.

Assumption II. The encryption schemes are secure.

The probability to deduce the vote by knowing only the encrypted ballot—is negligible. In other words it is equal to guessing the vote without knowing the encrypted ballot.

Assumption III. Adversaries do not have access to the private keys SK , $SK[S]$.

Assumption III is justified if the key management in Back-office is sufficient to prevent the key compromise. This subject has had a lot of attention and methods' explanations in scientific literature. Hence, we assume that this assumption is secure in voting systems.

Assumption IV. Adversaries do not have a large-scale access to the voters' private keys $SK[i]$.

The average voter is unable to keep its own workstation secure enough to exclude possible abuses of the private key. For example, adversaries can steal the voter's passwords for activating the private key in smart cards. For stolen ID-card, adversaries are able to calculate passwords by using sufficient computational power. However, it is not possible to have a large-scale attack against voters' private keys. Assumption IV is justified by risk analysis in Subsection 4.6.3.1.

Assumption V. The phase of voters' registration is secure.

The Estonian e-voting system has the national Public Key Infrastructure (PKI), which is used also in electronic elections, instead of the phase of voters' registration. Estonian Public Key Infrastructure releases ID-cards with authentication and digital signature certificates to all citizens. The ID-cards are used as official identity documents. Many information systems in Estonia use authentication and digital signatures. Handmade signature is equal to digital signature by law. If the Estonian national PKI is not secure, then the loss is much bigger than only non-trustful e-voting result. Security specialists and scientists have paid a lot of attention to Public Key Infrastructure. In this work, we declare that sharing of authentication data and digital signature certification is secure in the Estonian e-voting system. Therefore, also the voters' registration phase is secure.

We assumed that Estonian national PKI authentication's certificate is secure. For fair comparison of the two systems, we assume that also the phase of voter's registration of SERVE is secure.

Assumption VI. The phase of votes' counting behaves exactly as specified.

All correctly composed and cast votes received by Votes Counting Server participate in the phase of votes' counting. Generally, Assumption VI is unjustified in practical voting systems, because the insider threats are even more common than the outsider threats. However, in this analysis the insider threats of votes' counting phase are not taken into account. Hence, we assume that the phase of votes' counting behaves exactly as specified.

Assumption VII. The independent log files system in the Estonian e-voting system is secure.

All the records in log files are cryptographically linked for guaranteeing integrity. We assume that linking algorithm is secure.

Assumption VIII. If considerable attacks are detected that cause misbehavior of e-voting or damage the reliability of e-voting or democracy then the e-voting is immediately stopped and the result of e-voting cancelled.

Both the Estonian e-voting system and the SERVE project have justified this property in the requirements of the systems. The decision to cancel the e-voting could cause the decision to terminate the election and to arrange new elections. The decisions will be made by court or Electoral Committee.

Assumption IX. Adversaries are unable to take a large-scale control over the voters' processes.

If adversaries take a large-scale control over the voters' processes, it is a loss of privacy and damage of democracy. The adversary could deduce how voters have voted, to change votes or disfranchise eligible voters etc. Assumption IX is justified by risk analysis in Subsection 4.6.3.2. It is justified for the Estonian e-voting system and the SERVE project.

Assumption X. Large-scale buying/selling of votes is possible only if there is a possibility to prove a vote.

If it is possible to prove who voted for whom, then votes buying services would spring up. In case the voter could not prove how the voter had voted, the votes' buying and selling is not a trustful and successful deal.

4.5. Security analysis in the modeled environment

In this section, we analyze the security of the two e-voting systems. We show that the Estonian e-voting system is practically secure and the SERVE system is not secure. To declare that an e-voting system is secure, it must be as secure as the traditional voting system. The reason is that traditional voting methods are considered to be practically secure and do not allow large-scale misbehavior. This means that e-voting must be secure against large-scale voting-specific attacks and the security properties of e-voting must be justified. We use the activity model of adversary and analyze whether the e-voting systems are secure against the following voting-specific attacks:

- large-scale votes' theft;
- large-scale disfranchisement of votes;
- large-scale votes' buying and selling;
- large-scale privacy violation.

If an e-voting system is secure against these voting-specific attacks, then the following properties are justified:

- Eligible voters are capable to cast ballots that participate in the computation of the final tally.
- Non-eligible voters are disfranchised.
- Eligible voters are not capable to cast two ballots that both participate in the computation of the final tally.
- Votes are secret.

Additionally, we give also the informal justification for other security properties of e-voting:

- It is possible for auditors to check if all correct cast ballots participated in the computation of the final tally.
- The result of election must be secret before the end of election.
- It must be possible to repeat the computation of the final tally.
- All valid votes are counted correctly and a system outputs the final tally.

If all the security properties of e-voting are justified then e-voting system is secure.

To give the justified analysis of e-voting security we use security assumptions (Section 4.4.) and the attack game risk analysis (Subsection 4.6.3.). The attack game risk analysis is based on the defined environment model (Subsections 4.6.1. and 4.6.2.).

4.5.1. The security analysis of the Estonian e-voting system

In this subsection, we analyze the security of the Estonian e-voting system. We study the security of the Estonian e-voting system based on the security assumptions and the model of environment. We go through all the voting-specific attacks and show that they are unlikely to happen. More precisely, they are unprofitable for rational attackers. Therefore, we conclude that all the security properties of e-voting are justified. In the following, we give the security analysis against the voting-specific attacks.

Large-scale votes' theft

If the Estonian e-voting system is secure against a large-scale votes' theft then following two security properties must be justified: *Non-eligible voters are disfranchised* and *Eligible voters are not capable to cast two ballots that both participate in the computation of the final tally*.

A large-scale votes' theft has three possibilities:

- votes are forged;
- non-eligible voters are able to vote;
- eligible voters vote more than once.

Figure 18 depicts the possible ways to have a large-scale votes' theft in the Estonian e-voting system.

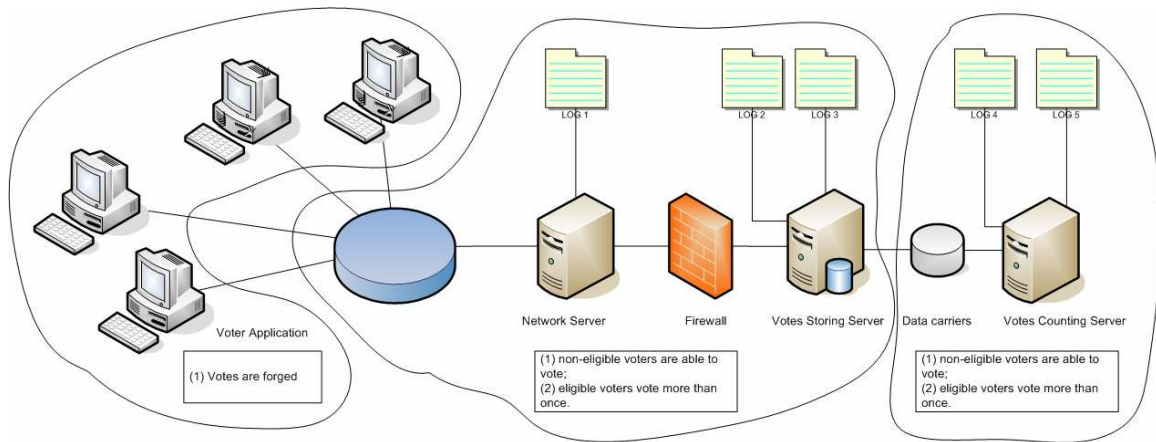


Figure 18. Possible ways for large-scale votes' theft in the Estonian e-voting system.

In this paragraph, we study a large-scale votes' theft e.g. the votes are forged. A voter generates a vote v and before it is encrypted an adversary changes without voter's knowledge the vote to v' , $v \neq v'$ (without voter's knowledge). For successful attack, the attacker needs a large-scale control over voters' processes for achieving the attack. Assumption IX states that adversaries are unable to take a large-scale control over voters' processes. Therefore, a large-scale forgery of votes by getting control over Voter Application processes is unlikely.

In Network Server, Votes Storing Server and in the connections between Voter Application, Network Server and Votes Storing Server a large-scale votes' theft is possible when non-eligible voters are able to cast votes or when eligible voters are able to vote more than once.

In the following, we study the threat that non-eligible voters are able to cast votes. This means that an adversary is able to create at least 1,000 new correctly verifying signed encrypted ballots for a vote in the name of voters. For creating a large amount of signed

and encrypted ballots in the name of voters, an adversary needs access to a large number of voters' private signature keys. Assumption IV states that adversaries do not have a large-scale access to the private keys of voters. Additionally, if the adversary is able (without having access to voters' private keys) to cast a vote then it is possible to construct an adversary that breaks the signature scheme. It is impossible by Assumption I. Therefore, under the assumptions we made, non-eligible voters are not able to cast large numbers of correct ballots.

Large scale votes' theft is also possible if a large number of eligible voters are able to vote twice or if eligible voters are able to cast large numbers of votes. In the Estonian e-voting system, voters are able to cast more than one ballot, but only the last one is counted. Votes Storing Server cancels multiple votes. In case an adversary has access to the server and modifies the multiple votes' canceling process, then eligible voters would be able to vote many times. We analyze this attack by using attack game risk analysis in Subsection 4.6.3.5. Risk analysis shows that this attack is not profitable.

A large-scale votes' theft against Votes Counting Server or the connection between Votes Storing Server and Votes Counting Server is an inside attack. The connection between servers is a data transfer by using data carriers i.e. data transfer is offline. Votes Counting Server of the Estonian e-voting system is not connected to the Internet. The phase of votes' counting is secure by using Assumption VI. Let us assume that before the phase of votes' counting an attack against Votes Counting Server or data carriers is possible. The encrypted ballots are transferred to Votes Counting Server. Therefore, for adding votes to Votes Counting Server, an adversary does not need voters' private signature keys. The attack needs a program for creating encrypted ballots and a possibility to add votes into the data carrier or to Votes Counting Server. In Votes Storing Server there is a log file LOG3 which consist of all encrypted ballots and voters' personal data. Assumption VII declares that the independent log file system in the Estonian e-voting system is secure. For successful attack against Votes Counting Server or data carriers, the attacker should attack also Votes Storing Server for adding encrypted ballots and voters' personal data to LOG3. This means that the attack consists of affecting the two e-voting servers. In Subsection 4.6.3.7 there is an attack tree analysis for this attack. The probability to succeed the attack is 0.01 and it is unprofitable. Moreover, to change the log file in Votes Storing Server, attackers should affect the log file also in Network Server. Obviously, an attack against three e-voting servers is unprofitable. Therefore, even when the attacker is able to get access to offline Votes Counting Server or data carriers, a large-scale votes' theft is unlikely.

Conclusions, in the Estonian e-voting system: (1) *Non-eligible voters are disfranchised* and (2) *Eligible voters are not capable to cast two ballots that both participate in the computation of the final tally* are completed.

Large-scale disfranchisement of votes

If the Estonian e-voting system is secure against a large-scale disfranchisement of votes then the security property *Eligible voters are capable to cast ballots that participate in the computation of the final tally* holds.

The aim of disfranchisement of votes is to eliminate undesired votes. The first possibility is to attack the phase of voters' registration. In Estonia the e-voting's certificates of authentication and digital signatures are distributed among voters by using the national Public Key Infrastructure. Assumption V says that the Estonian national Public Key Infrastructure is secure. Therefore, the attack against the phase of voters' registration is impossible by Assumption V.

There are the following possibilities to achieve a large-scale disfranchisement by attacks on e-voting system components:

- undesired votes are eliminated so that voters do not get a positive response from e-voting system;
- undesired votes are eliminated so that voters get a positive response from e-voting system;
- undesired voters' votes are eliminated and voters get a positive response from the e-voting system.

Figure 19 depicts the possible ways to disfranchise votes in the Estonian e-voting system.

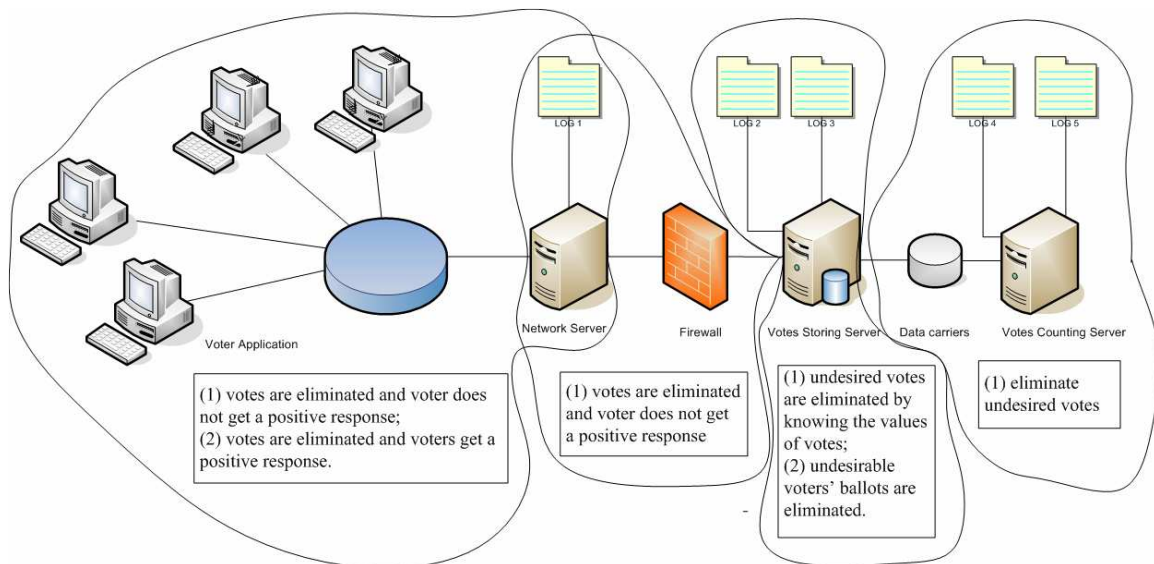


Figure 19. Possible ways of disfranchisement in the Estonian e-voting system.

In the following, we consider the cases when attacks are performed against Voter Application, Network Server or the connection between them. For example, a denial of service attack against Network Servers disfranchises voters so that signed encrypted ballots from eligible voters never reach Back-office. In the case when voters are not able to cast votes they will inform Electoral Committee. Therefore, by using Assumption VIII the attack is unlikely.

The second possibility is that a vote is eliminated, but the voter gets still a confirmation about accepted vote. In this case, Voter Application is injured for converting the error message from Network Server. We analyze this threat in Subsection 4.6.3.6 by using attack game risk analysis. The attack tree analysis shows proofs that this attack is not profitable.

Thirdly, if the attack is against Network Server or the connection between Network Server and Votes Storing Server, then voters are able to cast votes but they will get error messages. Voters inform Electoral Committee about misbehaviors and Electoral Committee terminates the e-voting process by using Assumption VIII.

A dedicated attack against Votes Storing Server is an attack against the pairs of voters' data and signed encrypted ballots, because the attacker needs information about voters or votes in order to eliminate undesired votes. If attackers want to eliminate votes based on values of votes then they should be able to decrypt ballots. Assumption II says that adversaries do not have access to the private decryption key SK . Therefore, adversaries could not eliminate undesired votes this way. To deduce the values of votes without having private keys, the adversary needs to know the random numbers inside the ballots. Therefore, attackers need a control over voters' voting processes. Assumption IX states that adversaries are not able to take a large-scale control over the voters' processes. Hence, the attack to eliminate undesired encrypted ballots is not possible.

Another possibility to achieve the aim of attack is to use undesired voters' list and delete their votes in Votes Storing Server. The Estonian e-voting system has an independent log files system, which guarantees the integrity of the e-voting. In case some votes are just deleted without modifying the log files, the sum of logs is not verifiable. Hence, the reliability of e-voting is damaged and the e-voting is terminated and will be cancelled by Assumption VIII.

Let us assume that the adversary eliminates voters' votes in Votes Storing Server so that deleted ballots never reach to the log file system. If adversary attacks Votes Storing Server in order to eliminate votes, then adversary should also attack Network Server for catching the votes data written in LOG1. We will analyze the threat in Subsection 4.6.3.7. The result of the attack tree analysis confirms that this attack is unlikely.

Finally, we analyze an attack against Votes Counting Server or against the data carriers which are used to transfer encrypted ballots to Votes Counting Server at the end of voting. To delete votes in purpose, adversaries must know the value of votes; therefore they need an access to the private key of e-voting. Assumption III states that adversaries do not have access to the private keys of e-voting. If adversaries know the encrypted ballots of undesired votes', they may compare these with the encrypted ballots in Votes Counting Server and deduce, which votes are undesired. In order to decide which ballot is undesired, an adversary needs a large-scale control over voters' e-voting processes and we have a contradiction with Assumption IX. Moreover, adversaries need to get control over the log file system and therefore they have to attack Network Server and Votes

Storing Server. As shown in the previous paragraphs and Assumptions III and IX the attack against Votes Counting Server or data carrier is not successful.

To conclude, the verification of log files is highly important for achieving the completeness of security property *Eligible voters are capable to cast ballots that participate in the computation of the final tally*. If independent auditors do not verify the integrity of the independent log file system then the Estonian e-voting system would not be secure enough.

Large-scale votes' buying and selling

Assumption X says that large-scale votes' buying and selling is possible only if voters are able to prove how they voted. The analysis of possibilities to have a large-scale votes' buying attack is described in Subsection 4.6.3.4. To summarize, in the Estonian e-voting system it is unlikely to have a large-scale votes buying. It gives us the security property that voters are not able to prove for whom they voted.

Large-scale privacy violation

A large-scale privacy violation means that adversaries are able to check reliably how voters voted. For analyzing the privacy violation threat we created an attack tree in Subsection 4.6.3.3. The attack tree analysis for the Estonian e-voting systems justifies that a large-scale privacy violation is unlikely.

The justifications of threats: *Large-scale vote buying and selling* and *Large-scale privacy violation* are unlikely and it gives the justification of security property *Votes are secret*.

The justification of other security properties

In the following, we analyze other security properties in the Estonian e-voting system. The security property *All valid votes are counted correctly and system outputs the final tally* is justified with Assumption VI, which says that the phase of votes' counting behaves exactly specified.

There is a security property named as *It is possible for auditors to check if all correct cast ballots participated in the computation of the final tally*. The Estonian e-voting system has an independent log files system, which enables us to check that received votes participate in the computation of the final tally or are cancelled in the votes' managing phase. Assumption VII states that the independent log files system in the Estonian e-voting system is secure.

Next, we analyze the property *The result of election must be secret before the end of election*. In the Estonian e-voting system all ballots are stored and transferred in encrypted form all through in the e-voting system. Only the processing in Votes Counting Server can decrypt encrypted ballots. For counting encrypted ballots, adversaries need to decrypt ballots with the private key of e-voting. By Assumption III, the private key management is secure. Additionally, the transmission of encrypted ballots to Votes Counting Server takes place after the end of the voting period. Hence, no one knows the tally before the end of voting. Another theoretical possibility is to obtain a

control over all voters' processes. Assumption IX says that adversaries are not able to take a large-scale control over the voters' processes. To summarize, the security property *The result of election must be secret before the end of election* is justified in the Estonian e-voting system.

The last security property says that *It must be possible to repeat the computation of the final tally*. Assumption VI states that the phase of votes' counting behaves exactly as specified. In the design of the Estonian e-voting system it is claimed that the computation of the final tally must be repeatable. Therefore, the security property is justified for the Estonian e-voting system.

Conclusion

To consider, security assumptions and modeled environment characteristics, all the defined security properties have been justified for the Estonian e-voting system. Therefore, the Estonian e-voting system is secure in the defined environment model.

4.5.2. The security analysis of the SERVE system

In this subsection, we analyze the security of the SERVE system. We study whether SERVE is practically secure. It is studied that if the e-voting system is not secure against voting-specific attacks and all the security properties are not justified then the e-voting system is not practically secure. The SERVE system is not practically secure.

Large-scale votes' theft

If SERVE is secure against a large-scale votes' theft then the following two security properties are justified: *Non-eligible voters are disfranchised* and *Eligible voters are not capable to cast two ballots that both participate in the computation of the final tally*.

A large-scale votes' theft has three possibilities:

- votes are forged;
- non-eligible voters are able to vote;
- eligible voters vote more than once;
- adversary multiply the received votes.

Figure 20 depicts possible ways to a large-scale votes' theft in the SERVE system.

In this paragraph we study how a large number of votes could be forged in the injured Voter Application. In the process of casting a vote, before the encrypting of vote, an adversary is able to change the vote without voter's knowledge. To achieve it, the adversary needs the control over voters' e-voting processes. Assumption IX states that adversaries are unable to take a large-scale control over the voters' processes.

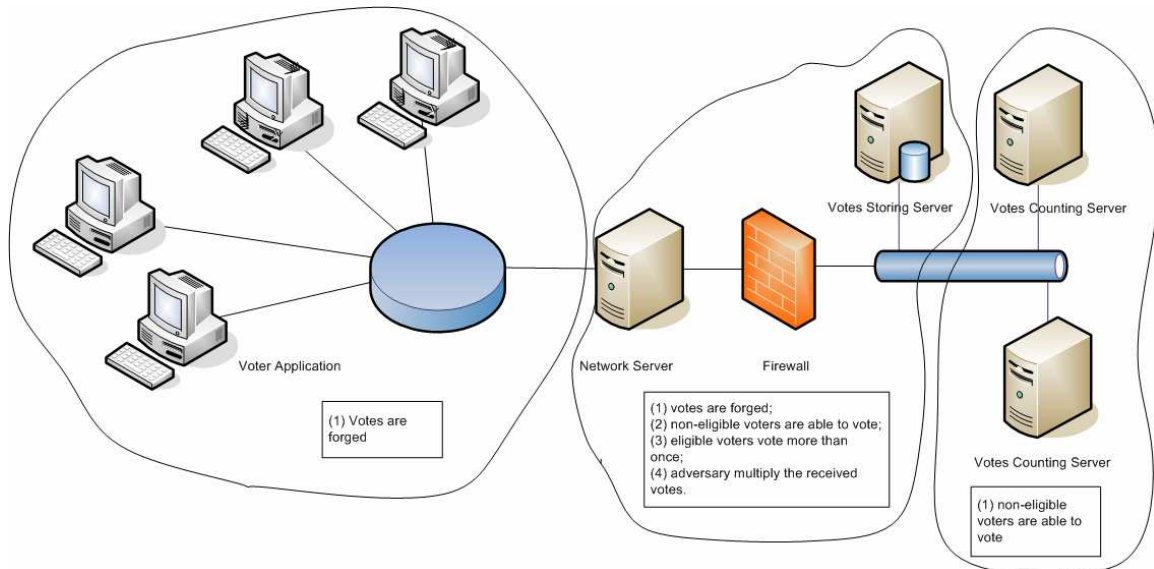


Figure 20. Possible ways of votes' theft in SERVE.

Let us assume that adversaries want to attack the connection between Voter Application and Network Server for changing votes. It is possible to create a Man in the Middle Attack so that the adversary could act as an SSL gateway, forwarding voters' personal data and encrypted ballot between Voter Application and Network Server. The attacker is able to see the content of traffic and to create new encrypted ballots by using the public key of e-voting and replace the encrypted ballots in the pair of voting data. The e-voting system checks, if the voter is eligible and if the encrypted ballot is in the correct format. Voters receive positive *Response* from the e-voting system. *Response* does not contain any information about the received encrypted ballot. We analyze the efficiency of *The attack against the connection between Voter Application and Network Server for changing the ballot* in Subsection 4.6.3.8 by using risk analysis. The attack game shows that this attack is unlikely.

If adversaries attack Network Server or the connection between Network Server and Votes Storing Server for the purpose of casting votes, then they should attack also Votes Storing Server, because the franchise control¹ processes are performed in Votes Storing Server. Therefore, in the following we analyze the attacks against Votes Storing Server. If such attacks are impossible, then it is also impossible to attack Network Server or the connection between Network Server and Votes Storing Server.

First, we analyze the case if eligible voters vote more than once. Votes Storing Server of SERVE checks whether voters have already voted. Therefore, for the attack to be successful, an adversary needs access to Votes Storing Server in order to affect the processes running in the server. We analyze the profitability of such attacks by using the attack game in Subsection 4.6.3.5. The result of analysis shows that this kind of attack is not profitable.

¹ The franchise control checks (1) whether voter has franchise and (2) whether voter has already voted.

Second, we analyze the attacks when adversaries take control over Votes Storing Server. For example, non-eligible voters would be able to cast correctly verifying encrypted ballots if they get control over Votes Storing Server for passing the franchise control and for preventing their names being saved into the list of voters. In Subsection 4.6.3.9 we analyze the attack tree *Control over processes of Votes Storing Server of SERVE*. The computation of attack tree shows that the SERVE system is insecure against smuggling the encrypted ballots to Votes Storing Server.

Votes Storing Server decrypts encrypted ballots. In case an adversary has control over these processes, it is possible to change votes before the votes are encrypted again. This serious large-scale votes' theft is possible if adversaries have gained control over Votes Storing Server. Risk analysis for this attack is presented in Subsection 4.6.3.9. It shows that the attack is likely.

The third possibility to add ballots is to copy the ballots that already exist in Votes Storing Server. To conclude, a large-scale votes' theft by attacking Votes Storing Server is possible in the SERVE system.

Subsequently we analyze the large-scale votes' theft attacks against Votes Counting Server or against the link between Votes Storing Server and Votes Counting Server. Votes Counting Servers are online servers that download the list of voters and encrypted ballots from Votes Storing Server. In order to add the encrypted ballots to Votes Counting Server, the adversary needs to know the public key of Votes Counting Server. We assume that the public key of Votes Counting Server is not published to everyone and the adversary should attack Votes Storing Server for getting the public key. It means an attack against two e-voting servers. We analyze this attack by using risk analysis in Subsection 4.6.3.10. The result of the attack game shows that the attack is not profitable. Therefore, a large-scale votes' theft in Votes Counting Server is unlikely.

To summarize, a large-scale votes' theft is possible in the SERVE system. The adversaries are able to add votes and change the votes so that they participate in the computation of the final tally. The main problem in the SERVE system is that ballots are decrypted in Votes Storing Server and they are accessible by bribing a server administrator. Additionally, there is no a well-developed independent log file system in SERVE for guaranteeing the integrity of processes of e-voting processes. The security properties named *Non-eligible voters are disfranchised* and *Eligible voters are not capable to cast two ballots that both participate in the computation of the final tally* are not justified in SERVE.

Large-scale disfranchisement of votes

If SERVE is secure against a large-scale disfranchisement of votes then the security property *Eligible voters are capable to cast ballots that participate in the computation of the final tally* holds.

The aim of disfranchisement of votes is to eliminate the undesired votes. The first possibility is to attack the phase of voters' registration. Assumption V declares that the

phase of voters' registration in the SERVE system is secure. Therefore, this attack is not taken into account in this work.

In the following, we study three possible ways to disfranchise votes:

- undesired votes are eliminated so that voters do not get a positive response from e-voting system;
- undesired votes are eliminated so that voters get a positive response from e-voting system;
- undesired voters' votes are eliminated and voters get a positive response from the e-voting system.

Figure 21 depicts possible ways of votes' disfranchisement in the SERVE system.

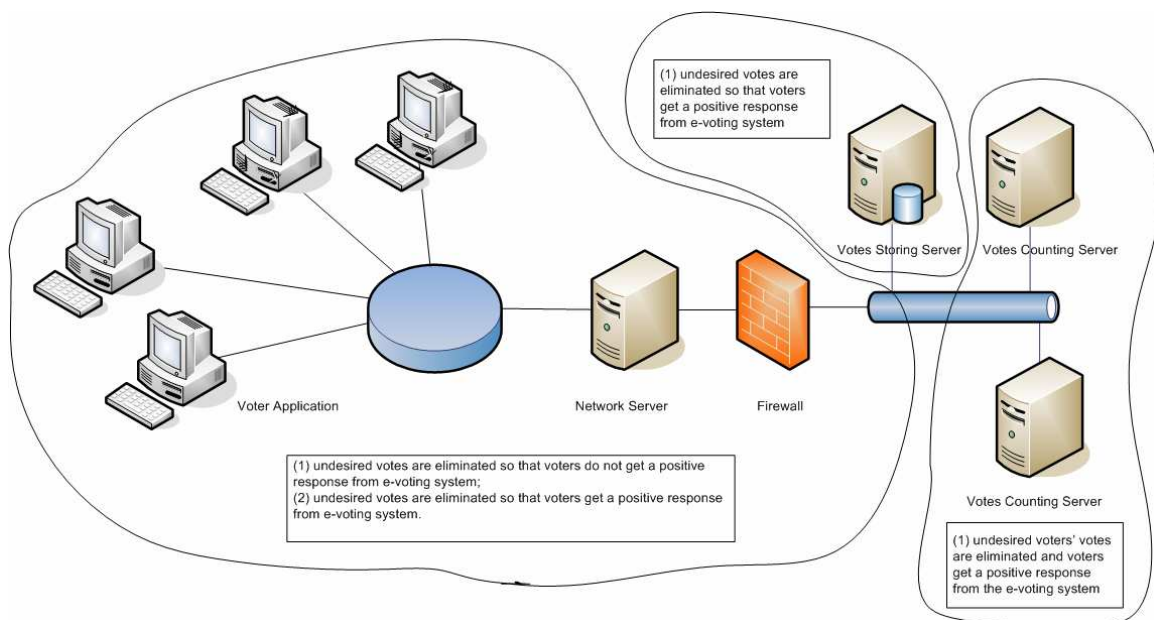


Figure 21. Possible ways of disfranchisement in SERVE.

Next, we analyze disfranchisement attacks against Voter Application, Network Server, and connections between Voter Application, Network Server and Votes Storing Server. A classical threat is the Man in the Middle Attack, when voters connect to a forged server. If votes are disfranchised and they do not get the *Response* message then a fraction of voters who will notice this misbehavior will inform the Electoral Committee and e-voting will be cancelled by Assumption VIII. Another possibility is that a forged Network Server sends a malicious ActiveX code that creates a forged *Response* message. Such attack is possible, if voters do not notice that Voter Application is injured. We analyze this threat by using risk analysis in Subsection 4.6.3.6. The attack tree analysis shows that this attack is not profitable.

If an adversary damages Network Server so that voters are not able to create connections with Network Server (e.g. with a denial of service attack) then voters inform Electoral Committee and e-voting is cancelled by using Assumption VIII.

In the following, we analyze attacks against Votes Storing Server. If an adversary has control over Votes Storing Server, then it is possible to delete undesired ballots. The attack tree analysis in Subsection 4.6.3.9 shows that adversaries are able to take control over Votes Storing Server and, it is possible to disfranchise voters in Votes Storing Server of SERVE.

Next, we analyze attacks against Votes Counting Server. The connections between Votes Storing Server and Votes Counting Server are online. Votes Counting Server pulls encrypted ballots and the list of voters when it updates its database. If the adversary is able to diminish the time between two updates, then, with high probability, the adversary is able to associate encrypted ballots with voters' names. Hence, it is possible to eliminate encrypted ballots of undesired voters. For a successful attack it is sufficient to attack Votes Counting Server only. Therefore the attack has analogous attack game with *Control over processes of Votes Storing Server of SERVE* in Subsection 4.6.3.9. The attack game analysis shows that a large-scale disfranchisement is possible in the SERVE system by attacking Votes Counting Server.

Let us assume that the system compares the number of voters with the number of votes. In this case the adversary has to delete the names of voters in the list of voters in Votes Counting Server and in Votes Storing Server. This would mean an attack against two e-voting servers. The attack tree *Eliminating votes in two e-voting servers* represents also attack against these two servers in Subsection 4.6.3.7. The result of the attack game shows that such attack is not profitable.

In conclusion, a large-scale disfranchisement of votes is possible in SERVE. The main threat is the possibility to gain control over Votes Storing Server. Moreover, online Votes Counting Server, which downloads encrypted ballots and voters' list during the e-voting period, represents also threats to disfranchisement of votes. The security property *Eligible voters are capable to cast ballots that participate in the computation of the final tally* does not hold in SERVE.

Large-scale votes' buying and selling

A large-scale votes' buying and selling is possible only, if voters are able to prove how they voted by using Assumption X. The attack game risk analysis for large-scale votes' buying is described in Subsection 4.6.3.4. It turns out that large-scale votes' buying and selling is possible in SERVE.

Large-scale privacy violation

A large-scale privacy violation means that adversaries are able to deduce reliably how the voters voted. For analyzing the privacy violation threat we created an attack tree in Subsection 4.6.3.3. The analysis shows that a large-scale privacy violation is possible in SERVE.

In conclusion, the security property *Votes are secret* is not justified in SERVE.

The justification of other security properties

In the following, we analyze the other security properties in SERVE. The security property *All valid votes are counted correctly and system outputs the final tally* is justified by Assumption VI, which says that the phase of votes' counting behaves exactly as specified.

The property *It is possible for auditors to check if all correct cast ballots participated in the computation of the final tally* does not hold in SERVE, because there are no independent audit trails.

Next, we analyze the property *The result of election must be secret before the end of election*. The phase of votes' counting in Votes Counting Server is secure by using Assumption VI. To count encrypted ballots non-officially, adversaries either need to decrypt the ballots with the private key of e-voting or to break the encryption scheme. It is impossible by Assumption II and III. Another theoretical possibility is to have a control over all voters' processes for recording how voters voted. This is impossible by Assumption IX.

However, in Votes Storing Server ballots are not encrypted. To count the non-official final tally before the end of election it is sufficient to take control over Votes Storing Server. The attack game analysis in Subsection 4.6.3.9 shows that this attack is possible and therefore the security property *The result of election must be secret before the end of election* does not hold in SERVE.

The last security property says that *It must be possible to repeat the computation of the final tally*. Assumption VI states that the phase of votes' counting behaves exactly as specified. In the design of SERVE, it is required that the computation of the final tally must be repeatable. Therefore, the security property is justified for SERVE.

Conclusion

To summarize, in SERVE the following security properties are not justified:

- Non-eligible voters are disfranchised;
- Eligible voters are not capable to cast two ballots that both participate in the computation of the final tally;
- Eligible voters are capable to cast ballots that participate in the computation of the final tally;
- Votes are secret;
- It is possible for auditors to check if all correct cast ballots participated in the computation of the final tally;
- The result of election must be secret before the end of election.

Therefore, the SERVE system is not practically secure in our environment model.

4.6. Attack game risk analysis in the modeled environment

The security analysis uses really strict security assumptions and threats' potentiality in the previous sections. It is impossible to prove them, but it is possible to justify them empirically by using the attack game risk analysis. To analyze the empirical security in e-voting systems we create an e-voting environment model as close as possible to the real world model. We will justify the security assumptions and analyze the e-voting threats in the environment model by using attack game risk analysis.

4.6.1. General characteristic for the environment

A meaningful comparison of two systems must be based on the same benchmarks. Hence, we create the same environment for the both e-voting systems. It is clear, that the environment of the Estonian e-voting system and SERVE are different in real life, for example, in number of voters. Moreover, it is even hard to describe these environments adequately and give the real characteristics of environment. For example, it is complicated to estimate what is the probability of catching and convicting attackers, if voters deliberately create connections to an actively compromised voting server. For adequately specifying the characteristics of an environment for e-voting systems, it is necessary to study the purposes of attacks, success probabilities of attacks, detection probabilities of attacks, awareness of computers' users, punishments for cyber crimes, etc. In order to make rational decisions about practical security of e-voting systems we have to know these parameters of environment with sufficient accuracy. If we are not able to estimate these parameters with sufficient accuracy, this would also mean that we do not know whether these systems are practically secure. This work also indicates the necessity of future work for obtaining better estimates of these parameters.

We create a hypothetical environment for analyzing security of the two e-voting systems. We try to estimate parameters of the environment as close as possible to real society. For estimating the parameters we have used information from Internet, from research papers, interviews with specialists and typical attacking scenarios. We assume that typical attackers do not make extensive social research for getting information whether it is profitable to attack. Quite probably, a gain-oriented attacker would analyze the same information from Internet, gather opinions from friends and make decisions intuitively. Definitely, this hypothetical environment is not perfect, but it is the best we know for comparing the security of the two e-voting systems.

Voting is a main right in democratic society. It is the substructure for democracy. Hence, we consider the environment to be a well-developed democratic society. In well-developed society the aim of crime determines the seriousness of crime. Therefore, if the aim of the crime is to affect the result of voting then the crime is viewed as serious crime against the society no matter how the crime was performed. Moreover, the punishment for crime is at least dispossession of the gain of the crime. Therefore, we may assume that the punishment is always at least as large as gains,

$$Gains \leq Penalties .$$

In this work, we consider the limit that *Penalties* are equal with *Gains*. This gives us the simplicity for calculating nodes of attack trees:

$$\begin{aligned} & -Costs + p \cdot (Gains - q \cdot Pealties) - (1 - p) \cdot q_- \cdot Pealties_- \\ & \leq -Costs + Gains \cdot (p \cdot (1 - q) - (1 - p) \cdot q_-) . \end{aligned}$$

Through elections people choose their leaders, they give the power to their favorites. Parties spend lots of money for campaigns of election. Probably, the gain is even bigger. In Estonia, parties spend about 20 million [16] Estonian kroons for a campaign of election. We assume that *Gains* of affecting the result of election is at least 5 times bigger, so 100 million monetary units.

By the data available in Internet the price of malicious code is about \$50. A person can be bribed for about half a million monetary units [12]. We assume that attackers are rationally and economically thinking. Hence, to calculate the cost of attack, we focus on self-cost. Additionally, in case the attackers need 100 computers for attacking the system, they create a bot network with 100 computers. Even, if we assume that the price of developing a forged Network Server is 2 million monetary units, the expenses of attacks are small compared to the gain of election.

To summarize, considering the specificity of elections, *Costs* are always much smaller than *Gains*. Hence, the value of *Costs* does not affect attacker's final decision to attack an e-voting system or not. Therefore, we may even assume that $Costs \approx 0$. If the e-voting system is secure when $Costs = 0$ then the system is also secure when $Costs > 0$.

We know that the attack is unprofitable when

$$Outcome < 0 .$$

Therefore, under the simplification we made the formula

$$Outcome = -Costs + Gains \cdot (p \cdot (1 - q) - (1 - p) \cdot q_-)$$

is always negative when

$$p \cdot (1 - q) - (1 - p) \cdot q_- < 0 .$$

To summarize, considering the particularity of e-voting we may estimate only three parameters p , q and q_- of attack game for estimating the profitability of attacks.

4.6.2. Environment's characteristics

In this subsection, we bring out the environment's characteristics for comparing two e-voting systems by risk analysis. If the environment characteristic is deduced from interviews or from security researches or from some another source it has references to that. Otherwise, the environment characteristic is a logical derivation from Internet information.

- 1) About 1 per cent of voters will notice and reveal that their computers are infected.**

Thereby, the success probability of attacking large number (1000) of voters' workstations is $p \leq 0.99^{1000}$.

- 2) At least 1% of electronic voters verify the authenticity of the Network Server certificate, the signature of ActiveX component and wait for the confirmation of e-voting, i.e. $P, X, Q \approx 1$.**

We assume that if a voter is aware of the need to verify the certificate of Network Server, then he is also aware of the need to verify the signature of ActiveX component and to wait for the confirmation about accepted vote.

The probability that 1,000 voters do not verify the certificate of Network Server or the signature of ActiveX component or do not wait for the signed confirmation from the e-voting server is $p \leq 0.99^{1000}$.

Such a modeling of voters is somewhat idealistic, because all voters are assumed to have the same values of P and Q and X . In practice, the attacker may estimate these values by guessing the technical skills and carefulness of the voters and then to attack those with lower skill and careless.

- 3) The probability to discover the misbehaviors of software is 0.3 by using code review and auditing.**
- 4) 0.33 of people are corrupted at least with 0.5 millions monetary units [12].**
- 5) Bribing, which has caused damage to somebody is detected with probability $q \leq 0.3$ [12].**
- 6) The probability to exploit the bug in operational system or hardware and get the access to system is 0.002.**

We assume that a bug in an operational system or in hardware is discovered once per 3 years. Within 2 days there is a virus to exploit this bug. Within 7 days the bug has countermeasure. There is one week per three years to exploit this bug. Hence, a bug is in operational system or hardware with probability 0.0064. The probability to enable an unauthorized access to administrative areas of system or other internal modules of an application is 0.21. [14]. Hence, the probability to exploit a bug and get the access to system is $p = 0.0064 \cdot 0.21 \leq 0.002$. We assume that the probability to get an access to Network Server has probability 0.002, but to Votes Storing Server 0.001, because it is located in internal network.

7) The probability of detecting the attack, which has used an insecure configuration management in server is 0.05.

8) The probability that voter clicks on a malicious link is $p \leq 0.599^{1000}$ per cent.

We assume that the malicious link is served to the voters professionally so that 50 per cent of people click on a malicious link. Hence, the probability that 1000 people use a malicious link for connecting e-voting server is $p \leq 0.599^{1000}$.

9) The probability that a successful crime against the e-voting system will be convicted is 0.8. If the crime was not successful, the probability that it will be convicted is 0.2 per cent [12].

10) About 1% of the people involved in an attack will leak information that causes the attackers to be caught.

Hence, the probability that an attacking group of 10 people will get caught is $q \geq 1 - 0.99^{10} \approx 0.096$.

11) The probability that a forged Network Server or malicious code succeeds in attack is $p \approx 0.95$.

Usually, the accordance between functions of developed information system and claimed system requirements is not 95%. However, for estimating the security of system we promote attackers. If the system is secure against powerful and penetrating attacks, then it is secure against weaker attacks.

12) The probability that voter sells actively his vote, is 0.5 and anonymously 0.7.

We assume intuitively that voter would sell his vote with probability of 0.5 by using active votes' selling environment. The probability that voter would sell the vote by using more anonymous way, is 0.7. For example, a voter would feel more secure to participate in a scheme of votes' selling and buying by using computer based voting data saving and proving software.

13) The probability that a voter agrees to vote many times for a purpose of affecting the result of voting, is 0.9.

14) The probability that voters' computers are vulnerable for session controlling is 31% [11].

15) The probability that adversaries have succeeded to gain control over the connection between the e-voting servers is 15%.

We assume intuitively that if the probability that voters' computers are vulnerable for session controlling is 31% [11], then the control over the session between servers is harder at least twice, therefore $p \leq 0.15$.

4.6.3. Attack game risk analysis

The aim of this subsection is to give the empirical analysis for security assumptions and analyze the vote-specific attacks by using attack game risk analysis.

Attack trees are built by considering the justification of security assumptions. For example, Assumption II says that the encryption scheme is secure. Therefore, we don't build attack trees, which require breaking the encryption algorithm. For the same reason we do not analyze the attacks when the private keys of e-voting are necessary or adversaries steal e-voting servers. Assumption III says that adversaries do not have access to the private keys of e-voting. Assumption VIII says that if considerable attacks are detected, the e-voting will be cancelled and the theft of e-voting server is a considerable and detectable attack.

To create attack trees, we assume that the aim of attacker is not to damage the e-voting completely so that the entire system is injured. If it happens, then by Assumption VIII e-voting is cancelled and new elections are arranged.

Attack trees in this analysis are not perfect; all the possible attacking ways are not considered. However, it is the first attempt to analyze voting-specific attacks by using multi-parameter attack game method.

4.6.3.1. The justification of Assumption IV

Adversaries do not have large-scale access to the voters' private keys $SK[i]$.

Voters have their private keys' information on ID-cards or on e-voting password cards. It is not possible to arrange a large-scale theft of cards, because voters would notice it immediately and the e-voting would be cancelled by Assumption VIII.

In the Estonian e-voting system and in the SERVE project, large-scale attacks are possible either by infecting computers one-by-one or by using automatically propagating attack software (viruses etc.). We assume that both of these methods have the same expenses. We assume that with probability $p \leq 0.99^{1000}$ attackers are able to smuggle malicious code into voters' computers and get the desired data by Characteristic 1. A large-scale access to voters' private keys is a serious attack and the estimation of detecting the attack is 0.8 by Characteristic 9. If we assume that the attack was not successful, then the probability of getting caught is $q \geq 0.096$ by Characteristic 10. For estimating, if the attack is profitable we create the model of attack and calculate the value of profitable *Outcome*. The calculation of the value of *Outcome* is as follows:

$$\begin{aligned}
Outcome &= -Costs + p \cdot (Gains - q \cdot Pealties) - (1 - p) \cdot q_- \cdot Pealties_- \\
&\leq -Costs + Gains \cdot (p \cdot (1 - q) - (1 - p) \cdot q_-) \\
&= -Costs + Gains \cdot (0.99^{1000} \cdot (1 - 0.8) - (1 - 0.99^{1000}) \cdot 0.096) \quad (1) \\
&< -Costs + Gains \cdot (-0.096) \\
&< 0.
\end{aligned}$$

Considering the general characteristics, *Gains* are always bigger than *Costs* of attack. Hence, *Costs* do not affect the attacker's final decision. The attack is unprofitable, if $Outcome < 0$. Therefore, the formula (1) is always negative when $p \cdot (1 - q) - (1 - p) \cdot q_- < 0$ and only the parameters p, q, q_- affect the value of *Outcome*. Additionally, even if the probabilities q and q_- of getting caught are 0.096, the attack is not profitable. Therefore, gain-oriented attacks of large-scale access to the private keys are hardly profitable. Assumption IV is justified for the Estonian e-voting system and for SERVE.

4.6.3.2. The justification of Assumption IX

Adversaries are unable to take large-scale control over the voters' processes.

The success probability of large-scale attack against voters' computers is $p \leq 0.99^{1000}$ by Characteristic 1. According to the previous argument of Assumption IV, taking large-scale control over the voters' processes is not profitable. Hence, Assumption IX is justified for both e-voting systems.

4.6.3.3. Attack against voting privacy: Attack tree analysis

In this subsection, we analyze the attacks with the aim of which is to identify how voters voted. First, it is possible to reveal how voters voted without affecting Voter Application. For example, voters' computers can be injured by malicious code, which saves all activities of voters and informs adversaries. Such an attack is possible only, if the adversary is able to gain the large-scale control over the outputs of voters' processes. According to Assumption IX, the attack is not profitable.

Another possibility to reveal how voters voted is to direct voters to a forged Network Server and use the Man in the Middle Attack for logging voters' encrypted ballot. An adversary cannot decrypt votes, because the private key of e-voting is secure by Assumption III. If an adversary knows random numbers in voters' ballots, then he is able to create all possible encrypted ballots per vote and deduce how voter voted. For this one needs a large-scale attack against the voters' computers so that malicious code will log the voters' voting actions. By Assumption IX, this attack is not profitable.

To summarize, it is not efficient to reveal how voters voted without affecting Voter Application.

To justify the attack against voting privacy, we will build two attack trees, one for SERVE and another for the Estonian e-voting system. The name of the attack tree is *Identify how voters voted*.

Analysis of the SERVE system

The attack tree for SERVE is depicted in Appendix 1. The root node of the tree named *Identify how voters voted* is OR-node with four child nodes:

- A. Attack against Voter Application
- B. Attack against Network Server
- C. Attack against Votes Storing Server
- D. Attack against Votes Counting Server

Node A: *Attack against Voter Application* also represents an attack against the connection between Voter Application and Network Server.

Node A has two child nodes A.1: *Voters use an actively compromised Network Server* and A.2: *Voters download a forged ActiveX from Network Server*.

A.1. *Voters use an actively compromised Network Server* means that voters connect to a forged Network Server and download corrupted ActiveX components of Voter Application. The node is AND node with two child nodes: A.1: *Voters deliberately download a forged ActiveX*; A.2: *Redirecting voters to a forged Network Server*.

Node A.1.1: *Voters deliberately download a forged ActiveX* is successful with probability $p = 0.99^{1000}$ by Characteristic 2. The attack will be detected with probability 0.096 by Characteristic 10.

Node A.1.2: *Redirecting voters to a forged Network Server*. There are many possibilities, which cause the situation where voters deliberately create sessions with a compromised Network Server. For example, if the attacker has control over the local network environment, such as an employer in a workplace or anyone sharing a wireless network, then the attacker can interpose himself as a Man in the Middle of any network communications. Additionally, attacks against the DNS could route traffic to an attacker instead of the legitimate voting service. The success of the attack depends on the probability of whether voters verify the certificate of Network Server or not.

Node A.1.2 is AND-node with three child nodes: *Forged Network Server is developed*; *Voters accept a forged Network Server certificates* and *Voters connect to a forged Network Server*. Developing a forged Network Server is successful with probability $p = 0.95$. The probability of getting caught is $q = 0.8$ and if the attack was not successful then it is $q_- = 0.2$ by Characteristic 9. The probability of voters accepting a forged certificate of Network Server is $p = 0.99^{1000}$ by Characteristic 2 and the probability of getting caught is 0.096 by Characteristic 10.

Node A.1.2.3: *Voters connect to a forged Network Server*. There are four possibilities: voters are directed to a forged web page, voters inadvertently download malicious code, network configuration is affected or local network administrator is bribed. Voters can be directed to the forged web page, if they click on a malicious link, which directs them to a forged Network Server and they accept a forged Network Server certificate. The

probability of voters clicking on a malicious link is $p = 0.599^{1000}$ by Characteristic 8. The probability of it being detected is $q = q_- = 0.096$ by Characteristic 10.

The parameters of the node A.1.2.3.2: *Voters unwillingly download malicious code* have the following values $p = 0.99^{1000}$, $q = 0.8$ and $q_- = 0.096$ by Characteristics 2, 9 and 10. Even if the probability of getting caught is $q = 0.096$, this attack is not efficient for attackers.

Node A.1.2.3.3: *Network configuration is affected*. The probability of exploiting vulnerabilities in network or voters' computers for redirecting voters to a forged server is 0.31 by Characteristic 14. The probability of getting caught is 0.096 by Characteristic 10.

Node A.1.2.3.4: *Local network administrator is bribed*. The probability of bribing a local network administrator with the purpose of affecting the network configuration is 0.33 by Characteristic 4. The probability of the local administrator being caught is 0.096 by Characteristic 10.

Sub tree A.1: *Voters use an actively compromised Network Server* has a negative *Outcome*, which means that this attack is not efficient for the attacker in our attack game analysis.

Node A.2: *Voters download a forged ActiveX from Network Server* means that a corrupted component is smuggled into legal Network Server and voters deliberately download a malicious Voter Application.

Node A.2.1: *Developing malicious code*. By Characteristic 10, we assume that developing a properly working malicious code has success probability $p = 0.95$. It is possible to get caught only when the information about the attack will leak out, so $q = q_- = 0.096$ by Characteristic 10.

A.2.2. *Voters have downloaded an untrustworthy Voter Application* has similar environment characteristics as A.1.1.

Node A.2.3 *Malicious code is smuggled into Network Server* has three child nodes: *Software developer of Voter Application is bribed*; *Server administrator is bribed*; *Insecure configuration management is exploited*.

Node A.2.3.1: *Software developer of Voter Application is bribed*. According to Characteristic 4, a software developer is bribed with probability 0.33. Based on the assumption that development teams use code reviews, misbehaviors in software code are being detected with probability 0.3 by Characteristic 3. Therefore, for estimating the probability of a software developer getting caught, we consider information leaking and the detection rate of misbehaviors in Voter Application. Hence, the probability of getting caught without succeeding for node A.2.2.1 is $q_- = 0.096 + 0.3 = 0.396$ by Characteristics 10 and 3. Bribery is detected with probability 0.3 by Characteristic 5. The success probability of detecting a software developer is $q \approx 0.096 + 0.3 + 0.3 \approx 0.7$.

Node A.2.3.2: *Server administrator is bribed*. According to Characteristic 4, a server administrator is bribed with probability $p = 0.33$. In the event that the attack was not successful, the probability of detecting that the server administrator was bribed is at least $q_- \geq 0.096$, by Characteristic 10. Considering the value of q_- and Characteristic 5, the probability of a server administrator being caught is $q \approx 0.096 + 0.3 \approx 0.4$.

A.2.3.3 *Insecure configuration management is exploited* is successful with probability $p \leq 0.002$ by Characteristic 6. We assume intuitively that the probability of the attack being discovered is 0.05 by Characteristic 7.

When calculating *Outcome*, we see that the attack A.2 is not profitable. To summarize, the attack against Voter Application for identifying how voters voted is not profitable.

Node B: *Attack against Network Server*. The attack against Network Server is achieved by three attacks: B.1 *Logging voters' actions with malicious code*; B.2 *Malicious code is developed* and B.3 *Malicious code is smuggled into the server*. This attack describes the situation where malicious code is inserted into Network Server for the purpose of stealing the received encrypted ballots. If the attackers know the random numbers inside the ballots then it is possible for them to deduce how voters voted by using encrypted ballots.

Node B.1: *Developing malicious code for logging voters' action* represents a large-scale control over voters' processes in order to obtain ballots' random numbers. According to Assumption IX, it is not a profitable attack.

Node B.2: *Malicious code is developed* has the same parameters as node A.2.1.

Node B.3: *Malicious code is smuggled into the server* represents the possibilities of inserting malicious code into Network Server. There are three ways B.3.1: *Software developer of server is bribed*, B.3.2: *Server administrator is bribed*, B.3.3: *Insecure configuration management is exploited*. Node B.3.1 has the same characteristics as A.2.2.1.

Node B.3.2: *Server administrator is bribed*. The node has the same characteristics as node A.2.3.2.

Node B.3.3: *Insecure configuration management is exploited* is successful with probability $p \leq 0.002$ and the probability of detecting the attack is 0.05 by Characteristic 6 and 7.

To summarize, the sub tree *Attack against Network Server* for discovering how voters voted is of negative *Outcome* and this attack is unprofitable.

Node C: *Attack against Votes Storing Server* represents also an attack against the connection between Network Server and Votes Storing Server.

The encrypted ballots are decrypted in Votes Storing Server. Therefore, the attack divides into two branches C.1: *Attack against encrypted ballots* and C.2: *Attack against decrypted ballots*.

Node C.1: *Attack against encrypted ballots*. For identifying how voters voted, adversaries insert the code into Votes Storing Server or catch the connection between the servers in order to discover the voters' encrypted ballots. For deducing how voters voted, adversaries need to know the private key of e-voting or the random numbers in ballots. By Assumption III, adversaries do not have access to the private keys of voting. Therefore, it is necessary for them to attack voters' computers in order to obtain the random numbers.

The analysis of the sub attack C.1 is analogous to sub attack B. *Outcome* of node C.1 is negative, so the attack against encrypted ballots is not profitable.

Node C.2: *Attack against decrypted ballots*. This sub attack is similar to C.1. The only difference is that the adversaries do not have to gain control over the voters' computers. All stolen ballots are decrypted. *Outcome* of the attack is positive; hence, this attack against Votes Storing Server is profitable.

Node D: *Attack against Votes Counting Server*. This attack describes a situation where malicious code is inserted into Votes Counting Server in order to affect the processes. Malicious code changes the time of the database updates so that with every download Votes Counting Server receives one encrypted ballot and one voter's name from the list of voters. If the adversary knows the ballots' random numbers and the encrypted ballots then it is possible to deduce how voters voted. Therefore, the adversary needs to develop and spread a malicious program for logging ballots' random numbers and for smuggling malicious code into Votes Counting Server in order to get encrypted ballots. Attack against Votes Counting Server has characteristics analogous to the attack against Network Server in sub attack B. The probability of succeeding is 0.12. However, the attack is not efficient for the attacker in our attack game analysis.

When analyzing the SERVE project with the attack tree *Identify how voters voted* we see that attack $C \rightarrow C.2 \rightarrow (C.2.1 \text{ AND } C.2.2.2)$ is profitable. Hence, this attack against voting privacy is profitable for SERVE within the defined environment model.

Analysis of the Estonian e-voting system

For analyzing the attack *Identify how voters voted* in the Estonian e-voting system, we created the attack tree which is depicted in Appendix 2.

The root of the tree *Identify how voters voted* is OR node and has three child nodes:

- A. Attack against Voter Application
- B. Attack against Network Server
- C. Attack against Votes Storing Server

Votes Counting Server of the Estonian e-voting system is offline and only encrypted ballots are transferred to server. Therefore, it is not possible to deduce how voter has voted and we do not analyze the attack tree for Votes Counting Server.

Sub attack A has analogous characteristics and attack game analysis as sub tree A.2 in attack tree *Identify how voters voted* analysis for SERVE.

A Man in the Middle Attack between Voter Application and Network Server does not achieve the goal in the Estonian e-voting system. On received signed ballot Network Server verifies, if the signer of the ballot is the same voter who created the session. Otherwise, Network Server will not receive the vote.

Sub attack B: *Attack against Network Server* is analogous to the respective sub attack tree of SERVE in the attack tree *Identify how voters voted* analysis.

In the Estonian e-voting system the encrypted ballots are not decrypted in Votes Storing Server. Therefore, the attack tree analyzes only the attack against the encrypted ballots. This analysis is analogous to previously described node C.1 in the attack tree analysis *Identify how voters voted* for SERVE. In conclusion, the attack against Votes Storing Server is not profitable with the given environment characteristics.

The attack tree *Identify how voters voted* for the Estonian e-voting system does not have profitable attacks in our attack game analysis. Hence, the attack against voting privacy is not profitable for the Estonian e-voting system.

4.6.3.4. Large-scale votes' buying: Attack tree analysis

There is a theoretical advantage for adversaries in the e-voting systems compared to adversaries in traditional voting. The adversaries do not have to physically contact with every voter for affecting his choice. The adversaries should affect at least 1,000 voters for affecting the result of the e-voting. Obviously, the easiest way to affect many voters is to offer votes' buying and selling services. In Subsection 4.6.1 we assumed that *Gains* of attack could be 100 millions monetary units. Let us analyze, whether it is possible to eliminate the *Costs* parameter like we did previously. Obviously, the price of organizing and preparing the attack is much smaller than *Gains*. The biggest expense is the price of votes. In the case when adversaries spend 20% of the profit for buying 1,000 votes, the price of vote is 20,000 monetary units. We assume that such price is attractive for vote sellers. Therefore, *Costs* for buying at least 1,000 votes is smaller than *Gains*.

The deal of vote's buying and selling is possible, if voters can prove how they voted as it is claimed in the conditions of deal. In the following, we create attack tree of *Large scale votes' buying* for the SERVE project and for the Estonian e-voting system.

Analysis of the SERVE system

There are three possibilities to arrange votes' buying and selling in SERVE. First, votes' buying by using votes selling and buying web server. Voters connect to votes buying server for casting votes. The server saves voters' choices and sends ballots to Network

Server of e-voting. Second, voters use votes saving software for getting the receipt of voting and cast a vote directly to Network Server. The voters send the receipt of voting to the adversary for proving how they voted. The adversary attacks the e-voting server for getting a proof that voters' ballot is received. Third, the adversary attacks the servers of e-voting for checking how voters had voted.

The attack tree *Large-scale votes' buying* for SERVE is depicted in Appendix 3. The root node is an OR-node with three child nodes:

- A. *Votes buying server;*
- B. *Spreading votes' receipt software;*
- C. *Attack e-voting server for getting voters' votes.*

Node A: *Votes buying server* describes the situation where voters connect to votes' buying and selling server for casting votes. Votes' buying and selling server saves voters' votes and transfers them in encrypted way to Network Server of e-voting. Sub tree *Votes buying server* has two child nodes: A.1 *Votes buying server and software are developed* and A.2 *Voters connect to votes buying server*.

A.1 *Votes buying server and software are developed*. We assume that with probability 0.95 votes' buying and selling information system is developed successfully by Characteristic 11. To consider the active and public attack, the probability of detecting and punishing the attacking group is 0.8 by Characteristic 9.

A.2 *Voters connect to votes buying server*. We assume that that 50 per cent of voters would sell their vote by Characteristic 12. The probability of detecting voters who have voted by using votes buying server is 0.8 because this is the probability of detecting the votes buying server.

Sub attack A: *Votes buying server* is not profitable for attacker, because the risk to getting caught is big and the probability of buying votes small.

Sub attack B: *Spreading votes' receipt software* describes the situation when software for saving voting data receipt is spread to voters' computers and encrypted ballots are stolen from an e-voting server. Voters install votes' saving software in their computers before e-voting. After e-voting, they can deliver receipt of voting data to the adversary. The receipt of voting data consists of voter's data, a vote, a random number and an encrypted ballot. Adversaries attack the following e-voting servers: Network Server or Votes Storing Server or Votes Counting Server or connections between them for getting pairs of voters' data and encrypted ballots. By comparing voters' receipts and the pairs of voters' data and encrypted ballots one can prove that voters voted as required.

Node B: *Spreading votes' receipt software* has three nodes: B.1 *Developing voting data saving software*, B.2 *Voters use software for saving voting data*, B.3 *Attack voting server for getting received encrypted ballots*.

Node B.1: *Developing voting data saving software*. The probability of the votes' saving software functioning correctly is 0.95 by Characteristic 11. The probability of detection will be 0.096 by Characteristic 10.

Node B.2: *Voters use software for saving voting data*. The success probability of the attack is $p = 0.7$ by Characteristic 12. The probability of detection and punishment is $q = q_- = 1 - 0.99^{1000}$ by Characteristic 10. There are at least 1,000 people involved and Characteristic 10 says that 1 per cent of people leak the information.

B.3 *Attack voting server for getting received encrypted ballots* is AND-node with two child nodes B.3.1: *Developing malicious code for getting voter data and encrypted ballots* and B.3.2: *Inserting the code into server*.

B.3.1 *Developing malicious code for getting voter data and encrypted ballots*. The success probability of the malicious code getting voters' data and encrypted ballots from a voting server is 0.95 by Characteristic 11. The probability of detection is 0.096 by Characteristic 10.

B.3.2 *Inserting the code into server* is OR-node with four child nodes to describe possibilities to smuggle malicious code into Network Server or Votes Storing Server or Votes Counting Server.

B.3.2.1 *Software developer is bribed*. Node A.2.2.1 in Subsection 4.6.3.3 has description of the parameters of attack tree for attack *Software developer is bribed*. B.3.2.2 *Server administrator is bribed* has analogous characteristics as node A.2.3.2 in Subsection 4.6.3.3 The description of B.3.2.3 *Insecure configuration management is exploited* is brought out by node B.3.3 in Subsection 4.6.3.3. B.3.2.4 *Control the connection between servers* is successful with probability 0.15 and the detection of the probability of the attack is 0.096 by Characteristics 15 and 10.

Node B: *Spreading votes' receipt software* does not give a profitable attack.

Subsequently we analyze the sub tree C: *Attack Votes Storing Server for getting voters' ballots*. SERVE's Votes Storing Server decrypts encrypted ballots (Figure 14). Adversary attacks against Votes Storing Server with purpose of stealing pairs of voters' data and ballots. These pairs give the proof how voters have voted. The node C is AND-node with two child nodes C.1: *Developing malicious code for saving votes* and C.2: *Inserting the code into server*.

Node C.1: *Developing malicious code for saving votes*. The probability to succeed and to get caught is explained by description of node B.3.1 in this subsection. Node C.2: *Inserting the code into server* has analogous explanations of attack game parameters with node B.3.2 in this subsection.

Attack Votes Storing Server for getting voters' ballots is successful with probability $p \approx 0.32$ and it has positive *Outcome* of the attack game. Therefore, *Large-scale votes' buying* is profitable under the given characteristics.

Analysis of the Estonian e-voting system

The attack tree *Large-scale votes' buying* for the Estonian e-voting system is depicted in Appendix 4. Votes buying attack against the Estonian e-voting system has one option. Adversaries develop the software for saving voting data in voters' computers. Voters who wish to participate in votes buying and selling use the software for delivering the voting receipt after voting. The receipt of voting data consists of voter's data, a vote, a random number and an encrypted ballot. Adversaries attack Network Server or Votes Storing Server or the connection between them for getting pairs of voters' data and encrypted ballots. The comparison of voters' voting receipt and pairs of voters' data and encrypted ballots give the proof how voters had voted.

It is not successful to sell votes by using votes' buying and selling server because Network Server verifies is the session owner the same voter who signed ballot. If voters use votes buying server then it creates the session with Network Server and Network Server does not accept these votes.

The root of the attack tree is AND-node. It has three child nodes: A *Developing voting data saving software*, B. *Voters use software for saving voting data*, C. *Attack voting server for getting received encrypted ballots*. The description of the attack tree is analogous to the explanation of the node B in the attack tree analysis of SERVE in this subsection.

Outcome of the attack tree is negative. The Estonian e-voting system is secure against a large-scale votes' buying in the defined environment model.

4.6.3.5. Large-scale votes' theft: attack tree analysis

Subsequently we analyze the attack tree in case the eligible voters vote more than once and all votes participate in the computation of the final tally. We model the same attack tree for the Estonian e-voting system and for SERVE. The attack tree *Eligible voters cast votes more than once* is depicted in Appendix 5.

In the Estonian e-voting system all eligible voters can vote more than once. In the Votes Storing Server votes' canceling process eliminates multiple votes. All multiple votes are saved into LOG2. In case the attacker has an access to Votes Storing Server, he could smuggle malicious code into the server. Malicious code injures votes canceling process in order to stop the canceling of desired votes. Moreover, malicious code would change the list of voters in order to avoid the possibility of discovering multiple voters in the list of voters. There is no any control to check whether the number of voters is equal to the numbers of votes. Moreover, the e-voting system does not log the fact that voters revote.

Before we analyze the attack tree *Eligible voters cast votes more than once* we also introduce briefly the attack against SERVE. Each voter can cast a vote only once. For

considering the design of SERVE, Votes Storing Server checks, whether the voter has already voted. Therefore, adversary attacks Votes Storing Server for affecting the functions so that concerned eligible voters could vote more than once.

The calculations of attack tree *Eligible voters cast votes more than once* are analogous for both e-voting systems. The root of tree is AND-node and it has three child nodes: A. *Voters vote more than once*, B. *Developing malicious code for changing votes canceling phase*, C. *Malicious code is smuggled in to server*.

A. *Voters vote more than once*. In the case when 1,000 eligible voters vote twice the probability to succeed voting is $p = 0.9^{1000}$ by Characteristic 13. The probability of getting caught is $q = q_- = 1 - 0.99^{1000} = 0.999$ by Characteristic 10.

B. *Developing malicious code for enabling vote more than once*. The development of malicious code is successful with probability $p = 0.95$; $q = q_- = 0.096$ by Characteristics 11 and 10.

Node C is OR-node with three child nodes. The most profitable way to have access to Votes Storing Server is to bribe the administrator.

To summarize, *Outcome* of the attack game is negative, so it is not profitable. Even in case the adversary is able to convince 10 eligible voters to vote 100 times, the attack is not profitable.

4.6.3.6. Large-scale disfranchisement before receiving votes

In this subsection, we analyze the attack tree named *Large-scale votes' disfranchisement before receiving votes*. The Estonian e-voting system and the SERVE system have the same analysis of attack tree with the same environment characteristics. This attack tree is depicted in Appendix 6. The attack tree is divided into two sub trees: node A: *Voter Application is injured* and node B: *Votes are eliminated before receiving*.

Node A is AND-node with two child nodes A.1: *Malicious code is downloaded unwillingly* and A.2: *Developing malicious code*. Node A: *Malicious code is downloaded unwillingly* has the following values of parameters $p = 0.99^{1000}$, $q = 0.8$ and $q_- = 0.096$ by Characteristics 2, 9 and 10. The sub node *Developing malicious code* has the parameters $p = 0.95$; $q = q_- = 0.096$ by Characteristics 11 and 10.

Node B is OR-node with two child nodes B.1: *Redirecting voters to a forged Network Server* and B.2: *Passively compromised Network Server*. The sub tree *Redirecting voters to a forged Network Server* is described in Subsection 4.6.3.3 by node A.1.2. It is not efficient for attacker with these environment characteristics, so let us analyze the sub tree B.2 *Passively compromised Network Server*. Node B.2 is AND-node. It divides into two child nodes: *Developing malicious code* and *Malicious code is smuggled into server*. Node named *Developing malicious code* is analogous to node A.2. Node B.2.2: *Malicious code is smuggled into server* has three child nodes: *Software developer of server is*

bribed, Server administrator is bribed, and Insecure configuration management is exploited. The analysis is analogous to nodes A.2.3.1, A.2.3.2 and B.3.3 description in Subsection 4.6.3.3.

Outcome of the tree Large-scale votes' disfranchisement before receiving votes is negative with defined model of environment. Therefore, the attack is unlikely efficient for rationally thinking attacker in the Estonian e-voting system and the SERVE system.

4.6.3.7. Large-scale disfranchisement of votes in two servers

In this subsection, we present the analysis of the attack against two e-voting servers. This attack tree named *Eliminating votes in two e-voting servers* covers for the Estonian e-voting system the attack analysis against Network Server and Votes Storing Server and for the SERVE system attack against Votes Storing Server and against Votes Counting Server. Additionally, this attack tree analysis represents any kind of attack against the two e-voting servers, for example the attack against Votes Counting Server and Votes Storing Server for adding votes in the Estonian e-voting system. The attack tree is depicted in Appendix 7.

The tree has two AND nodes which correspond to the attacks against the two servers. Sub attacks A and B have similar description of the parameters of the attack game with node B.2 *Passively compromised Network Server* in Subsection 4.6.3.6.

In our environment model the most profitable way to have an access to the server is to bribe the server administrator. Two server administrators can be corrupted with probability $p \approx 0.1$ by Characteristic 3. The probabilities to getting caught and be convicted are $q = 0.3 + 0.096$ and $q_- = 0.096$ by Characteristics 5 and 10.

To summarize, it is not efficient to attack two e-voting servers in the Estonian e-voting system and SERVE. However, it is more likely that attacks against only one server are profitable.

4.6.3.8. Large-scale modification of ballots in the connection between Voter Application and Network Server of SERVE

Subsequently we analyze a Man in the Middle Attack between Voter Application and Network Server in SERVE. To perform the attack, the attackers should develop a forged server and the voters should connect to the forged server instead of Network Server. The corresponding attack tree named *Attack against the connection between Voter Application and Network Server for changing the ballots* is depicted in Appendix 8.

The root of the tree is AND-node with two child nodes A: *Forged server is developed* and B: *Voters connect to a forged server*. Node A is successful with probability 0.95 and the probability of getting caught is $q = 0.8$ and $q_- = 0.2$. Node B has four child nodes:

- B.1: *Voters click on wrong link,*
- B.2: *Voters accept a forged Network Server certificate,*
- B.3: *Voters unwillingly download malicious code,*

B.4: Access to voters' computer by using local network.

Node B.1: *Voters click on wrong link* represents social engineering attacks when voters are fooled into impression that they are communicating with voting server. This could happen for example, by emailing or announcing a link that should point to the voting server, but in fact it does not. The probability of voters clicking on wrong link is 0.599^{1000} by Characteristic 8. The probability of detecting the attack and proving it in court is 0.096 by Characteristic 10.

Node B.2: *Voters accept a forged Network Server certificate* represents the situation when voters do not check the certificate of Network Server or accept the forged Network Server certificate and create unwillingly a session with wrong server. The probability to succeed the attack is $p \leq 0.99^{1000}$ by Characteristic 2. The probabilities to getting caught are $q = q_- = 0.096$ by Characteristic 10.

Node B.3: *Voters unwillingly download malicious code*. Malicious code in voters' computers gives control over voters' voting processes. The probability of downloading malicious ActiveX is 1 per cent, therefore the success probability of a large-scale attack is $p \leq 0.99^{1000}$ and the probabilities of getting caught are $q = q_- = 0.096$.

Node B.4: *Access to voters' computers by using local network* has two possibilities: B.4.1 *Malicious code is inserted into voters' computer or into local network* and B.4.2 *Local network administrator is bribed*.

Node B.4.1: *Malicious code is inserted into voters' computer or into local network*. The probability of taking control over the local network or smuggling malicious code into the voters' computers is $p = 0.002 + 0.31 = 0.312$ by Characteristics 6 and 14. With probability 0.05 attackers will be caught and convicted in court by Characteristic 7.

Node B.4.2: *Local network administrator is bribed*. According to Characteristic 4, a server administrator is bribed with probability $p = 0.33$. We assume that in the case when attack was not successful, the probability to detect that the server administrator was bribed, is at least $q_- \geq 0.096$ by Characteristic 10. Considering the value of q_- and the Characteristic 5, the probability of catching a server administrator is $q \approx 0.4$.

The calculation of attack tree has negative *Outcome*, therefore the attack is unlikely profitable. The threat *Attack against the connection between Voter Application and Network Server for changing ballots* is unlikely.

4.6.3.9. Control over processes of Votes Storing Server of SERVE

The attack represents cases when adversary has an access to Votes Storing Server for smuggling malicious code into server. Malicious code can have many aims. For example, to change decrypted votes or to add votes. The attack tree named *Control over processes of Votes Storing Server of SERVE* is depicted in Appendix 9.

The root of tree is AND-node with two child nodes A: *Developing malicious code* and B: *Malicious code is smuggled into server*.

Node A: *Developing malicious code*. We assume that developing of a properly working malicious code has the probability $p = 0.95$ by Characteristic 11. The possibility of getting caught will happen only when the information about attack will leak out, so $q = q_- = 0.096$ by Characteristic 10.

Node B: *Malicious code is smuggled into server*. It is divided into three child nodes.

Node B.1: *Software developer of server is bribed*. According to Characteristic 4, a software developer is bribed with probability 0.33. The software developer's probability of being caught without successful attack is $q_- = 0.096 + 0.3 = 0.396$ by Characteristics 10 and 3. The success probability of catching a software developer is $q \approx 0.096 + 0.3 + 0.3 \approx 0.7$ by Characteristics 10 and 3 and 5.

Node B.2: *Server administrator is bribed*. The explanations of characteristics are brought out in Subsection 4.6.3.3 by node A.2.3.2. It is efficient for the attacker to bribe a server administrator.

Node B.3: *Insecure configuration management is exploited* is successful with probability $p \leq 0.001$ by Characteristic 6. The probability of detecting the attack is 0.05 by Characteristic 7.

The attack tree analysis shows that it is likely that the adversary tries to take control over processes of Votes Storing Server of SERVE by bribing the server administrator. Votes Storing Server is a weak link in the SERVE system.

4.6.3.10. Large-scale votes' adding in Votes Counting Server of SERVE

Votes Counting Server of SERVE is online server which downloads the list of voters and encrypted ballots from Votes Storing Server. We analyze the threat that the adversary obtains the public key of Votes Counting Server and adds the encrypted ballots to Votes Counting Server or the transmission session between Votes Storing Server and Votes Counting Server. For attacking Votes Storing Server it is possible to obtain the public key of Votes Counting Server. The corresponding attack tree named *Large-scale votes' adding in Votes Counting Server of SERVE* is depicted in Appendix 10.

The root of tree is AND-node with three child nodes A: *Attack against Votes Storing Server* and B: *Create encrypted ballots* and C: *Passively compromised Votes Counting Server*.

Node A: *Attack against Votes Storing Server*. The subtree is similar to the attack tree *Control over processes of Votes Storing Server of SERVE* in Subsection 4.6.3.9. This sub attack is likely successful and efficient for attacker.

Node B: *Create encrypted ballots* represent the process of creating correctly verified encrypted ballots. It is successful with probability 0.95 by Characteristic 11. Characteristic 10 says that with probability 0.096 the group of attackers will be caught and convicted.

Node C: *Passively compromised Votes Counting Server*. It represents beside attack against Votes Counting Server also the attack against the connection between Votes Storing Server and Votes Counting Server. This subtree has analogous analysis as attack against Votes Storing Server. The values of the attack game parameters are given in Subsection 4.6.3.9.

The success probability of the attack *Large-scale votes' adding in Votes Counting Server of SERVE* is 0.093. *Outcome* of attack game is negative, therefore the attack is unlikely profitable. To summarize, a large-scale votes being added to Votes Counting Server is not an efficient attack in SERVE.

4.6.3.11. Attack tree analysis with alternative environment characteristics

In the previous subsections, we analyzed the attack games in a fixed environment model. Three attacks were profitable in SERVE. Seven attack trees gave the negative results in the attack games. In this subsection, we point out alternative environment characteristics which change the values of the attack games. It gives us additional information about the seriousness of threats. We see that the results of attack games are quite similar in analogous environment models. On the other hand, when the result of the attack game is very sensitive to small changes of parameters, it should be reasonable to consider the threat as a risk. It turns out that reasonable changes do not have influence on the final results of the analysis, which means that the result of the analysis is likely truthful regardless of limited knowledge about the real values.

Attack tree *Identify how voters voted* analysis for SERVE has a weak link $C \rightarrow C.2. \rightarrow (C.2.1.ANDC.2.2.2)$. It means that to bribe a server administrator of Votes Storing Server for smuggle malicious code into server is likely a successful attack. Malicious code has the aim to read decrypted ballots and voters' personal data and to forward them to the adversary. Hence, the adversary is able to identify how voters voted. In Appendix 11, we depict the alternative attack game analysis with different result. If we assume that the probability of getting caught in the case when the attack succeeds is $q = 0.6$ instead of $q = 0.4$ in the node *C.2.2.2 Server administrator is bribed* then the attack against Votes Storing Server is unprofitable. We see that this change of the parameter is relatively big. Therefore, the SERVE project is not practically secure against attack *Identify how voters voted* in our and similar environment models.

Attack tree named *Large-scale votes' buying* analysis for SERVE shows that the sub attack against Votes Storing Server for getting voters' ballots is likely successful. The sub attack C has two child nodes: *Developing malicious code for saving votes* and *Inserting the code into server*. It is profitable to develop malicious code and a server administrator bribing is useful for inserting the code into server. If we change the probability of achieving successful malicious code so that it is $p = 0.80$ instead of $p = 0.95$ and assume

that a server administrator will be caught with probability $q = 0.5$ then the attack against Votes Storing Server is unlikely profitable. Obviously, attackers develop malicious codes quite correctly to achieve the attack. Even, if the probability that code does what is required is 90%, then the result of attack game is positive. Moreover, this probability is insufficient to give the practical security for e-voting system. We presented the analysis of the parameter of detecting a server administrator in previous paragraph. Therefore, the SERVE project is not secure against a large-scale votes' buying. The alternative attack tree *Large-scale votes' buying* is depicted in Appendix 12.

The attack tree *Control over processes of Votes Storing Server of SERVE* has positive result from the attack game. It means that it is likely successful to smuggle malicious code into Votes Storing Server of SERVE with the purpose of affecting the processes or reading the ballots, etc. The attack tree consists of developing malicious code and smuggling this malicious code into the server. The alternative attack game risk analysis is depicted in Appendix 13. As we noticed before, the most profitable way is to bribe a server administrator. If the server administrator is detected and convicted with probability $q = 0.5$, then attack game gives negative result. In case we do not change this parameter of detecting bribing, then in order to get the negative value of the attack game, we should change others parameters even more. However, such parameter changing does not provide sufficient trust to believe that SERVE is secure against the attack.

Subsequently we analyze the attack trees which gave the negative results of attack games in previous subsections. The alternative attack game risk analysis of the Estonian e-voting system for the attack tree *Identify how voters voted* is depicted in Appendix 14. It gives the positive *Outcome* of the attack game, if the following parameters are changed. First, we change the attack tree's parameter to succeed the attack $p = 0.999^{1000}$. It means that 0.1 per cent of voters verify the correctness of ActiveX component of Voter Application. Additionally, if about 0.1 per cent of the people involved in an attack will leak information that causes the attackers to be caught then $q = q_- = 0.01$, instead of $q = q_- = 0.096$. Second, the probability of bribing a server administrator is detected by probability $q = 0.2$. With these parameters the attack against Voter Application in order to identify how the voters had voted, would be likely successful in attack game. On the other hand, the probability that number of voters are able to detect an affection of their computers is 0.01 is too negligible. The environment characteristic about the voters' awareness to check the signature of Voter Application is decreased for 10 times. Additionally, the probability of detecting a server administrator bribing is decreased for 2 times. It would change our environment model highly. Therefore, it is reasonable to declare that the e-voting system is practically secure against the attack in the defined environment model.

The attack tree *Large-scale votes' buying* analysis for the Estonian e-voting system shows that a large-scale votes' buying is not successful attack for a rationally thinking attacker. It would be successful, if we change Characteristic 10, so that only 0.01 per cent of the people will leak the information and the probability of the detection of a server administrator bribing is $q = 0.2$. The nodes should have the probability of getting caught of $q = q_- \geq 1 - 0.9999^{10} \approx 0.001$ instead of 0.096. Additionally, the fact that voters have used software for saving the voting data would be revealed with negligible probability q

$= q_- \approx 0.1$. Obviously, if at least 1,000 voters have sold their votes, the probability to leak the information is bigger. Even, if the probability of detection of a server administrator bribing is $q = 0.3$, the attack game would have negative result. Only highly changed environment model would give the positive result in the attack game analysis. Therefore, it is reasonable to believe that the Estonian e-voting system is secure against a large-scale votes' buying in hereby defined and similar environments. This alternative analysis is depicted in Appendix 15.

The attack game analysis *Eligible voters cast votes more than once* is not an efficient attack by the attack game risk analysis in the Estonian e-voting system and SERVE in Subsection 4.6.3.5. If we change all the parameters q and q_- which influence the result of the attack game and the parameter p for node *Voters vote more than once* then the game gives positive *Outcome*. The probability of succeeding that the voters vote more than once, should be $p = 0.9^{10}$ instead of $p = 0.9^{1000}$. It means that 10 voters are convinced to vote more than 100 times and each voter will be involved with the probability 0.99. If the attack involves 10 voters and attacking team consists of 10 people, then the probability of getting caught is $q = q_- \geq 1 - 0.999^{20} \approx 0.0198$. The detection probability is $q_- \geq 1 - 0.999^{10} \approx 0.01$. The probabilities to detect the development of malicious code for affecting votes canceling phase and the unsuccessful bribing of the server administration are negligible $q = q_- \geq 1 - 0.999^{10} \approx 0.01$. The probability of detecting malicious code attack against the e-voting server is almost 10 times smaller than environment characteristic in the defined environment model. Therefore, we consider that SERVE and the Estonian e-voting system are secure against the attack *Eligible voters cast votes more than once* in our environment model. The alternative analysis is depicted in Appendix 16.

The attack *Eliminating votes in two e-voting servers* would have positive *Outcome* in the attack game if all the parameters q in the game are $q = q_- \geq 1 - 0.999^{10} \approx 0.01$. The parameter is changed at least 10 times. Therefore, we declare that SERVE and the Estonian e-voting system are secure against the attack *Eliminating votes in two e-voting* in defined and similar environment models. This alternative attack game analysis is depicted in Appendix 17.

The attack *Large-scale votes' disfranchisement before receiving votes* is not successful in the Estonian e-voting and the SERVE system. The alternative attack game analysis is depicted in Appendix 18. Analogically to previous analysis the probability of detecting the attack should be small $q = q_- \approx 0.01$. Additionally, the probability of noticing and revealing if votes' computers or Voter Applications are infected is diminished 10 times to 0.1 per cent. Therefore, the nodes A.1 and B.1.3.2 have the probability of $p = 0.368$. Moreover, it causes the situation that a large-scale attack against voters' computer is likely successful. This is contradiction with Assumption IX. Therefore, we consider that the e-voting systems are secure against *Large-scale votes' disfranchisement before receiving votes*.

The tree *Attack against the connection between Voter Application and Network Server for changing ballots* analysis for SERVE gives the negative *Outcome* in the attack game with defined environment characteristics. If we assume that the environment model is friendlier to attackers, then attack game has positive *Outcome*. Hence, the probability of getting caught should be $q = q_{-} \approx 0.096$ and a local network administrator would be bribed and malicious code should be successfully inserted into the voters' computer or local network with probability of 0.6. It means that even a forged Network Serve would be discovered only in case, if somebody will leak the information. Obviously, e-voting has gained a lot of attention and the probability to detect a forged Network Server is bigger. If the probability is 0.2, then the attack game gives the negative result. Therefore, we consider that the SERVE project is secure against the attack for changing the ballots by compromising the connection between Voter Application and Network Server. The alternative attack game analysis is depicted in Appendix 19.

The attack *Large-scale votes' adding in Votes Counting Server of SERVE* is not a profitable attack by using attack game risk analysis in Subsection 4.6.3.10. If the environment model has the following characteristics then the attack would be successful in the attack game analysis. As we have seen previously, the probability to detect the attack should be in all sub attacks $q = q_{-} \approx 0.01$. A server administrator should be bribed with probability 0.6. It means that the probability of detecting the attack is 10 times smaller. Therefore, we assume that the attack is also unlikely profitable in similar environment models. This attack game analysis is depicted in Appendix 20.

Summary

In this work, we adopted rational security analysis methods for studying the practical security in the two e-voting systems: the Estonian e-voting system and the SERVE project. In order to declare that e-voting system is secure, it must be as secure as the traditional voting. The traditional voting methods are considered to be practically secure and are resistant to large-scale threats. This means that the e-voting systems must be also secure against the large-scale voting-specific attacks and the security properties of the e-voting must be justified.

At the beginning of our analysis we modeled the Estonian and the SERVE e-voting systems. For proving practical security of the systems, we created the environment model as similar as possible to the real world. We defined the environment model by using society characteristics, security assumptions and the properties of adversaries. We assumed that adversaries are rationally thinking persons and attack with purpose to affect the result of elections. Therefore, we analyzed large-scale attacks that affect many votes. To construct the behavioral model of adversaries we used the attack game method. This method represents purposed attacks in multi-parameter attack trees by considering environment characteristics. For giving the security justification, we analyzed the success of adversaries against the e-voting systems in the defined environment model.

We developed a rational method in order to analyze the practical security of the e-voting systems and to compare objectively their security levels. We showed that the Estonian e-voting system is practically secure in the defined environment model. The SERVE project has vulnerabilities in the system design, which make voting-specific attacks profitable for attackers. Therefore, the SERVE system is not practically secure in the defined environment model. We also showed that reasonable changes with environment characteristics will not change the main conclusions of this work.

The results of the work are disputable, because the characteristics of the defined environment model are arguable. However, this work is one of the first attempts to analyze rationally the security of e-voting by creating the model of society. It is necessary to continue to study the society characteristics for achieving more realistic environment models. Additionally, future work is needed for determining how the environment characteristics affect the result of the security analysis.

Kokkuvõte

Käesolevas töös kohandame ratsionaalse turvaanalüüsi meetodit, analüüsimeks praktilist turvalisust kahes e-valimiste süsteemis: Eesti e-valimiste süsteem ja Ameerika e-valimiste projekt SERVE. E-valimiste süsteemi loetakse turvaliseks kui ta on ligikaudu sama turvaline kui tava-hääletamissüsteem. Traditsioonilisi valimissüsteeme võib lugeda praktiliselt turvaliseks ja vastupidavaks ulatuslike tagajärgedega, valimisi tervikuna mõjutavatele ohtudele. See tähendab, et e-valimiste süsteemi võib samuti lugeda praktiliselt turvaliseks kui ta on vastupidav ulatuslike tagajärgedega rünnete suhtes.

Töö esimeses osas on modelleeritud Eesti ja SERVE e-valimiste süsteeme. Analüüsimeks süsteemide praktilist turvalisust, loome reaalsele keskkonnale võimalikult lähedase mudeli. Keskkonnamudeli defineerimisel kasutame ühiskonna omadusi, turvalisuse eeldusi ja ründaja käitumuslikke karakteristikuid. Eeldame, et ründajad on ratsionaalselt mõtlevad inimesed, kelle eesmärk on mõjutada valimiste tulemusi. Sellest tulenevalt analüüsime ründeid, mis mõjutavad suurel hulgal hääli. Ründaja käitumismudeli konstrueerimiseks kasutame ründemängu meetodit. See meetod võimaldab analüüsida eesmärgistatud ründeid mitme parameetriga ründepuudes keskkonnaparameetreid arvestades. E-valimiste turvaanalüüsis hindame ründaja edukust mõjutada hääletamise tulemusi modelleeritud keskkonnas.

Käesolevas töös arendame ratsionaalset meetodit, analüüsimeks praktilist turvalisust e-valimiste süsteemides ja hindamaks objektiivselt turvalisuse taset. Näitame, et Eesti e-valimiste süsteem on praktiliselt turvaline defineeritud keskkonnamudelil. SERVE süsteemis on nõrkusi, mida ründajad on võimelised edukalt ära kasutama valimisspetsiifiliste rünnete saavutamiseks. Seepärast ei ole SERVE süsteem defineeritud keskkonnamudelil praktiliselt turvaline. Lisaks näitame töös, et mõistikul tasemel muudatused keskkonnaparameetrites ei muuda turvaanalüüsi lõpptulemust. Seega kui meie keskkonnamudel on lähedane reaalsele keskkonnale, siis on meie turvaanalüüsi järeldused tõesed.

Töö tulemused ei ole täiuslikud, sest defineeritud keskkonnamudeli karakteristikud on vaieldavad. Sellegipoolest on see töö üks esimesi katseid ratsionaalselt analüüsida e-valimiste turvalisust tervikuna, luues selleks ühiskonnamudeli. Kahtlemata on vaja jätkata ühiskonna omaduste uurimist, saavutamaks täpsemat keskkonnamudelit. Edaspidi tuleks uurida, kuidas keskkonnaparameetrid mõjutavad turvaanalüüsi tulemusi.

Appendices

Appendix 1

Node	Node name	Node type	p	q	q_{-}	π	π	Outcome
	Identify how voters voted	OR	0.313	0.038	0.009	49.6·10⁶	13.976·10⁶	6.205·10⁶
A	Attack against Voter Application	OR	1.35·10⁻⁵	0.004	0.001	59.2·10⁶	37.867·10⁶	-37.866·10⁶
A.1	Voters use an actively compromised Network Server	AND	5.84·10⁻¹⁰	0.00070779	0.00018	108.8·10⁶	1.04·10⁸	-10.448·10⁷
A.1.1	<i>Voters deliberately download a forged ActiveX</i>		4.32·10 ⁻⁵	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	-9.595·10 ⁶
A.1.2	<i>Redirecting voters to a forged Network Server</i>	AND	1.35·10 ⁻⁵	0.007	0.002	99.2·10 ⁶	94.886·10 ⁶	-94.885·10 ⁶
A.1.2.1	Forge Network Server is developed		0.950	0.800	0.200	80·10 ⁶	2·10 ⁶	18·10 ⁶
A.1.2.2	Voters accept a forged Network Server certificate		4.32·10 ⁻⁵	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	-9.595·10 ⁶
A.1.2.3	Voters connect to a forged Network Server	OR	0.330	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	23.4·10 ⁶
A.1.2.3.1	Voters click on wrong link		2.7·10 ⁻²²³	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	-9.6·10 ⁶
A.1.2.3.2	Voters unwillingly download malicious code		4.32·10 ⁻⁵	0.800	0.096	80·10 ⁶	9.6·10 ⁶	-9.598·10 ⁶
A.1.2.3.3	Network configuration is affected		0.310	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	21.4·10 ⁶
A.1.2.3.4	Local network administrator is bribed		0.330	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	23.4·10 ⁶
A.2	Voters download a forged ActiveX from Network Server	AND	1.35·10⁻⁵	0.004	0.001	59.2·10⁶	37.867·10⁶	-37.866·10⁶
A.2.1	<i>Developing malicious code</i>		0.95	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	85.4·10 ⁶
A.2.2	<i>Voters have downloaded an untrustworthy Voter Application</i>		4.32·10 ⁻⁵	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	-9.595·10 ⁶
A.2.3	<i>Malicious code is smuggled into Network Server</i>	OR	0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
A.2.3.1	Software developer of Voter Application is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
A.2.3.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
A.2.3.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶
B	Attack against Network Server	AND	1.35E-05	0.031	0.001	129.6·10⁶	37.870·10⁶	-37.87·10⁶
B.1	Logging voters' actions with malicious code		4.32·10⁻⁵	0.8	0.096	80·10⁶	9.6·10⁶	-9.598·10⁶
B.2	Malicious code is developed		0.95	0.096	0.096	9.6·10⁶	9.6·10⁶	85.4·10⁶
B.3	Malicious code is smuggled into server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13.368·10⁶
B.3.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
B.3.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
B.3.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶

C	Attack against Votes Storing Server	OR	0.314	0.039	0.009	49.6·10⁶	13.976·10⁶	6.205·10⁶
C.1	Attack against encrypted ballots	AND	0.116	0.031	0.001	129.6·10⁶	66.054·10⁶	-61.852·10⁶
<i>C.1.1</i>	<i>Logging ballots' random numbers with malicious code</i>		<i>0.368</i>	<i>0.8</i>	<i>0.096</i>	<i>80·10⁶</i>	<i>9.6·10⁶</i>	<i>1.283·10⁶</i>
<i>C.1.2</i>	<i>Developing malicious code</i>		<i>0.95</i>	<i>0.096</i>	<i>0.096</i>	<i>9.6·10⁶</i>	<i>9.6·10⁶</i>	<i>85.4·10⁶</i>
<i>C.1.3</i>	<i>Malicious code is inserted into Votes Storing Server</i>	<i>OR</i>	<i>0.33</i>	<i>0.4</i>	<i>0.096</i>	<i>40·10⁶</i>	<i>9.6·10⁶</i>	<i>13.368·10⁶</i>
<i>C.1.3.1</i>	<i>Software developer of server is bribed</i>		<i>0.33</i>	<i>0.7</i>	<i>0.396</i>	<i>70·10⁶</i>	<i>39.6·10⁶</i>	<i>-16.632·10⁶</i>
<i>C.1.3.2</i>	<i>Server administrator is bribed</i>		<i>0.33</i>	<i>0.4</i>	<i>0.096</i>	<i>40·10⁶</i>	<i>9.6·10⁶</i>	<i>13.368·10⁶</i>
<i>C.1.3.3</i>	<i>Insecure configuration management is exploited</i>		<i>0.002</i>	<i>0.05</i>	<i>0.05</i>	<i>5·10⁶</i>	<i>5·10⁶</i>	<i>-4.8·10⁶</i>
C.2	Attack against decrypted ballots	AND	0.314	0.039	0.009	49.6·10⁶	13.976·10⁶	6.205·10⁶
<i>C.2.1</i>	<i>Developing malicious code</i>		<i>0.95</i>	<i>0.096</i>	<i>0.096</i>	<i>9.6·10⁶</i>	<i>9.6·10⁶</i>	<i>85.4·10⁶</i>
<i>C.2.2</i>	<i>Malicious code is inserted into server</i>	<i>OR</i>	<i>0.33</i>	<i>0.4</i>	<i>0.096</i>	<i>40·10⁶</i>	<i>9.6·10⁶</i>	<i>13.368·10⁶</i>
<i>C.2.2.1</i>	<i>Software developer of server is bribed</i>		<i>0.33</i>	<i>0.7</i>	<i>0.396</i>	<i>70·10⁶</i>	<i>39.6·10⁶</i>	<i>-16.632·10⁶</i>
<i>C.2.2.2</i>	<i>Server administrator is bribed</i>		<i>0.33</i>	<i>0.4</i>	<i>0.096</i>	<i>40·10⁶</i>	<i>9.6·10⁶</i>	<i>13.368·10⁶</i>
<i>C.2.2.3</i>	<i>Insecure configuration management is exploited</i>		<i>0.002</i>	<i>0.05</i>	<i>0.05</i>	<i>5·10⁶</i>	<i>5·10⁶</i>	<i>-4.8·10⁶</i>
D	Attack against Votes Counting Server	AND	0.116	0.031	0.001	129.6·10⁶	66.054·10⁶	-61.852·10⁶
D.1	Logging voters' actions with malicious code		0.368	0.8	0.096	80·10⁶	9.6·10⁶	1.283·10⁶
D.2	Developing malicious code		0.95	0.096	0.096	9.6·10⁶	9.6·10⁶	85.4·10⁶
D.3	Malicious code is inserted into Votes Counting Server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13.368·10⁶
<i>D.3.1</i>	<i>Software developer of server is bribed</i>		<i>0.33</i>	<i>0.7</i>	<i>0.396</i>	<i>70·10⁶</i>	<i>39.6·10⁶</i>	<i>-16.632·10⁶</i>
<i>D.3.2</i>	<i>Server administrator is bribed</i>		<i>0.33</i>	<i>0.4</i>	<i>0.096</i>	<i>40·10⁶</i>	<i>9.6·10⁶</i>	<i>13.368·10⁶</i>
<i>D.3.3</i>	<i>Insecure configuration management is exploited</i>		<i>0.002</i>	<i>0.05</i>	<i>0.05</i>	<i>5·10⁶</i>	<i>5·10⁶</i>	<i>-4.8·10⁶</i>

Appendix 2

Node	Node name	Node type	p	q	q ₋	π	π ₋	Outcome
	Identify how voters voted	OR				59.2·10⁶	37.867·10⁶	-37.866·10⁶
A	Attack against Voter Application	AND	1.35·10⁻⁵	0.004	0.0009	59.2·10⁶	37.867·10⁶	-37.866·10⁶
A.1	Developing malicious code		0.95	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	85.4·10 ⁶
A.2	Voters have downloaded an untrustworthy Voter Application		4.32·10 ⁻⁵	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	-9.595·10 ⁶
A.3	Malicious code is smuggled into Network Server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13.368·10⁶
A.3.1	Software developer of Voter Application is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
A.3.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
A.3.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶
B	Attack against Network Server	AND	1.35·10⁻⁵	0.031	0.0009	129.6·10⁶	37.870·10⁶	-37.870·10⁶
B.1	Logging voters' actions with malicious code		4.32·10 ⁻⁵	0.8	0.096	80·10 ⁶	9.6·10 ⁶	-9.598·10 ⁶
B.2	Developing malicious code		0.95	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	85.4·10 ⁶
B.3	Malicious code is smuggled into server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13.368·10⁶
B.3.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
B.3.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
B.3.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶
C	Attack against Votes Storing Server	AND	1.35·10⁻⁵	0.031	0.0009	129.6·10⁶	37.870·10⁶	-37.870·10⁶
C.1	Logging voters' ballots' numbers with malicious code		4.32·10 ⁻⁵	0.8	0.096	80·10 ⁶	9.6·10 ⁶	-9.598·10 ⁶
C.2	Developing malicious code		0.95	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	85.4·10 ⁶
C.3	Malicious code is inserted into Votes Storing Server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13.368·10⁶
C.3.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
C.3.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
C.3.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶

Appendix 3

Node	Node name	Node type	p	q	q ₋	π	π ₋	Outcome
	Large-scale votes' buying	OR						6.205·10⁶
A	Votes buying server	AND	0.475	0.64	0.64	160·10⁶	87.619·10⁶	-74.5·10⁶
A.1	Votes buying server and software are developed		0.95	0.8	0.8	80·10 ⁶	80·10 ⁶	15·10 ⁶
A.2	Voters connect to votes buying server		0.5	0.8	0.8	80·10 ⁶	80·10 ⁶	-30·10 ⁶
B	Spreading votes' receipt software	AND	0.208	0.0037	0.00089	159.19·10⁶	142.208·10⁶	-124.9·10⁶
B.1	Developing voting data saving software		0.95	0.096	0.096	9.6·10⁶	9.6·10⁶	85.4·10⁶
B.2	Voters use software for saving voting data		0.7	0.999	0.999	99.995·10⁶	99.995·10⁶	-29.99·10⁶
B.3	Attack voting server for getting received encrypted ballots	AND	0.3135	0.0384	0.0092	49.6·10⁶	13.976·10⁶	6.205·10⁶
B.3.1	<i>Developing malicious code for getting voter data and encrypted ballots</i>		0.95	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	85.4·10 ⁶
B.3.2	<i>Inserting the code into server</i>	OR	0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
B.3.2.1	Software developer is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
B.3.2.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
B.3.2.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶
B.3.2.4	Control the connection between servers		0.15	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	5.4·10 ⁶
C	Attack Votes Storing Server for getting voters' ballots	AND	0.3135	0.0384	0.0092	49.6·10⁶	13.976·10⁶	6.205·10⁶
C.1	Developing malicious code for saving votes		0.95	0.096	0.096	9.6·10⁶	9.6·10⁶	85.4·10⁶
C.2	Inserting the code into server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13.368·10⁶
C.2.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
C.2.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
C.2.3	Insecure configuration management is exploited		0.002	0.05	0.1	5·10 ⁶	10·10 ⁶	-9.79·10 ⁶

Appendix 4

Node	Node name	Node type	p	q	q_{-}	π	π_{-}	Outcome
	Large-scale votes' buying	AND	0.2085			$159.195 \cdot 10^6$	$142.208 \cdot 10^6$	$-124.902 \cdot 10^6$
A	Developing voting data saving software		0.95	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$85.4 \cdot 10^6$
B	Voters use software for saving voting data		0.7	0.999	0.999	$99.995 \cdot 10^6$	$99.995 \cdot 10^6$	$-29.99 \cdot 10^6$
C	Attack voting server for getting received encrypted ballots	AND	0.3135	0.0384	0.0092	$49.6 \cdot 10^6$	$13.976 \cdot 10^6$	$6.205 \cdot 10^6$
C.1	Developing malicious code for getting voter data and encrypted ballots		0.95	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$85.4 \cdot 10^6$
C.2	Inserting the code into server	OR	0.33	0.4	0.096	$40 \cdot 10^6$	$9.6 \cdot 10^6$	$13.368 \cdot 10^6$
C.2.1	Software developer is bribed		0.33	0.7	0.396	$70 \cdot 10^6$	$39.6 \cdot 10^6$	$-16.632 \cdot 10^6$
C.2.2	Server administrator is bribed		0.33	0.4	0.096	$40 \cdot 10^6$	$9.6 \cdot 10^6$	$13.368 \cdot 10^6$
C.2.3	Insecure configuration management is exploited		0.002	0.05	0.05	$5 \cdot 10^6$	$5 \cdot 10^6$	$-4.8 \cdot 10^6$
C.2.4	Control the connection between servers		0.15	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$5.4 \cdot 10^6$

Appendix 5

Node	Node name	Node type	p	q	q_{-}	π	π_{-}	Outcome
	Eligible voters cast votes more than once	AND	$5.4 \cdot 10^{-50}$	0.038	0.009	$149.6 \cdot 10^6$	$125.24 \cdot 10^6$	$-125.24 \cdot 10^6$
A	Voters vote more than once		$1.7 \cdot 10^{-46}$	0.999	0.999	$99.9 \cdot 10^6$	$99.9 \cdot 10^6$	$16.66 \cdot 10^6$
B	Developing malicious code for enabling vote more than once		0.95	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$85.4 \cdot 10^6$
C	Malicious code is smuggled into server	OR	0.33	0.400	0.096	$40 \cdot 10^6$	$9.6 \cdot 10^6$	$13.368 \cdot 10^6$
C.1	Software developer of server is bribed		0.33	0.7	0.396	$70 \cdot 10^6$	$39.6 \cdot 10^6$	$-16.632 \cdot 10^6$
C.2	Server administrator is bribed		0.33	0.4	0.096	$40 \cdot 10^6$	$9.6 \cdot 10^6$	$13.368 \cdot 10^6$
C.3	Insecure configuration management is exploited		0.001	0.05	0.05	$5 \cdot 10^6$	$5 \cdot 10^6$	$-4.9 \cdot 10^6$

Appendix 6

Node	Node name	Node type	p	q	q ₋	π	π ₋	Outcome
	Large-scale votes' disfranchisement before receiving votes	AND	$1.285 \cdot 10^{-5}$			$139.2 \cdot 10^6$	$48.433 \cdot 10^6$	$-48.433 \cdot 10^6$
A	Voter Application is injured	AND	$4.1013E-05$	0.077	0.009	$89.6 \cdot 10^6$	$19.2 \cdot 10^6$	$-19.198 \cdot 10^6$
A.1	Malicious code is downloaded unwillingly		$4.317 \cdot 10^{-5}$	0.8	0.096	$80 \cdot 10^6$	$9.6 \cdot 10^6$	$-9.598 \cdot 10^6$
A.2	Developing malicious code		0.95	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$85.4 \cdot 10^6$
B	Votes are eliminated before receiving	OR	0.314	0.038	0.009	$49.6 \cdot 10^6$	$19.93 \cdot 10^6$	$2.118 \cdot 10^6$
B.1	Redirecting voters to forged Network Server	AND	$1.353 \cdot 10^{-5}$	0.007	0.002	$99.2 \cdot 10^6$	$94.886 \cdot 10^6$	$-94.885 \cdot 10^6$
B.1.1	<i>Forged Network Server is developed</i>		0.95	0.8	0.2	$80 \cdot 10^6$	$20 \cdot 10^6$	$18 \cdot 10^6$
B.1.2	<i>Voters accept a forged Network Server certificate</i>		$4.317 \cdot 10^{-5}$	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$-9.595 \cdot 10^6$
B.1.3	<i>Voters connect to a forged Network Server</i>	OR	0.330	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$23.4 \cdot 10^6$
B.1.3.1	Voters click on wrong link		$2.672 \cdot 10^{-223}$	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$-9.6 \cdot 10^6$
B.1.3.2	Voters unwillingly download malicious code		$4.317 \cdot 10^{-5}$	0.8	0.096	$80 \cdot 10^6$	$9.6 \cdot 10^6$	$-9.598 \cdot 10^6$
B.1.3.3.1	Network configuration is affected		0.31	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$21.4 \cdot 10^6$
B.1.3.3.2	Local network administrator is bribed		0.33	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$23.4 \cdot 10^6$
B.2	Passively compromised Network Server	AND	0.314	0.038	0.009	$49.6 \cdot 10^6$	$19.93 \cdot 10^6$	$2.118 \cdot 10^6$
B.2.1	<i>Developing malicious code</i>		0.95	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$85.4 \cdot 10^6$
B.2.2	<i>Malicious code is smuggled into server</i>	OR	0.33	0.4	0.096	$40 \cdot 10^6$	$9.6 \cdot 10^6$	$13.368 \cdot 10^6$
B.2.2.1	Software developer of server is bribed		0.33	0.7	0.396	$70 \cdot 10^6$	$39.6 \cdot 10^6$	$-16.632 \cdot 10^6$
B.2.2.2	Server administrator is bribed		0.33	0.4	0.096	$40 \cdot 10^6$	$9.6 \cdot 10^6$	$13.368 \cdot 10^6$
B.2.2.3	Insecure configuration management is exploited		0.002	0.05	0.05	$5 \cdot 10^6$	$5 \cdot 10^6$	$-4.8 \cdot 10^6$

Appendix 7

Node	Node name	Node type	p	q	q _∞	π	π	Outcome
	Eliminating votes in two e-voting servers	AND	0.098	0.001	0.00008	99.2·10⁶	54.024·10⁶	-48.635·10⁶
A	Passively compromised Votes Storing Server	AND	0.314	0.038	0.009	49.6·10⁶	19.93·10⁶	2.118·10⁶
A.1	Developing malicious code		0.95	0.096	0.096	9.6·10⁶	9.6·10⁶	85.4·10⁶
A.2	Malicious code is smuggled into server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13.368·10⁶
A.2.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
A.2.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
A.2.3	Insecure configuration management is exploited		0.001	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.9·10 ⁶
B	Passively compromised Network Server	AND	0.314	0.038	0.009	49.6·10⁶	19.93·10⁶	2.118·10⁶
B.1	Developing malicious code		0.95	0.096	0.096	9.6·10⁶	9.6·10⁶	85.4·10⁶
B.2	Malicious code is smuggled into server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13.368·10⁶
B.2.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
B.2.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
B.2.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶

Appendix 8

Node	Node name	Node type	p	q	q ₋	π	π	Outcome
	Attack against the connection between Voter Application and Network Server for changing the ballots	AND	$4.1 \cdot 10^{-5}$	0.16	0.019	$100 \cdot 10^6$	$86.599 \cdot 10^6$	$-86.596 \cdot 10^6$
A	Forged server is developed		0.95	0.8	0.2	$80 \cdot 10^6$	$20 \cdot 10^6$	$18 \cdot 10^6$
B	Voters connect to forged server	OR	$4.317 \cdot 10^{-5}$	0.2	0.096	$20 \cdot 10^6$	$9.6 \cdot 10^6$	$-9.596 \cdot 10^6$
B.1	Voters click on wrong link		$2.67 \cdot 10^{-223}$	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$-9.6 \cdot 10^6$
B.2	Voters accept a forged Network Server certificate		$4.317 \cdot 10^{-5}$	0.2	0.096	$20 \cdot 10^6$	$9.6 \cdot 10^6$	$-9.596 \cdot 10^6$
B.3	Voters unwillingly download malicious code		$4.317 \cdot 10^{-5}$	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$-9.595 \cdot 10^6$
B.4	Access to voters' computers by using local network	AND	0.103	0.015	0.005	$35 \cdot 10^6$	$19.763 \cdot 10^6$	$-11.036 \cdot 10^6$
B.4.1	Malicious code is inserted to voters' computer or local network		0.312	0.05	0.05	$5 \cdot 10^6$	$5 \cdot 10^6$	$26.2 \cdot 10^5$
B.4.2	Local network administrator is bribed		0.33	0.3	0.096	$30 \cdot 10^6$	$9.6 \cdot 10^6$	$16.668 \cdot 10^6$

Appendix 9

Node	Node name	Node type	p	q	q ₋	π	π	Outcome
	Control over processes of Votes Storing Server of SERVE	AND	0.314	0.038	0.009	$49.6 \cdot 10^6$	$19.930 \cdot 10^6$	$2.118 \cdot 10^6$
A	Developing malicious code		0.95	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$85.4 \cdot 10^6$
B	Malicious code is smuggled into server	OR	0.33	0.4	0.096	$40 \cdot 10^6$	$9.6 \cdot 10^6$	$13.37 \cdot 10^6$
B.1	Software developer of server is bribed		0.33	0.7	0.396	$70 \cdot 10^6$	$39.6 \cdot 10^6$	$-16.64 \cdot 10^6$
B.2	Server administrator is bribed		0.33	0.4	0.096	$40 \cdot 10^6$	9.6	$13.37 \cdot 10^6$
B.3	Insecure configuration management is exploited		0.001	0.05	0.05	$5 \cdot 10^6$	$5 \cdot 10^6$	$-4.9 \cdot 10^6$

Appendix 10

Node	Node name	Node type	p	q	q ₁	π	π	Outcome
	Large-scale votes' adding in Votes Counting Server of SERVE	AND	0.093	14.2·10⁻⁵	8.153·10⁻⁶	108.8·10⁶	63.868·10⁶	-58.727·10⁶
A	Attack against Votes Storing Server	AND	0.314	0.038	0.009	49.6·10⁶	19.930·10⁶	2.118·10⁶
A.1	Developing malicious code		0.95	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	85400000
A.2	Malicious code is smuggled into server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13368000
A.2.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16632000
A.2.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13368000
A.2.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4800000
B	Create encrypted ballots		0.95	0.096	0.096	9.6·10⁶	9.6·10⁶	85.4·10⁶
C	Passively compromised Votes Counting Server	AND	0.314	0.038	0.009	49.6·10⁶	19.930·10⁶	2.118·10⁶
C.1	Developing an attacking code		0.95	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	85.4·10 ⁶
C.2	Malicious code is smuggled into server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13.368·10⁶
C.2.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16632000
C.2.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13368000
C.2.3	Insecure configuration management is exploited		0.001	0.05	0.05	5·10 ⁶	5·10 ⁶	-4900000

Appendix 11

Node	Node name	Node type	ρ	q	q_{-}	π	π	Outcome
	Identify how voters voted	OR	0.314	0.0576	0.009	69.6·10⁶	14.306·10⁶	-290755
A	Attack against Voter Application	OR	1.35·10⁻⁵	0.004	0.001	59.2·10⁶	37.867·10⁶	-37.866·10⁶
A.1	Voters use an actively compromised Network Server	AND	5.84·10⁻¹⁰	0.00070779	0.00018	108.8·10⁶	1.04·10⁸	-10.448·10⁷
A.1.1	<i>Voters deliberately download a forged ActiveX</i>		4.32·10 ⁻⁵	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	-9.595·10 ⁶
A.1.2	<i>Redirecting voters to a forged Network Server</i>	AND	1.35·10 ⁻⁵	0.007	0.002	99.2·10 ⁶	94.886·10 ⁶	-94.885·10 ⁶
A.1.2.1	Forge Network Server is developed		0.950	0.800	0.200	80·10 ⁶	2·10 ⁶	18·10 ⁶
A.1.2.2	Voters accept a forged Network Server certificate		4.32·10 ⁻⁵	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	-9.595·10 ⁶
A.1.2.3	Voters connect to a forged Network Server	OR	0.330	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	23.4·10 ⁶
A.1.2.3.1	Voters click on wrong link		2.7·10 ⁻²²³	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	-9.6·10 ⁶
A.1.2.3.2	Voters unwillingly download malicious code		4.32·10 ⁻⁵	0.800	0.096	80·10 ⁶	9.6·10 ⁶	-9.598·10 ⁶
A.1.2.3.3	Network configuration is affected		0.310	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	21.4·10 ⁶
A.1.2.3.4	Local network administrator is bribed		0.330	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	23.4·10 ⁶
A.2	Voters download a forged ActiveX from Network Server	AND	1.35·10⁻⁵	0.004	0.001	59.2·10⁶	37.867·10⁶	-37.866·10⁶
A.2.1	<i>Developing malicious code</i>		0.95	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	85.4·10 ⁶
A.2.2	<i>Voters have downloaded an untrustworthy Voter Application</i>		4.32·10 ⁻⁵	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	-9.595·10 ⁶
A.2.3	<i>Malicious code is smuggled into Network Server</i>	OR	0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
A.2.3.1	Software developer of Voter Application is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
A.2.3.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
A.2.3.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶
B	Attack against Network Server	AND	1.35E-05	0.031	0.001	129.6·10⁶	37.870·10⁶	-37.87·10⁶
B.1	Logging voters' actions with malicious code		4.32·10⁻⁵	0.8	0.096	80·10⁶	9.6·10⁶	-9.598·10⁶
B.2	Malicious code is developed		0.95	0.096	0.096	9.6·10⁶	9.6·10⁶	85.4·10⁶
B.3	Malicious code is smuggled into server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13.368·10⁶
B.3.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
B.3.2	Server administrator is bribed		0.33	0.4	0.096	40·10 ⁶	9.6·10 ⁶	13.368·10 ⁶
B.3.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶

C	Attack against Votes Storing Server	OR	0.314	0.0576	0.009	69.6·10⁶	14.306·10⁶	-290755
C.1	Attack against encrypted ballots	AND	0.116	0.046	0.001	149.6·10⁶	72.954·10⁶	-70.262·10⁶
<i>C.1.1</i>	<i>Logging ballots' random numbers with malicious code</i>		<i>0.368</i>	<i>0.8</i>	<i>0.096</i>	<i>80·10⁶</i>	<i>9.6·10⁶</i>	<i>1.283·10⁶</i>
<i>C.1.2</i>	<i>Developing malicious code</i>		<i>0.95</i>	<i>0.096</i>	<i>0.096</i>	<i>9.6·10⁶</i>	<i>9.6·10⁶</i>	<i>85.4·10⁶</i>
<i>C.1.3</i>	<i>Malicious code is inserted into Votes Storing Server</i>	<i>OR</i>	<i>0.33</i>	<i>0.6</i>	<i>0.096</i>	<i>60·10⁶</i>	<i>9.6·10⁶</i>	<i>6.768·10⁶</i>
<i>C.1.3.1</i>	<i>Software developer of server is bribed</i>		<i>0.33</i>	<i>0.7</i>	<i>0.396</i>	<i>70·10⁶</i>	<i>39.6·10⁶</i>	<i>-16.632·10⁶</i>
<i>C.1.3.2</i>	<i>Server administrator is bribed</i>		<i>0.33</i>	<i>0.6</i>	<i>0.096</i>	<i>60·10⁶</i>	<i>9.6·10⁶</i>	<i>6.768·10⁶</i>
<i>C.1.3.3</i>	<i>Insecure configuration management is exploited</i>		<i>0.002</i>	<i>0.05</i>	<i>0.05</i>	<i>5·10⁶</i>	<i>5·10⁶</i>	<i>-4.8·10⁶</i>
C.2	Attack against decrypted ballots	AND	0.314	0.0576	0.00922	69.6·10⁶	14.306·10⁶	-290755
<i>C.2.1</i>	<i>Developing malicious code</i>		<i>0.95</i>	<i>0.096</i>	<i>0.096</i>	<i>9.6·10⁶</i>	<i>9.6·10⁶</i>	<i>85.4·10⁶</i>
<i>C.2.2</i>	<i>Malicious code is inserted into server</i>	<i>OR</i>	<i>0.33</i>	<i>0.6</i>	<i>0.096</i>	<i>60·10⁶</i>	<i>9.6·10⁶</i>	<i>6.768·10⁶</i>
<i>C.2.2.1</i>	<i>Software developer of server is bribed</i>		<i>0.33</i>	<i>0.7</i>	<i>0.396</i>	<i>70·10⁶</i>	<i>39.6·10⁶</i>	<i>-16.632·10⁶</i>
<i>C.2.2.2</i>	<i>Server administrator is bribed</i>		<i>0.33</i>	<i>0.6</i>	<i>0.096</i>	<i>60·10⁶</i>	<i>9.6·10⁶</i>	<i>6.768·10⁶</i>
<i>C.2.2.3</i>	<i>Insecure configuration management is exploited</i>		<i>0.002</i>	<i>0.05</i>	<i>0.05</i>	<i>5·10⁶</i>	<i>5·10⁶</i>	<i>-4.8·10⁶</i>
D	Attack against Votes Counting Server	AND	0.116	0.031	0.001	129.6·10⁶	66.054·10⁶	-61.852·10⁶
D.1	Logging voters' actions with malicious code		0.368	0.8	0.096	80·10⁶	9.6·10⁶	1.283·10⁶
D.2	Developing malicious code		0.95	0.096	0.096	9.6·10⁶	9.6·10⁶	85.4·10⁶
D.3	Malicious code is inserted into Votes Counting Server	OR	0.33	0.4	0.096	40·10⁶	9.6·10⁶	13.368·10⁶
<i>D.3.1</i>	<i>Software developer of server is bribed</i>		<i>0.33</i>	<i>0.7</i>	<i>0.396</i>	<i>70·10⁶</i>	<i>39.6·10⁶</i>	<i>-16.632·10⁶</i>
<i>D.3.2</i>	<i>Server administrator is bribed</i>		<i>0.33</i>	<i>0.4</i>	<i>0.096</i>	<i>40·10⁶</i>	<i>9.6·10⁶</i>	<i>13.368·10⁶</i>
<i>D.3.3</i>	<i>Insecure configuration management is exploited</i>		<i>0.002</i>	<i>0.05</i>	<i>0.05</i>	<i>5·10⁶</i>	<i>5·10⁶</i>	<i>-4.8·10⁶</i>

Appendix 12

Node	Node name	Node type	p	q	q ₋	π	π ₋	Outcome
	Large-scale votes' buying	OR						-2.37·10⁶
A	Votes buying server	AND	0.4	0.64	0.64	160·10⁶	87.6·10⁶	-74.5·10⁶
A.1	Votes buying server and software are developed		0.95	0.8	0.8	80·10 ⁶	80·10 ⁶	15·10 ⁶
A.2	Voters connect to votes buying server		0.5	0.8	0.8	80·10 ⁶	80·10 ⁶	-30·10 ⁶
B	Spreading votes' receipt software	AND	0.147	0.0037	0.00089	169.19·10⁶	137.08·10⁶	-127.05·10⁶
B.1	Developing voting data saving software		0.8	0.096	0.096	9.6·10⁶	9.6·10⁶	70.4·10⁶
B.2	Voters use software for saving voting data		0.7	0.999	0.999	99.995·10⁶	99.995·10⁶	-29.99·10⁶
B.3	Attack voting server for getting received encrypted ballots	AND	0.264	0.048	0.0093	59.6·10⁶	17.72·10⁶	-2.37·10⁶
B.3.1	<i>Developing malicious code for getting voter data and encrypted ballots</i>		0.8	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	70.4·10 ⁶
B.3.2	<i>Inserting the code into server</i>	OR	0.33	0.5	0.096	50·10 ⁶	9.6·10 ⁶	10.068·10 ⁶
B.3.2.1	Software developer is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
B.3.2.2	Server administrator is bribed		0.33	0.5	0.096	50·10 ⁶	9.6·10 ⁶	10.068·10 ⁶
B.3.2.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶
B.3.2.4	Control the connection between servers		0.15	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	5.4·10 ⁶
C	Attack Votes Storing Server for getting voters' ballots	AND	0.264	0.048	0.0093	59.6·10⁶	17.72·10⁶	-2.37·10⁶
C.1	Developing malicious code for saving votes		0.8	0.096	0.096	9.6·10⁶	9.6·10⁶	70.4·10⁶
C.2	Inserting the code into server	OR	0.33	0.5	0.096	50·10⁶	9.6·10⁶	10.068·10⁶
C.2.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
C.2.2	Server administrator is bribed		0.33	0.5	0.096	50·10 ⁶	9.6·10 ⁶	10.068·10 ⁶
C.2.3	Insecure configuration management is exploited		0.002	0.05	0.1	5·10 ⁶	10·10 ⁶	-9.79·10 ⁶

Appendix 13

Node	Node name	Node type	p	q	q ₋	π	π	Outcome
	Control over processes of Votes Storing Server of SERVE	AND	0.314	0.048	0.009	59.6·10⁶	20.171·10⁶	-1.182·10⁶
A	Developing malicious code		0.95	0.096	0.096	9.6·10⁶	9.6·10⁶	85.4·10⁶
B	Malicious code is smuggled into server	OR	0.33	0.5	0.096	50·10⁶	9.6·10⁶	10.1·10⁶
B.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.64·10 ⁶
B.2	Server administrator is bribed		0.33	0.5	0.096	50·10 ⁶	9.6	10.1·10 ⁶
B.3	Insecure configuration management is exploited		0.001	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.9·10 ⁶

Appendix 14

Node	Node name	Node type	p	q	q ₋	π	π ₋	Outcome
	Identify how voters voted	OR	0.116	1.98·10⁻⁵	9.87·10⁻⁷	21.991·10⁶	9.525·10⁶	565226.7
A	Attack against Voter Application	AND	0.116	1.98·10⁻⁵	9.87·10⁻⁷	21.991·10⁶	9.525·10⁶	565226.7
A.1	Developing malicious code		0.95	0.01	0.01	995511.9	995512	94.004·10 ⁶
A.2	Voters have downloaded an untrustworthy Voter Application		0.368	0.01	0.01	995511.9	995512	35.774·10 ⁶
A.3	Malicious code is smuggled into Network Server	OR	0.33	0.2	0.01	20·10⁶	995512	25.733·10⁶
A.3.1	Software developer of Voter Application is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
A.3.2	Server administrator is bribed		0.33	0.2	0.01	20·10 ⁶	995512	25.733·10 ⁶
A.3.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶
B	Attack against Network Server	AND	0.116	0.0016	9.87·10⁻⁷	100.995·10⁶	39.761·10⁶	-35.292·10⁶
B.1	Logging voters' actions with malicious code		0.368	0.8	0.01	80·10 ⁶	995512	6.724·10 ⁶
B.2	Developing malicious code		0.95	0.01	0.01	995511.9	995512	94.004·10 ⁶
B.3	Malicious code is smuggled into server	OR	0.33	0.2	0.01	20·10⁶	995512	25.733·10⁶
B.3.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
B.3.2	Server administrator is bribed		0.33	0.2	0.01	20·10 ⁶	995512	25.733·10 ⁶
B.3.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶
C	Attack against Votes Storing Server	AND	0.116	0.0016	9.87·10⁻⁷	100.995·10⁶	39.761·10⁶	-35.292·10⁶
C.1	Logging voters' ballots' numbers with malicious code		0.368	0.8	0.01	80·10 ⁶	995512	6.724·10 ⁶
C.2	Developing malicious code		0.95	0.01	0.01	995511.9	995512	94.004·10 ⁶
C.3	Malicious code is inserted into Votes Storing Server	OR	0.33	0.2	0.01	20·10⁶	995512	25.733·10⁶
C.3.1	Software developer of server is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
C.3.2	Server administrator is bribed		0.33	0.2	0.01	20·10 ⁶	995512	25.733·10 ⁶
C.3.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶

Appendix 15

Node	Node name	Node type	p	q	q _u	π	π _u	Outcome
	Large-scale votes' buying	AND	0.2085			29.716·10⁶	17.767·10⁶	589026.03
A	Developing voting data saving software		0.95	0.001	0.001	99955.1	99955.1	94.9·10⁶
B	Voters use software for saving voting data		0.70	0.0952	0.0952	9516710.6	9516710.644	60.483·10⁶
C	Attack voting server for getting received encrypted ballots	AND	0.3135	0.0002	9.99·10⁻⁷	20.099·10⁶	468647.3	24.726·10⁶
C.1	Developing malicious code for getting voter data and encrypted ballots		0.95	0.001	0.001	99955.1	99955.1	94.9·10⁶
C.2	Inserting the code into server	OR	0.33	0.2	0.001	20·10⁶	99955.1	26333030
C.2.1	Software developer is bribed		0.33	0.7	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
C.2.2	Server administrator is bribed		0.33	0.2	0.001	20·10 ⁶	99955.1	26.34·10 ⁶
C.2.3	Insecure configuration management is exploited		0.002	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.8·10 ⁶
C.2.4	Control the connection between servers		0.15	0.001	0.001	99955.1	99955.1	14.9·10 ⁶

Appendix 16

Node	Node name	Node type	p	q	q _u	π	π _u	Outcome
	Eligible voters cast votes more than once	AND	0.284	7.61·10⁻⁵	1.85·10⁻⁶	42941114	21.591·10⁶	707899.3
A	Voters vote more than once		0.905	0.0198	0.0198	1.981·10⁶	1.981·10⁶	88.457·10⁶
B	Developing malicious code for enabling vote more than once		0.950	0.01	0.01	960000	995512	94.038·10⁶
C	Malicious code is smuggled into server	OR	0.330	0.400	0.01	40·10⁶	960000	19.156·10⁶
C.1	Software developer of server is bribed		0.330	0.700	0.396	70·10 ⁶	39.6·10 ⁶	-16.632·10 ⁶
C.2	Server administrator is bribed		0.330	0.400	0.01	40·10 ⁶	960000	19.156·10 ⁶
C.3	Insecure configuration management is exploited		0.001	0.05	0.05	5·10 ⁶	5·10 ⁶	-4.9·10 ⁶

Appendix 17

Node	Node name	Node type	p	q	q _u	π	π	Outcome
	Eliminating votes in two e-voting servers	AND	0.098	$9.83 \cdot 10^{-9}$	$9.83 \cdot 10^{-9}$	$3.982 \cdot 10^6$	$3.982 \cdot 10^6$	$5.846 \cdot 10^6$
A	Passively compromised Votes Storing Server	AND	0.314	$9.92 \cdot 10^{-5}$	$9.92 \cdot 10^{-5}$	1991024	1991024	$29.358 \cdot 10^6$
A.1	Developing malicious code		0.95	0.01	0.01	995512	995512	$94.004 \cdot 10^6$
A.2	Malicious code is smuggled into server	OR	0.33	0.01	0.01	995512	995512	$32.004 \cdot 10^6$
A.2.1	Software developer of server is bribed		0.33	0.7	0.396	$70 \cdot 10^6$	$39.6 \cdot 10^6$	$-16.632 \cdot 10^6$
A.2.2	Server administrator is bribed		0.33	0.01	0.01	995512	995512	$32.004 \cdot 10^6$
A.2.3	Insecure configuration management is exploited		0.001	0.05	0.05	$5 \cdot 10^6$	$5 \cdot 10^6$	$-4.9 \cdot 10^6$
B	Passively compromised Network Server	AND	0.314	$9.92 \cdot 10^{-5}$	$9.92 \cdot 10^{-5}$	1991024	1991024	$29.358 \cdot 10^6$
B.1	Developing malicious code		0.95	0.01	0.01	995512	995512	$94.004 \cdot 10^6$
B.2	Malicious code is smuggled into server	OR	0.33	0.01	0.01	995512	995512	$32.004 \cdot 10^6$
B.2.1	Software developer of server is bribed		0.33	0.7	0.396	$70 \cdot 10^6$	$39.6 \cdot 10^6$	$-16.632 \cdot 10^6$
B.2.2	Server administrator is bribed		0.33	0.01	0.01	995512	995512	$32.004 \cdot 10^6$
B.2.3	Insecure configuration management is exploited		0.002	0.05	0.05	$5 \cdot 10^6$	$5 \cdot 10^6$	$-4.8 \cdot 10^6$

Appendix 18

Node	Node name	Node type	p	q	q ₋	π	π ₋	Outcome
	Large-scale votes' disfranchisement before receiving votes	AND	0.11			$3.982 \cdot 10^6$	$3.982 \cdot 10^6$	$6.968 \cdot 10^6$
A	Voter Application is injured	AND	0.349	$9.92 \cdot 10^{-5}$	$9.92 \cdot 10^{-5}$	1991024	1991024	$32.94 \cdot 10^6$
A.1	Malicious code is downloaded unwillingly		0.368	0.01	0.01	995511.9	995512	$35.774 \cdot 10^6$
A.2	Developing malicious code		0.950	0.01	0.01	995511.9	995512	$94.004 \cdot 10^6$
B	Votes are eliminated before receiving	OR	0.314	$9.92 \cdot 10^{-5}$	$9.92 \cdot 10^{-5}$	1991024	$1.991 \cdot 10^{-5}$	$29.358 \cdot 10^{-5}$
B.1	Redirecting voters to forged Network Server	AND	$1.36 \cdot 10^{-5}$	$9.87 \cdot 10^{-7}$	$9.87 \cdot 10^{-7}$	$2.986 \cdot 10^{-5}$	$2.886 \cdot 10^{-5}$	$-2.885 \cdot 10^{-5}$
B.1.1	<i>Forged Network Server is developed</i>		0.95	0.01	0.01	995511.9	995512	$94.004 \cdot 10^6$
B.1.2	<i>Voters accept a forged Network Server certificate</i>		$4.32 \cdot 10^{-5}$	0.01	0.01	995511.9	995512	-991194.8
B.1.3	<i>Voters connect to a forged Network Server</i>	OR	0.330	0.01	0.01	995511.9	995512	$32.004 \cdot 10^6$
B.1.3.1	Voters click on wrong link		$2.68 \cdot 10^{-223}$	0.01	0.01	$1 \cdot 10^6$	$1 \cdot 10^6$	-1000000
B.1.3.2	Voters unwillingly download malicious code		0.368	0.8	0.096	$80 \cdot 10^6$	$9.6 \cdot 10^6$	$1.283 \cdot 10^6$
B.1.3.3.1	Network configuration is affected		0.310	0.096	0.096	$9.6 \cdot 10^6$	$9.6 \cdot 10^6$	$21.4 \cdot 10^6$
B.1.3.3.2	Local network administrator is bribed		0.330	0.01	0.01	995511.9	995512	$32.004 \cdot 10^6$
B.2	Passively compromised Network Server	AND	0.314	$9.92 \cdot 10^{-5}$	$9.92 \cdot 10^{-5}$	$1.991 \cdot 10^{-5}$	$1.991 \cdot 10^{-5}$	$29.358 \cdot 10^{-5}$
B.2.1	<i>Developing malicious code</i>		0.950	0.01	0.01	995512	995512	$94.004 \cdot 10^6$
B.2.2	<i>Malicious code is smuggled into server</i>	OR	0.330	0.01	0.01	995512	995512	$32.004 \cdot 10^6$
B.2.2.1	Software developer of server is bribed		0.330	0.7	0.396	$70 \cdot 10^6$	$39.6 \cdot 10^6$	$-16.632 \cdot 10^6$
B.2.2.2	Server administrator is bribed		0.330	0.01	0.01	995511.9	995512	$32.004 \cdot 10^6$
B.2.2.3	Insecure configuration management is exploited		0.002	0.05	0.05	$5 \cdot 10^6$	$5 \cdot 10^6$	$-4.8 \cdot 10^6$

Appendix 19

Node	Node name	Node type	p	q	q ₋	π	π ₋	Outcome
	Attack against the connection between Voter Application and Network Server for changing the ballots	AND	0.342	0.001	0.001	28.8·10⁶	28.8·10⁶	5.4·10⁶
A	Forged server is developed		0.95	0.096	0.096	9.6·10⁶	9.6·10⁶	85.4·10⁶
B	Voters connect to forged server	OR	0.36	0.009	0.009	19.2·10⁶	19.2·10⁶	16.8·10⁶
B.1	Voters click on wrong link		2.67·10⁻²²³	0.096	0.096	9.6·10⁶	9.6·10⁶	-9.6·10⁶
B.2	Voters accept a forged Network Server certificate		4.317·10⁻⁵	0.096	0.096	9.6·10⁶	9.6·10⁶	-9.595·10⁶
B.3	Voters unwillingly download malicious code		4.317·10⁻⁵	0.096	0.096	9.6·10⁶	9.6·10⁶	-9.595·10⁶
B.4	Access to voters' computer by using local network	AND	0.36	0.009	0.009	19.2·10⁶	19.2·10⁶	16.8·10⁶
B.4.1	Malicious code is inserted to voters' computer or local network		0.6	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	50.4·10 ⁶
B.4.2	Local network administrator is bribed		0.6	0.096	0.096	9.6·10 ⁶	9.6·10 ⁶	50.4·10 ⁶

Appendix 20

Node	Node name	Node type	p	q	q ₋	π	π	Outcome
	Large-scale votes' adding in Votes Counting Server of SERVE	AND	0.309	$9.51 \cdot 10^{-11}$	$9.78 \cdot 10^{-11}$	$4.951 \cdot 10^6$	$4.948 \cdot 10^6$	$25.916 \cdot 10^6$
A	Attack against Votes Storing Server	AND	0.57	$9.96 \cdot 10^{-5}$	$9.912 \cdot 10^{-5}$	$1.995 \cdot 10^6$	$1.991 \cdot 10^6$	$55.006 \cdot 10^6$
A.1	Developing malicious code		0.95	0.01	0.01	995511.9	995512	$94.004 \cdot 10^6$
A.2	Malicious code is smuggled into server	OR	0.6	0.010	0.01	$1 \cdot 10^6$	995512	$59.001 \cdot 10^6$
A.2.1	Software developer of server is bribed		0.33	0.7	0.396	$70 \cdot 10^6$	$39.6 \cdot 10^6$	$-16.632 \cdot 10^6$
A.2.2	Server administrator is bribed		0.6	0.01	0.01	$1 \cdot 10^6$	995512	$59.001 \cdot 10^6$
A.2.3	Insecure configuration management is exploited		0.002	0.05	0.05	$5 \cdot 10^6$	$5 \cdot 10^6$	$-4.8 \cdot 10^6$
B	Create encrypted ballots		0.95	0.01	0.01	995512	995512	$94.004 \cdot 10^6$
C	Passively compromised Votes Counting Server	AND	0.57	$9.6 \cdot 10^{-5}$	$9.92 \cdot 10^{-5}$	$196 \cdot 10^4$	$1.959 \cdot 10^6$	$55.04 \cdot 10^6$
C.1	Developing an attacking code		0.95	0.01	0.01	960000	995512	$94.038 \cdot 10^6$
C.2	Malicious code is smuggled into server	OR	0.6	0.01	0.01	$1 \cdot 10^6$	995512	$59.001 \cdot 10^6$
C.2.1	Software developer of server is bribed		0.33	0.7	0.396	$70 \cdot 10^6$	$39.6 \cdot 10^6$	$-16.632 \cdot 10^6$
C.2.2	Server administrator is bribed		0.6	0.01	0.01	$1 \cdot 10^6$	995512	$59.001 \cdot 10^6$
C.2.3	Insecure configuration management is exploited		0.001	0.05	0.05	$5 \cdot 10^6$	$5 \cdot 10^6$	$-4.9 \cdot 10^6$

References

- [1] D. Jefferson, A.D. Rubin, B. Simons, D. Wagner. A Security Analysis of the Secure Electronic Registration and Voting Experiment (SERVE). 2004. <http://www.servesecurityreport.org/>. 21.01.2007.
- [2] A. Ansper, A. Buldas, M. Oruaas, J. Piirsalu, A. Veldre, J. Willemson, K. Kivinurm. The Security of Conception of E-voting: Analysis and Measures. E-hääletamise kontseptsiooni turve: analüüs ja meetmed. 2003. <http://www.vvk.ee/elektr/docs/Analüüs-01.pdf>. 21.01.2007.
- [3] A. Buldas, P. Laud, J. Piirsalu, M. Saarepera, J. Willemson. Rational Choice of Security Measures via Multi-Parameter Attack Trees. In Critical Information Infrastructured Security First International Workshop - CRITIS 2006, LNCS 4347, pp. 235-248, 2006.
- [4] Estonian National Electoral Committee home page, www.vvk.ee. 21.01.2007.
- [5] Estonian National Electoral Committee. General Description of the E-voting System. E-hääletamise süsteemi üldkirjeldus. 2004. <http://www.vvk.ee/elektr/docs/Yldkirjeldus-02.pdf>. 21.01.2007.
- [6] Estonian National Electoral Committee. Data Structures of the E-voting System. E-hääletamise süsteem: Andmestruktuurid. 2006. <http://www.vvk.ee/elektr/docs/siseprotokollid-1.6.pdf>. 21.01.2007.
- [7] Estonian National Electoral Committee. Overview of the Model of Use Cases of the E-voting System. Kasutusloomudeli ülevaade. 2004. <http://www.vvk.ee/elektr/docs/kasutusloomudeli-ylevaade-1.1.pdf>. 21.01.2007.
- [8] Estonian National Electoral Committee. Essential Use Cases of the E-voting System. Olulisemad kasutuslood eraldi dokumentidena. <http://www.vvk.ee/elektr/index.html>. 21.01.2007.
- [9] T. Martens. Organizational and Technical Conception of the E-voting. E-hääletamise organisatsiooniline ja tehniline kontseptsioon. 2003. <http://www.vvk.ee/elektr/docs/Kontsept-03.pdf>. 21.01.2007.
- [10] B. Schneier. Attack Trees. Dr. Dobb's Journal December 1999. Modeling Security Threats. <http://www.schneier.com/paper-attacktrees-ddj-ft.html>. 28.01.2007.
- [11] D. Geer, K. Soo Hoo, A. Jaquith. Information Security: Why the Future Belongs to the Quants. IEEE Security and Privacy. 2003. pages 32-40.

- [12] Interview with the public prosecutor Mr. Margus Kurm.
- [13] Research Center Faktum & Ariko. The e-voting and diminishing alienation: The summary of the result of the public poll. 2004
E-valimised ja võõrandumise vähendamine. Kokkuvõte küsitlustulemustest. 2004. www.riigikogu.ee/doc.php?37557. 21.01.2007.
- [14] M. Surf, A. Shulman. How safe is it out there? Zeroing in on the vulnerabilities of application security. Imperva Application Defense Center. 2004. <http://www.imperva.com/application-defense-center/papers/how-safe-is-it.html>. 21.01.2007.
- [15] The elections' atlas of the United States of America. <http://www.uselectionatlas.org/>. 21.01.2007.
- [16] The Parlimental Elections in Estonia in 2003. http://www.vvk.ee/r03/yld_kulud.stm. 21.01.2007.
- [17] Department of Defense Washington Headquarters Services Federal Voting assistance Program. Voting Over the Internet Pilot Project Assessment Report. 2001. <http://www.fvap.gov/services/voireport.pdf>. 21.01.2007.
- [18] Evaluation report. Experiment with Internet and telephone voting for voters abroad. http://www.minbzk.nl/bzk2006uk/subjects/constitution_and/publications?ActItemIdt=13253. 21.01.2007.
- [19] T. Kohno, A. Stubblefield, A. D. Rubin, D. S. Wallach. Analysis of an Electronic Voting System. 2004. <http://avirubin.com/vote.pdf>. 21.01.2007.
- [20] Local Government Association. The Implementation of Electronic Voting in the UK research summary. 2002. <http://www.dca.gov.uk/elections/e-voting/pdf/e-summary.pdf>. 21.01.2007.
- [21] M. G. Newkirk. US Public Opinion toward Voting Technologies. InfoSENTRY Services. 2004. http://www.infosentry.com/US_Public_Opinion_Toward_Voting_Technology_20040301.pdf. 21.01.2007.
- [22] Site officiel de l'Etat de Genève. <http://www.geneve.ch/evoting/english/welcome.asp>. 21.01.2007.
- [23] The home page of the Estonian ID-card. <http://www.id.ee/pages.php/03020705>. 27.01.2007.