

DUTCH E-VOTING OPPORTUNITIES

RUUD VERBIJ

Risk assessment framework based on attacker resources

Services, Cyber Security and Safety group

EEMCS

University of Twente

August 2014

Ruud Verbij: *Dutch e-voting opportunities*, Risk assessment framework
based on attacker resources, © August 2014

SUPERVISORS:

Dr. ir. Wolter Pieters

Prof. dr. Roel Wieringa

Ir. Erwin Hansen

ABSTRACT

The state-of-art on e-voting research and publications does not fulfill the needs in feeding the current e-voting debates in the Netherlands. Whereas most of the scientific literature is focused on highly theoretical environments in which e-voting schemes operate, the majority of the more practical research does not provide for quantified and practical results in a realistic setting. This research, however, fills this gap by establishing a quantified framework for reviewing and objectively comparing e-voting schemes in practice.

The proposed framework in this research first establishes an exhaustive list of all possible attacks on the e-voting scheme, beyond initial traditional research into the protocol, the cryptography and the implementation. Subsequently, the framework addresses each of these identified attacks in terms of effort for the attacker: how much time and money does an attacker need to pull off the attack? Thereafter the attacks are categorized according to the Dutch requirements for voting schemes, after which they can be compared to either a baseline proposed by politics or to other schemes. The final step of the framework helps in mitigating the attack vectors and assessing the impact of differences in implementation details of these schemes. As the framework is circular, all steps can be repeated to allow for a thorough analysis and mitigation of potential risks.

This thesis also presents a limited case analysis on the Estonian e-voting scheme in order to show how the framework can be used in practice. Results already show astonishing attacks, by means of which it would only cost \$40,000 to get one seat in the Estonian parliament.

Both the framework and the case analysis have been validated by three professionals in the discipline of IT Security; e-voting research; and the Dutch voting practice. While a few improvement points were identified, the framework is considered to be a strong method for realistic risk identification in e-voting schemes. Quantifying these risks in terms of effort for an attacker allows for effective risk management and strong mitigation strategies based on cost-benefit considerations. Especially the adaptive attacker model, the modular approach and the ability to test the effectiveness of implementation details are very well received.

As a conclusion, this framework provides for an effective and structured additional research method when deciding to adopt e-voting for elections. Furthermore, this research fuels the current debate about e-voting in the Netherlands, thereby reaching the initial goal of this research.

Those who vote decide nothing. Those who count the vote decide everything.

— Joseph Stalin [1923]

ACKNOWLEDGEMENTS

As a child I was fascinated by the fact that people could write each other notes that could not be read by anyone but the sender and the receiver. The earliest memory I have, is from a novel in which a child writes letters to a pen-pal, encrypted using the Caesar cipher with the key being the addition of the separate digits of the date on the letter. Of course I started to do the same with different friends in class too, though, soon to find out the shortcoming of such an elementary system: everybody who knew that the characters were shifted in the alphabet had only to try 25 options to be able to read a note (our knowledge of character frequency was quite limited at the age of 8) – not recognizing the power of the Kerckhoff's principle just yet. From that moment I can clearly remember being fascinated by ciphers and started to read whatever I could get my hands on (and could understand). By the age of 18 I ended my high school career with a report on encryption throughout the time, starting at Caesar and ending with RSA, including small scripts to break e.g. Vigenère ciphers.

I could not have ended up at a better place to do my masters than at the Kerckhoffs' Institute, combining the knowledge of University of Twente, the Eindhoven University of Technology and the Radboud University of Nijmegen. The master taught me a lot about cryptography, formal methods for protocol verification, software security, network security, organizational security, hardware security, mobile and wireless security, cyber crime, law and ethics, and much more. But above all, it taught me to *"think thief"*, being the most important toolkit of any security engineer.

For my graduation project I ended up with the Information Protection Services business unit within KPMG The Netherlands, being the perfect place to combine both the knowledge I gained from the courses I took and the eager I have to learn so much more about IT Security.

After having reviewed the topics I had learned by heart while studying for the exams, it came to me that I enjoyed the fact that the master combines both the technical and the social side of IT Security the most. It was exactly this combination that brought me to a topic that contains both these sides: e-voting. Besides being a topic which everybody has an opinion about – which I would find out during my thesis to be really helpful – it is a topic having a lot to do with ethics and law, containing social and political sides, needing security

at levels reaching from hardware to software to cryptography to networks to protocols to organizational security up until the point that you need to really *think thief* again to understand it from an attacker point of view. This is what kept me interested in the topic throughout my entire graduation project.

This thesis could not have been finished without the extensive help of my supervisors at the University of Twente, Wolter Pieters and Roel Woeringa, and from KPMG, Erwin Hansen for which I thank them all. Besides supervision of Wolter, Roel and Erwin, I also want to thank all the participants of the interviews I conducted within the political playing field of e-voting in the Netherlands: Rop Gonggrijp, Pamela Young, Jan-Jouke Vos and Jan Smit, without their willingness to participate, this thesis would definitely not have reached the current depth. Furthermore, I have had plenty of support from reviewers like Sander, Thijs, Jeroen, Jurriën and Rick, which helped me to improve my thesis a lot, a really warm “thank you” to them too.

I also want to thank KPMG for the possibility they offered to do my graduation at their office in Amstelveen and providing me with a lot of help and contacts.

Last but definitely not least, I would like to thank all those who helped me structure my thoughts and proofread my work, providing me with a lot of tough questions and new insights.

I hope you have as much fun reading my thesis as I had writing it.

Ruud Verbij

CONTENTS

1	INTRODUCTION	1
1.1	Problem statement	1
1.2	Research questions and methods	3
1.3	Framework characteristics	4
1.3.1	Defining attacker effort	4
1.3.2	Comparison with regular risk calculations	6
1.4	Framework requirements	7
1.5	Contribution of this research	7
1.6	Structure of this thesis	8
2	STATE-OF-ART	9
2.1	Background on the Dutch voting system	9
2.2	E-voting in Computer Science literature	11
2.2.1	Privacy-related terminology	12
2.2.2	Verification-related terminology	14
2.2.3	Conclusion	16
2.3	Underground economy	17
2.3.1	Research into the Russian Underground	18
2.3.2	Research on DDoSing Serbia	20
2.3.3	Research on Crimeware-As-A-Service	21
2.3.4	Research on Cybercrime-As-A-Service	22
2.3.5	Research on Pay-Per-Install (PPI)	23
2.3.6	Zero days	24
2.3.7	Conclusions	24
2.4	Attack trees	26
3	ELECTION SAFEGUARDS	29
3.1	Dutch approach to safeguards	29
3.2	Information Security approach to safeguards	30
3.2.1	Confidentiality	30
3.2.2	Integrity	30
3.2.3	Availability	31
3.2.4	Authenticity	31
3.2.5	Non-repudiation	31
3.2.6	Accountability	31
3.3	Information security compared to the Dutch safeguards	32
3.3.1	Login credentials	32
3.3.2	Number of cast votes per voter	32
3.3.3	Voting systems	32
3.4	Conclusion	33
4	CURRENT VOTING SCHEMES	35

4.1	Estonia	35	
4.1.1	Scheme	36	
4.1.2	Mobile e-voting	37	
4.1.3	Verification mechanism	38	
4.1.4	Incidents	39	
4.1.5	Statistics	40	
4.1.6	Conclusion	40	
4.2	France	41	
4.2.1	Scheme	41	
4.2.2	Verification mechanism	43	
4.2.3	Statistics	44	
4.2.4	Conclusion	44	
4.3	Norway	45	
4.3.1	Scheme	45	
4.3.2	Verification mechanism	47	
4.3.3	Statistics	48	
4.3.4	Conclusions	48	
4.4	Other countries piloting e-voting	49	
4.4.1	Australia	49	
4.4.2	Canada	49	
4.4.3	India	50	
4.4.4	Mexico	50	
4.4.5	United Kingdom	50	
4.4.6	Switzerland	50	
4.5	Findings and conclusion	51	
5	PROPOSED FRAMEWORK	53	
5.1	Actors	53	
5.1.1	Attackers	53	
5.1.2	Voters	54	
5.1.3	Election authorities	55	
5.1.4	IT Staff	55	
5.1.5	Auditors (or other external observers)	55	
5.1.6	Bribing or coercing in general	57	
5.2	Attacker model	57	
5.2.1	Adaptive attacker model	58	
5.3	Framework	59	
5.3.1	Find	60	
5.3.2	Quantify	68	
5.3.3	Categorize	79	
5.3.4	Compare	81	
5.3.5	Mitigate	83	
5.4	Conclusion	84	
6	FRAMEWORK APPLIED	85	
6.1	Find attacks	85	
6.1.1	Assets & Properties	86	

6.1.2	Processes	89
6.1.3	Workflow	90
6.1.4	Attack flow	92
6.2	Quantify	95
6.2.1	Attack on vote integrity	95
6.2.2	Attack on secret suffrage	103
6.3	Categorize, compare and mitigate	107
6.3.1	Categorize	107
6.3.2	Compare	108
6.3.3	Mitigate	108
6.4	Conclusion and reflection	109
7	FRAMEWORK VERIFIED	111
7.1	Interview setup	111
7.2	Interview results	112
7.3	Requirement verification	115
7.4	Conclusion	116
8	CONCLUSION	119
8.1	Safeguards	119
8.2	Schemes	120
8.3	Framework	120
8.3.1	Find-step	120
8.3.2	Quantify-step	121
8.3.3	Categorize-step	121
8.3.4	Compare-step	121
8.3.5	Mitigation-step	122
8.4	Verification	122
8.5	Conclusion	123
9	DISCUSSION AND FUTURE WORK	125
9.1	Discussion	125
9.1.1	Efficiency of the find step	125
9.1.2	Comparing cost functions	125
9.1.3	Limited case study	126
9.1.4	Steps in premature state	126
9.1.5	Time investments	126
9.1.6	Estimates and statistics	127
9.1.7	Conclusion	127
9.2	Future work	127
9.2.1	Case study	127
9.2.2	Advances on intermediate and server side attacks	127
9.2.3	Bribes and coercion	127
9.2.4	Determining baseline	128
9.2.5	Advancements categorize and mitigate steps	128

A	INTERVIEWS	129
A.1	Vereniging Nederlandse Gemeenten (Association of Dutch Municipalities)	129
A.2	Kiesraad (Electoral Council Secretariat)	130
A.3	Nederlandse Vereniging Voor Burgerzaken (Dutch Association of Civil Affairs)	132
A.4	Wij vertrouwen stemcomputers niet (Pressure group "We don't trust voting computers")	134
A.5	Questions used for the interviews	138
B	APPROACHES TO SAFEGUARDS	141
B.1	International approach to safeguards	141
B.1.1	International laws and regulations	141
B.1.2	International safeguards	142
B.1.3	Conclusion	143
B.2	Literature approach to safeguards	144
B.2.1	Norwegian e-voting project	144
B.2.2	Threat analysis	146
B.2.3	Ethics of e-voting	149
B.2.4	Functional requirements	151
B.2.5	Conclusions on compliance with e-voting literature	151
C	CRYPTOGRAPHY	153
C.1	Encryption	153
C.1.1	Symmetric encryption	153
C.1.2	Asymmetric encryption	154
C.1.3	Probabilistic encryption	155
C.1.4	Homomorphic encryption	155
C.1.5	Non-malleable encryption	156
C.1.6	Threshold encryption	156
C.1.7	Commitments	157
C.2	Zero-knowledge proofs	157
C.3	Cryptographic systems within e-voting	159
C.3.1	Two agents	159
C.3.2	Double envelope	159
C.3.3	Homomorphic tallying	161
C.3.4	Mix-nets	161
C.3.5	Pollsterless schemes	162
C.4	Conclusion	163
D	EXPERT INTERVIEWS FOR VALIDATION	165
D.1	IT Security professional	165
D.2	Dutch voting practice professional	167
D.3	E-voting research professional	169
	BIBLIOGRAPHY	173

LIST OF FIGURES

Figure 1.1	Risk plotted on likelihood and impact	6
Figure 2.1	Verifiability types compared	16
Figure 2.2	Partial screenshot from the underground economy	17
Figure 2.3	Example of an attack tree	26
Figure 2.4	Event tree in the attacker game	27
Figure 4.1	Estonian e-voting scheme (simplified)	36
Figure 4.2	Estonian verification mechanism (simplified)	38
Figure 4.3	Estonian voting and verification mechanism (simplified)	39
Figure 4.4	France e-voting scheme (simplified)	42
Figure 4.5	Norwegian e-voting scheme (simplified)	45
Figure 5.1	Proposed framework	59
Figure 5.2	The framework, with the detailed steps for Find	61
Figure 5.3	Concepts of the methodological approach to procedural security analysis	62
Figure 5.4	Asset-flow in a procedural security analysis	62
Figure 5.5	Replace process	63
Figure 5.6	Snippet of NuSMV source code	65
Figure 5.7	Example of found attacks in workflow	66
Figure 5.8	The framework, with the detailed steps for Find	67
Figure 5.9	The framework, with the detailed steps for Quantify	68
Figure 5.10	Three domains of locations in the voting process	70
Figure 5.11	Example of an attack tree	71
Figure 5.12	Time frame example	73
Figure 5.13	Estonian votes per hour during voting days	75
Figure 5.14	An example of a C&C-infrastructure.	77
Figure 5.15	An example of a quantified attack tree	78
Figure 5.16	Proposed framework	79
Figure 5.17	Proposed framework	82
Figure 5.18	Proposed framework	83
Figure 5.19	Complete framework including substeps	84
Figure 6.1	The framework, with the detailed steps for Find	86
Figure 6.2	Estonian e-voting scheme (simplified)	87
Figure 6.3	Conceptual visualization of the voting application	88
Figure 6.4	Conceptual visualization of the eID	88
Figure 6.5	Conceptual visualization of the assets	89
Figure 6.6	Attacker flow of confidentiality attack	93

Figure 6.7	The framework, with the detailed steps for Quantify	95
Figure 6.8	The attack tree of the integrity attack	96
Figure 6.9	Attack on the Estonian e-voting scheme (simplified)	97
Figure 6.10	Days decorated attack tree for the integrity attack	98
Figure 6.11	Timeframe for the integrity attack	99
Figure 6.12	Estonian votes received per hour	100
Figure 6.13	Estonian votes received per day	101
Figure 6.14	Completely decorated attack tree for the integrity attack	102
Figure 6.15	Effort as a function of the number of votes in an Estonian integrity hack	102
Figure 6.16	The attack tree of the confidentiality attack	104
Figure 6.17	Days decorated attack tree for the confidentiality attack	105
Figure 6.18	Completely decorated attack tree for the confidentiality attack	106
Figure 6.19	Effort as a function of the number of votes published per hour in an Estonian confidentiality hacking example	107
Figure 7.1	Venn-diagram explaining expertises covered by the experts.	112
Figure 8.1	Complete framework including substeps	120
Figure C.1	Two agents	160
Figure C.2	Double envelope voting system	160
Figure C.3	Homomorphic tallying	161
Figure C.4	Mix-nets	162
Figure C.5	Pollsterless scheme mCESG	163

LIST OF TABLES

Table 1.1	Research methods	5
Table 2.1	Postal vs internet voting safeguard compliance according to The Election Process Advisory Commission	11
Table 2.2	Costs of a crypter according to Trend Micro	18
Table 2.3	Costs of hosting according to Trend Micro	19
Table 2.4	Costs of Pay-Per-Install Services according to Trend Micro	19
Table 2.5	Costs of DDoS according to Trend Micro	20
Table 2.6	Costs of spamming according to Trend Micro	20
Table 2.7	Cost of crypters and obfuscators according to Sood and Enbody	21
Table 2.8	Estimated cost of botnet components according to Sood and Enbody	22
Table 2.9	Estimated cost of browser exploit packs (BEP) according to Sood and Enbody	23
Table 2.10	Estimated cost of cybercrime components according to FortiGuard Labs	23
Table 2.11	Prices of zero-days in the field	24
Table 2.12	Average prices in the underground economy	25
Table 3.1	The Election Process Advisory Commission vs Information Security view on e-voting safeguards	34
Table 4.1	Privacy and verification properties for international e-voting schemes compared	52
Table 4.2	Voter specification for international e-voting schemes	52
Table 5.1	Time constraints for the attacker model	74
Table 6.1	Items from the underground economy needed for the integrity attack	101
Table 7.1	Verification of the framework requirements	116
Table B.2	Comparison of requirements versus threats in e-voting according to Volkamer and Grimm	148
Table B.3	Volkamer and Grimm vs The Election Process Advisory Commission on election safeguards	149

INTRODUCTION

This introductory chapter will introduce both the problem statement and proposed research method for this thesis in sections 1.1 and 1.2 respectively. It will describe the characteristics and requirements for the solution in sections 1.3 and 1.4. Furthermore it will state the contribution of the research performed in section 1.5, and it will provide a structure for this thesis in section 1.6.

1.1 PROBLEM STATEMENT

Currently, voting in the Netherlands happens either by posting a physical ballot in a voting booth situated in a polling station, or by postal voting for Dutch citizens living or working abroad. This has been the manner since controversy about electronic ways of voting struck both voting machines at polling stations (in 2006) [83] and internet voting (in 2008) [42]. Up until then, those two additional voting methods (internet voting and voting machines at polling stations) were established voting methods among Dutch citizens.

*Background on
e-voting*

Following this decision, the Dutch government appointed two committees to investigate the past and the future of voting methods, respectively. The latter committee proposed eight requirements (called safeguards from now on) for future voting methods [103]: transparency; verifiability; fairness; eligibility to vote; free suffrage; secret suffrage; equal suffrage and accessibility. In conversations held with the author of this thesis, different people and organizations concerned with the Dutch voting system¹ confirmed these safeguards to be the requirements for *any* election debate. It is generally considered however, that no voting system can guarantee all these safeguards completely (e.g. by the committee themselves [103, p21] and by van Cleeff et al. [107, p5]).

The currently deployed voting methods are being discussed by several governmental organizations² and within parliament³, claiming that it is time to review these, by government deprecated, voting

¹ Interviews, conducted by the author of this thesis, were held with Pamela Young and Jan-Jouke Vos, members of the secretariat of the Electoral Council; Jan Smit, chairman of the committee Elections with the “Dutch Association for Civil Concerns” (NVVB); Rop Gonggrijp, chairman of the Pressure Group “We don’t trust voting computers”. The interviews can be found in appendix A

² For example by the NVVB:

<https://www.nvnb.nl/sites/default/files/nvnb/2013%20B&R%20nr%201%20stemcomputers%20in%20beeld.pdf>

³ Bill filed by Joost Taverne of the VVD:

<https://zoek.officielebekendmakingen.nl/kst-33354-3.html>

methods again. One of the arguments posed by the proponents of e-voting is that the participation level of citizens would become higher, especially for Dutch citizens living abroad ⁴ as can be found in appendix A. To be able to review these voting methods objectively, the safeguards presented by The Election Process Advisory Commission [103] form the base of the discussion.

Literature view

E-voting is not only a topic of interest to people and organizations concerned with elections, it is also a well studied topic in academic literature. Current research in the area of e-voting within computer science focuses around either privacy-related or verification-related topics, according to an extensive survey by Jonker et al. [60]. The main methods used in literature for reviewing these voting schemes is by formally analyzing their properties. This resulted in different notions of privacy and verification offered by the different voting schemes. The safeguards presented by the Dutch committee (The Election Process Advisory Commission [103]) show a much broader spectrum of safeguards than just privacy and verifiability though.

Current focus of research on e-voting schemes in real world settings [110, 38] on the other hand, lacks precision with respect to risks. Risks are assessed in categories like low / medium / high, while their expected likelihood is assessed based only on the complexity of pulling such an attack off. Attacker models and attacker resources to actually achieve such levels of complexity are not taken into account in such research.

Problem statement

The current discussion in the Netherlands about e-voting is split into advocates and opponents, who both discuss the security implications of e-voting schemes differently, as can be seen from the interviews in appendix A. Advocates wonder why e-voting security requirements are any different from those of e-banking (which they consider safe), while opponents consider the democratic process too fragile to leave to the digital world by referring e.g. to the current news about the NSA and banking fraud that is widespread. As can be concluded from the interviews, there is no common ground on which these two parties can objectively discuss the security implications posed by e-voting. Even worse, the safeguards, as presented by The Election Process Advisory Commission do not allow for a

Debate in the House of Representatives:

<https://zoek.officielebekendmakingen.nl/h-tk-20122013-58-4.html>

- ⁴ Currently, about 5% of 700,000 eligible voters abroad take part in elections, which is much lower than 74.6% participation from Dutch citizens living in the Netherlands. Sources on the 2012 national parliament elections: <http://www.rijksoverheid.nl/bestanden/documenten-en-publicaties/rapporten/2013/11/11/bijlage-rapportage-kiezers-buiten-nederland/bijlage-rapportage-kiezers-buiten-nederland.pdf>, <https://www.kiesraad.nl/nieuws/offici%C3%ABle-uitslag-tweede-kamerverkiezing-12-september-2012> and The Election Process Advisory Commission [103, footnote 37].

quantitative comparison of (e-)voting schemes. Their descriptions are very broad and lack precision compared to current research in computer science. To allow such a quantitative comparison of e-voting schemes, it is necessary to formalize the safeguards presented by [The Election Process Advisory Commission](#). Not only the current debate in the Netherlands about e-voting could benefit from these formalized safeguards, the academic world would also get a broader view than the current focus on privacy and verifiability.

The problem to be solved for this research is the lack of a quantified method of assessing e-voting schemes in real world settings.

This research will address this issue by presenting a framework that can be used to quantitatively describe the risks posed by an e-voting scheme. Different e-voting schemes pose different risks, which can then be objectively compared to enhance the public debate in the Netherlands (or other countries).

The following sections will address the research questions and methods, the characteristics of the designed framework, the requirements for this framework, the contribution of this research and the structure of this thesis in sections [1.2](#), [1.3](#), [1.4](#), [1.5](#) and [1.6](#) respectively.

1.2 RESEARCH QUESTIONS AND METHODS

The goal of this research is to *improve the current internet voting debate in the Netherlands, by establishing a quantified framework for reviewing and objectively comparing internet voting schemes according to the safeguards presented by [The Election Process Advisory Commission](#) [103] and in accordance with the state-of-art in computer science literature on voting schemes, such that the actors of the debate can objectively discuss the security implications of different internet voting scheme designs, in order to have a better informed debate on whether to start using internet voting in the Netherlands.*

Research goal

To reach this research goal, the following research question must be answered, which is split into the different (numbered) subquestions that follow.

Research questions

How do we formalize the safeguards for voting schemes, presented by [The Election Process Advisory Commission](#), to allow for a quantitative comparison of the risk of different e-voting schemes in the presence of an attacker?

1. How do the safeguards by [The Election Process Advisory Commission](#) compare to other voting safeguards (from other scientific disciplines)?
2. How can these safeguards be operationalized to allow for quantitative comparison of the risk of different e-voting schemes?

3. What are the currently deployed e-voting schemes used in other countries?
4. Apply the formalized safeguards to an e-voting scheme and verify the results and the framework.

Research methods

To be able to answer these subquestions, different research methods will be used. The first subquestion is a knowledge question about the safeguards presented by [The Election Process Advisory Commission](#), and will therefore be answered using a literature study. The second question is the actual design subquestion, which will be answered by designing a framework that can be used for a risk assessment on the different safeguards. The requirements for this framework can be found in section 1.4. The third subquestion is a knowledge question about the current state of e-voting schemes already in use, which can be used to evaluate the framework that is built for subquestion two. This evaluation is the base of subquestion four, which is a knowledge question about the effectiveness and utility of the designed framework for the actors in the e-voting discussion. This fourth subquestion therefore also involves expert interviews to verify both the effectiveness and usefulness of this framework.

In Table 1.1, an overview of the research questions and the corresponding methods can be found (K addresses a knowledge question and D a design question).

1.3 FRAMEWORK CHARACTERISTICS

Framework characteristics

The framework that will be designed for this research should allow for a quantitative comparison of the risks of different safeguards of e-voting schemes. To allow for such a comparison, the framework should act as a risk assessment method for the safeguards of [The Election Process Advisory Commission](#). To come to a risk assessment that can objectively compare e-voting schemes on these safeguards, a novel approach will be taken. Instead of the well known risk = likelihood * impact and ranking them in low, medium and high, this research will address the safeguards (security properties) in *effort for an attacker*. The central question of the framework will therefore be the following.

How much effort does an attacker need to put into breaking a specific safeguard?

1.3.1 Defining attacker effort

Attacker effort defined

To calculate the actual effort of an attacker, this research investigates the time and financial investments which are necessary to break the safeguards. To get to these numbers, the economy of the digital un-

Table 1.1: Research methods per research question

Research question	Research method and problem classification	Chapter
1. How do the safeguards by The Election Process Advisory Commission compare to other voting safeguards (from other scientific disciplines)?	Literature research (K)	Chapter 3
2. How can these safeguards be operationalized to allow for quantitative comparison of the risk of different e-voting schemes?	Design (D)	Chapters 5
3. What are the currently deployed e-voting schemes used in other countries?	Literature research (K)	Chapter 4
4. Apply the formalized safeguards to an e-voting scheme and verify the results and the framework.	Case study (self) and evaluation by expert interviews (K)	Chapter 6 and 7

derground is investigated. It appears that the underground economy offers hacking tools and services that can help an attacker to achieve his goals (e.g. to find a link between a voter and a vote), depending on his financial position. Furthermore, the attacker needs to invest time to put all the different tools and research into the e-voting scheme together, to violate the safeguards. These two together form the basis of the risk assessment into e-voting security properties. It is evaluated in this specific case of e-voting, but could be used across the board in risk assessments.

The specific characteristics of e-voting allow for some of the safeguards to be expressed in the *number of votes affected*. Take for example a safeguard like “*secret suffrage*”, this can be expressed in effort to break, but more specifically, it can be expressed in the effort needed to break the secret suffrage for a specific number of votes (e.g. enough for a seat in parliament). Leading to a graph that expresses the number of affected votes that can be traced back to a voter (x-axis) in effort for an attacker (y-axis). This gives an insight into the course of risk: with how much effort can elections be rigged or can the voters belonging to a specific party be identified, compared to the same case for regular physical voting.

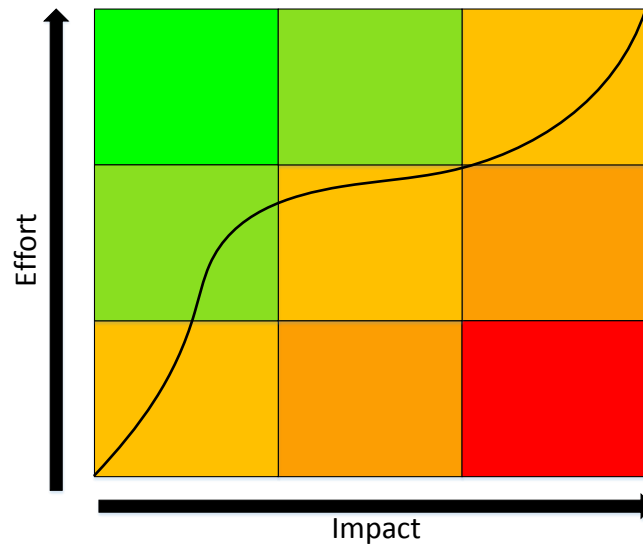


Figure 1.1: Risk plotted on likelihood and impact

1.3.2 Comparison with regular risk calculations

*Comparison with
regular risk
calculations*

The multiplication of likelihood and impact leading to risk is not the focus of this research. However, one *could* plot the effort an attacker has to make on the likelihood factor, with the impact factor being the number of affected votes. In regular risk calculations the impact factor would be a constant. The multiplication of the number of affected votes with the effort needed to get to them, is not a single point in the graph as with regular risk calculations, but would result in a function from effort \Rightarrow number of affected votes. It is not possible to pinpoint the risk to a single number and categorize it into low, medium or high, but the whole function should be evaluated as being acceptable or not. Figure 1.1 depicts this visually.

Once these functions from *number of affected votes* \Rightarrow *attacker effort* are known, the comparison of e-voting schemes based on these functions can be made.

Based on the weakest-adversary security metric proposed by Pamula et al. [79], e-voting schemes can be compared by a certain set of comparison characteristics. In their research, Pamula et al. leave open the question how to choose and score these characteristics, which for this research will be answered with the time and financial investments of the attacker. This allows for an objective comparison method.

1.4 FRAMEWORK REQUIREMENTS

Framework requirements

The risk assessment results should aid the debate about e-voting, they should therefore give a clear and concise overview of the risks associated with the different safeguards of [The Election Process Advisory Commission](#). The results should also allow for a comparison of different e-voting schemes. The risk assessment should furthermore give pointers on how to mitigate the found risks, for example by presenting the attack path for each of the attacks. Furthermore, these mitigation strategies and risk assessment results are meant for policy and decision makers and should therefore allow for some level of abstraction away from the technical details.

The requirements explained above are summarized as follows.

1. The framework should allow for a quantitative comparison of the risks of different safeguards of an e-voting scheme;
2. The comparison should be based on a risk assessment for the safeguards proposed by [The Election Process Advisory Commission](#);
3. The risk should be addressed according to the effort an attacker has to put into breaking this safeguard;
4. The effort of the attacker should be split into time and monetary investments for the attacker, based on actual dollars and days;
5. The gains for the attacker should be addressed in number of affected votes;
6. The risk assessment should give a concise and clear overview of the different risks associated with each safeguard;
7. The risk assessment should give some pointers for mitigation strategies based on the attacks;
8. The risk assessment results and mitigation strategies should be easily understandable by non-technical decision makers.

1.5 CONTRIBUTION OF THIS RESEARCH

Contribution

The risk assessment framework allows for policy makers to get a clear and concise overview of the actual risks different e-voting schemes have according to the safeguards presented by [The Election Process Advisory Commission](#). The e-voting debate thus benefits from having an objective comparison on the most important safeguards (requirements) for fair election in the Netherlands. Moreover, the e-voting literature in general is enriched by having a framework for finding and quantifying risks within e-voting schemes.

The use of *effort needed to break a system* is not new in the context of risk calculations, for example *attack trees*⁵ [90] use it to calculate the attacker path to an attacker goal with the least costs/effort. The difference with the use of attack trees, which could be used in the e-voting case as well, is that the ultimate goal for an attacker in the e-voting case is a continuous set of affected votes, instead of a fixed impact. The use of this variable impact factor, and the effort factor calculated according to the state-of-art on the underground economy have both not been seen in literature before.

The contribution of this research lies in the new approach to risk assessment, but also specifically to the case study of security properties in e-voting schemes. The framework allows for comparison of schemes, but it also helps answering questions about the scalability of attacks on these schemes and mitigation strategies. For example it is suggested by VKA [110] that every attack on an e-voting scheme (e.g. breaking secret suffrage) can easily be scaled up to an election wide attack, but that has not been investigated. This research will allow for answering such knowledge questions about e-voting too.

1.6 STRUCTURE OF THIS THESIS

Structure

The following chapters will address the different research questions and background needed to answer the general research question and achieving the research goal. To give some context on the topics that will be discussed for these questions, Chapter 2 will explain the state-of-art literature on e-voting security properties, the economy of the digital underground, information on the current discussion on e-voting in the Netherlands and a short introduction to attack trees. The first subquestion (safeguards) is discussed in chapter 3. Chapter 5 discusses the outcome of the second subquestion (framework). The different voting schemes that are currently deployed are presented in chapter 4 (third subquestion), of which one is assessed using the new framework in chapter 6 (Estonia). The framework is verified by domain experts in chapter 7 (fourth subquestion). The conclusions are can be found in chapter 8 and the future work in chapter 9.

⁵ Attack trees show the different attack paths that lead to an ultimate attacker goal in a tree like structure, more on this in section 2.4.

To give some context on the topic of e-voting, and the addressed method for the risk assessment of e-voting schemes according to the effort an attacker needs to put into breaking voting safeguards, this chapter will explain the state-of-art on different topics. section 2.1 briefly covers the current e-voting discussion in the Netherlands. Section 2.2 covers the different security properties that follow from e-voting literature in computer science. The economy of the digital underground that will be used for the risk assessment is analyzed and explained in section 2.3. Finally, a short introduction into attack trees will be given in section 2.4. For reference, appendix C provides an insight into the cryptographic building blocks used within e-voting.

2.1 BACKGROUND ON THE DUTCH VOTING SYSTEM

In the Netherlands, one can vote by either filling in a ballot at the polling station or by postal voting when living abroad. Since 1928, the Netherlands also offers the ability to vote by proxy (*“volmacht”*), allowing eligible citizens to proxy their vote to another eligible citizen. This citizen can use two of these proxy votes at maximum. There has been a lot criticism on the Dutch proxy voting, including comments by the OSCE Office for Democratic Institutions and Human Rights [77], the watch dog for democracy in Europe, because it allows for so called *family voting*¹.

Dutch experiments

In the Netherlands, there have been two experiments considering e-voting, besides the physical voting computers that served the elections many years. Both of these experiments, as well as the voting computers have been abolished from the allowed voting methods due to protests from pressure group *“We don’t trust voting computers”* (*“Wij vertrouwen stemcomputers niet”*) and controversy surrounding the experiments in politics after several publications [47, 42, 51].

The first e-voting experiment took place in 2004, called *“Kiezen op Afstand”* (*“Distance voting”*), and was intended for eligible Dutch citizens living abroad to vote for the European elections, developed by Logica CMG [53].

The second experiment took place between 2004 and 2008 with a system called Rijnland Internet Election System (RIES), specially developed for the Dutch District Water Control Board Elections (*“Hoogheemraadschap”*). It was piloted in a non binding election for these water

¹ Family voting is a form of vote coercion

control boards in 2004, after which it was used in the 2006 general elections. In 2008, all *Hoogheemraadschappen* planned to use a slightly modified version of the RIES again, but due to controversy posed by the pressure group “Wij vertrouwen stemcomputers niet” [47] and investigation by both the University of Eindhoven [51] and security firm Fox-IT [42], the plans had to be canceled. Ever since, there have been no more experiments with e-voting.

Dutch committees

During the controversy surrounding the physical voting computers, the Ministry of Interior ordered two committees to investigate both the events in the past surrounding voting [50] and the opportunities for new voting methods [103]. The latter committee investigated what was necessary to allow for the extension of currently employed voting methods. They concluded that elections in the Netherlands should comply with the following safeguards, although they immediately admit that compliance with all is impossible [103, p.21].

- Transparency: The election process should be organised in such a way that the structure and organisation is clear, so that everyone in principle can understand it. There must be no secrets in the election process: questions must be able to be answered, and the answers must be verifiable;
- Verifiability: The election process should be objectively verifiable. The verification tools may differ, depending on the method of voting that is decided upon;
- Fairness: The election process should operate in a proper manner, and the results must not be capable of being influenced other than by the casting of lawful votes;
- Eligibility to vote: Only persons eligible to vote must be allowed to take part in the election;
- Free suffrage: Every elector must be able to choose how to vote in complete freedom, free from influence;
- Secret suffrage: It must be impossible to connect the identity of a person casting a vote to the vote cast. The process should be organised in such a way that it is impossible to make a voter indicate how he or she voted;
- Equal suffrage: Each voter, given the Dutch election system, must be allowed to cast only one vote in each election, which must be counted precisely once;
- Accessibility: Voters should be enabled as far as possible to participate directly in the election process. If this is impossible, there must be a way of taking part indirectly, i.e. by proxy.

In their report, The Election Process Advisory Commission [103] performed an analysis on both postal voting and internet voting according to these safeguards. Table 2.1 summarizes the results of their analysis.

Table 2.1: Postal vs internet voting safeguard compliance according to The Election Process Advisory Commission [103]

Safeguard	Postal	Internet
Transparency	Does comply	Does not comply
Verifiability	Does not comply	Does not comply
Fairness	Does not comply	Does not comply
Eligibility	Does almost comply	Does not comply
Free suffrage	Does not comply	Does not comply
Secret suffrage	Does not comply	Does not comply
Equal suffrage	Does comply	Does comply
Accessibility	Does comply	Does almost comply

After reviewing the Table, postal voting seems to be the preferable option over e-voting, however The Election Process Advisory Commission [103, p73] argues otherwise. They consider the inaccuracy of the postal system as a big impact on the eligibility safeguard: it could very well be the case that a certain vote is delivered way too late to the election authorities. The Election Process Advisory Commission suggest to offer e-voting besides postal voting, with an emphasis on e-voting.

In spring 2013, the Dutch Minister of Interior asked the consultancy firm Verdonck, Kloosters and Associates to research the Dutch e-voting possibilities for Dutch citizens living or working abroad. The firm wasn't asked to investigate a specific implementation of e-voting schemes, but to do a general risk assessment on e-voting, and to find different mitigation strategies to allow for e-voting. In their report [110], the different safeguards concerning e-voting in general were discussed as well. They note possible attacks for almost all safeguards. Some very general mitigation strategies were found, but the report is quite high level due to the lack of a specific implementation in mind.

2.2 E-VOTING IN COMPUTER SCIENCE LITERATURE

In a recent survey by Jonker et al. [60], the two main areas of e-voting literature have been identified: privacy-related and verification-related research themes. In their extensive survey, the authors cover different e-voting schemes according to their privacy and verifica-

tion properties and present literature that found weaknesses in the claims the inventors of the scheme made. They look at highly theoretical schemes, as well as at schemes already used in practice for non-binding elections.

In e-voting research, different properties can be identified for privacy and verification, with very diverse definitions depending on the researchers and the approaches they took. In general, the following properties and definitions can be identified. The list of researchers and articles cited is far from exhaustive, but indicates interesting further reading.

2.2.1 *Privacy-related terminology*

Privacy-related terminology

The privacy-related terminology used for this research is described according to the moment in time an attacker can have contact with the voter. The voter can fall victim by either actively cooperating with an attacker (because he is being coerced, threatened or just because he doesn't care) or without cooperating with the attacker.

- **Ballot-privacy:** *“Ballot-privacy ensures that no outside observer learns for whom a voter voted, using publicly available data and communication”* [33, 67, 60, 32, 14]. In this definition, the attacker can have no communication with the voter before, during or after the voter voted. This is the most basic form of privacy, recognized by everybody to be very important.
- **Work by Cortier and Smyth [21], Dreier et al. [32], Gennaro [43]** focuses on the new concept called ballot-independence: *“Observing another voter’s interaction with the election system does not allow a voter to cast a meaningfully related vote”*. With this definition, one can exclude related votes, which in a small sized election can interfere with ballot-privacy in the following way: casting a related vote to your victims vote multiple times, could reveal the original vote being copied because of a large peak for a particular candidate. Other work by Smyth and Bernhard [97], Dreier et al. [33] help formalizing this property.
- **Receipt-freeness:** *“Receipt-freeness ensures that a voter is not able to convince an attacker he actually casted his (final) vote in a particular way after the elections, assuming that the attacker can view all message exchanges between the voter and the voting authority and knows all public information and information known to the voter before the voting phase. The attacker may have contact with the voter before and after the voting phase, in which the voter may reveal his secret data”* [33, 28, 17, 32]. In this definition, the attacker can communicate with the voter before and after the voter voted, but not during the voting process. This definition is considered to be a

stronger assumption than ballot-privacy. Other researchers use definitions that do not leave any room for a receipt at all [60, 67]. To allow for procedural measures such as used in the Estonian voting system [72], the definition used in this research does leave room for a receipt, however one that cannot convince an attacker (and thus should not convince the voter either).

- **Coercion-resistance:** *“A coercion-resistant scheme offers not only receipt-freeness, but also defense against randomization, simulation and vote spoiling attacks—all potentially in the face of corruption of a minority of tallying authorities. An attacker may have contact with the voter before, during and after the voting phase and may reveal his secret data before the voting phase”* [61, 33, 57]. This definition allows an attacker to communicate with the voter before, during and after the casting of the ballot. This definition is considered to be a stronger assumption than receipt-freeness [27, 66]. In the definition of coercion-resistance used throughout this research, it is even explicitly stronger, although not all researcher agree the definition should include receipt-freeness [8]. The concepts that are additionally taken into account include the ability of forcing a voter to either spoil his vote, or to vote randomly and to force a voter to give his credentials away so that an attacker can vote on his behalf.
- **Coercion-evident:** *“This means that unforgeable evidence about the degree of coercion that took place is included in the election output. Authorities, observers and voters can examine this evidence and use it to determine whether the election result carries a mandate for the winning candidate. Thus, election authorities can decide to consider the election as valid or not, leading to disincentivisation of coercion.”* Grewal et al. [48]. This different notion of how to deal with coercion has recently (2013) been proposed to fight the problem of achieving both coercion-resistance and verifiability (in the next section). The authors suggest to annul all votes from a particular voter, if he voted for multiple candidates in multiple votes, arguing that re-votes are only casted to further *strengthen* a vote, and not to change a vote. Note that in general elections with paper ballots, it is also not possible to re-vote for another candidate (since re-voting in general is not possible) and thus makes this property a quite realistic one.
- **Unconditional privacy:** *“Even a computationally unbounded party does not gain any information about individual votes other than what can be inferred from the final tally”* [75, 11, 17]. This definition expands the ballot-privacy definition in the sense that a vote stays private forever. This points out the fundamental difference between statistically and computationally hiding of a vote.

- Recent work by Jonker et al. [59, 58] presented a measure to quantify instead of (binary) qualify the ballot-privacy a voter has. They do this by determining the *choice group* that is left for the voter. The choice group consists of all candidates the voter can vote for, without the attacker being able to distinguish that vote from his preferred vote. One could name this the *degrees of freedom* left for the voter given his preferred vote. This approach is radically different from for example the unconditional privacy approach: letting go off just *any* privacy was unthinkable for researchers like Moran and Naor [75], Chaum [11], Chevallier-Mames et al. [17] who primarily focused on unconditional privacy.

E-voting schemes offer subsets of these privacy properties to the voters.

2.2.2 Verification-related terminology

Verification-related terminology

There is a clear tension between providing receipt-freeness or coercion-resistance to a voter, whilst at the same time offering him an opportunity to verify his vote. The verification related terminology will be explained next.

- From the early days of research into e-voting [14], verifiability of votes has been a topic in literature. Starting with *individual verifiability*, which allows a voter to verify if his vote reached the voting authorities [91], or that he can verify that his vote affected the results correctly [67] or similar explanations [33, 32, 83, 28, 89]: “the voter can verify that her vote affects the result correctly”, sometimes including a *bulletin board*² [64, 96]. [8] also includes the option to let a voter correct his vote if necessary.
- Later on, being able to verify an individual vote did not satisfy the needs for voters: they also wanted to verify that the sum of all individual votes corresponds to the election results, called *universal verifiability* [33, 32, 67, 28, 83, 91]: “anyone can verify that the announced result is the correct accumulation of the individual votes”. Again, the bulletin board is used in some definitions [64, 96, 89].
- Another interesting view was proposed by Pieters [82] in 2006. He splits each definition of verifiability into two different ones in which he allows to differentiate between verifiability that reconstructs the actual vote (*constructive verifiability*) and verifiability that proofs without reconstructing the vote (*classical verifiability*). Both of these notions can be applied to individual

² A bulletin board is a public place, displaying all the (encrypted) votes that have been casted. [57]

and universal verifiability. Pieters also relates the differences between classical and constructive verifiability to the current election process that is used within the Netherlands. Pieters argues that, in the current situation, it is not possible for a voter in the Netherlands to reconstruct his own individual vote to verify if it was correctly added up to the final tally, but that only the process in which votes are tallied provides proofs that a voter's vote is added up correctly. This situation is described by Pieters as *classical individual verifiability*. Pieters goes on to conclude that the current situation in the Netherlands allows for *constructive universal verifiability* because of the paper trail that ballots cause, it is possible for anybody to recount all the votes to come up with the final result, reconstructing every single vote (without being able to link them to individual voters).

A rather related notion was expressed by Essex et al. [36], describing their Eperio e-voting scheme. They argue that the verifiability should be offered to an external auditor instead of being offered to voters in general.

- The combination of individual and universal verifiability is often called end-to-end verifiability [13, 5, 62, 12, 60], which is split into two components: *cast-as-intended* (correctly encrypted) and *counted-as-cast* (correctly counted) of which the latter is again split into two components: *recorded-as-cast* (correctly received by the voting authorities) and *counted-as-recorded* (correctly counted by the voting authorities). On an individual basis, the combination of the three is similar to individual verifiability, but on a universal basis, counted-as-recorded corresponds exactly with universal verifiability. If a voting system accepts re-voting (overwriting a previous vote), the cast-as-intended and stored-as-cast verifiability will be seen as part of one particular vote, whereas the counted-as-recorded will cover the "overwriting of the votes" part. This is intuitively the case, considering the names these verification methods have, but must be made explicit to classify voting schemes accordingly.
- In 2010 Langer et al. [67] introduced the *cast-by-me* and *contains-correct-vote* as an extension on *cast-as-intended*. The cast-by-me verifiability corresponds to the ability of a voter to correctly identify his own vote among a set of other votes while the contains-correct-vote actually verifies that the inside of the encrypted vote correctly reflects the voter's intention. No proof is given that cast-by-me and contains-correct-vote together imply cast-as-intended or vice versa.
- Eligibility verifiability, introduced by Kremer et al. [64] in 2010 is defined as "*anyone can check that each vote in the election outcome was cast by a registered voter and there is at most one vote per voter*".

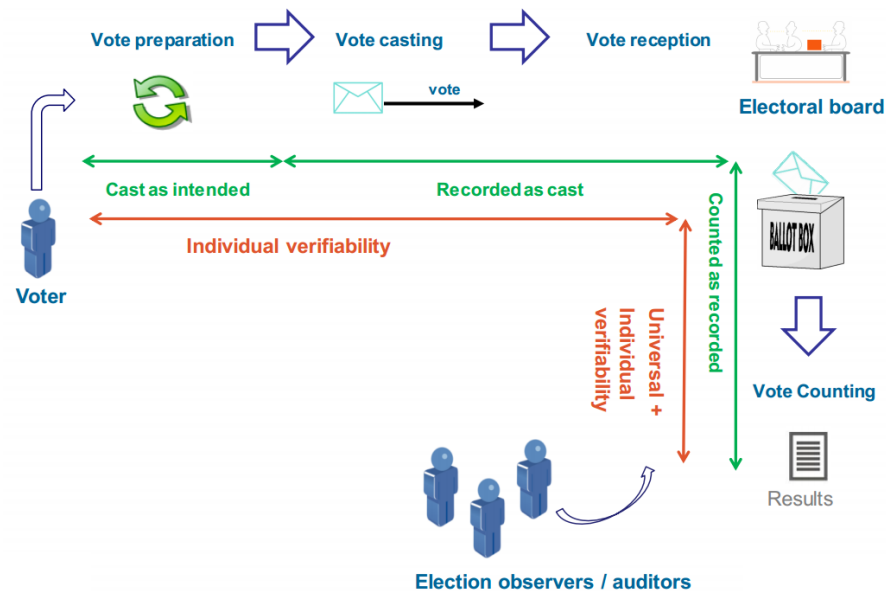


Figure 2.1: Verifiability types compared

They also define “election verifiability” as the combination of individual, universal and eligibility verifiability. For this research, the verifiability of eligibility is considered out of scope, but an interesting topic for further research.

E-voting schemes offer subsets of these verifiability methods to the voters.

Scytl, a company specialized in building e-voting systems visually depicted the most commonly used types of verifiability in Figure 2.1³

2.2.3 Conclusion

The privacy and verification properties that have been described in the foregoing sections vary in importance among different authors. Whereas some authors suggest a major focus on the privacy aspects, some other authors focus more on the verifiability. As discussed, it is very hard, if not impossible, to offer e-voting schemes that satisfy all of the properties discussed. Especially the fundamental problem with offering both the highest level of privacy and the highest level of verification is considered to be unsolvable [17]. When considering voting from an uncontrolled environment, Pieters and Becker [81] argue to emphasize on the verifiability of the vote and the transparency of the voting scheme since the identity of the actual voter in the uncontrolled environment is extremely hard to check (making the privacy-related properties somewhat less important).

³ Source (sic): http://www.scytl.com/wp-content/uploads/2013/05/Cryptographic_protocols_for_providing_transparency_and_auditability_in_remote_electronic_voting_schemes.pdf

Under reasonable conditions however, it is generally accepted that an e-voting scheme should provide for *ballot-privacy* and some level of *receipt-freeness* (although that may be done procedural by, for example, offering the option to re-vote). Furthermore, to allow for transparency, either a constructive or even a classical type of *individual* and *universal verifiability* should be offered to the voter.


2.3 UNDERGROUND ECONOMY

To better understand the underground economy, a small investigation by the author of this thesis into the underground market was conducted ⁴ to compare the prices actually found, to what is known from literature. During the research into some of these underground markets, it was found that the prices from literature were quite accurate and correspond to currently asked prices. Furthermore, literature seems to agree on these prices among different authors too. The following subsections will present a few of these researches from literature to come to a general list of underground prices which will be concluded in section 2.3.7.

Figure 2.2 shows a partial screenshot of what this underground could look like.

Botnet Rental for Installs

- Load Service: **Buy \$110 / 1K installs (USA)**



CONTACTS:
Support #1: ICQ 59612
Support #2: ICQ 59076
Support #3: ICQ 975

ABOUT US:
We are pleased to introduce you a brand new service. We sell unique loads from different countries.
If you are determined to be a regular customer - we are ready to give you a discount. The more you buy- the less you pay!

RULES:

- We sell unique loads from different countries except for Ru.
- Don't forget that we accept only prepayment via WebMoney and you are to take at least 1k.
- You must be sure of individual approach to each customer and a 24/7/365 friendly support.

OUR PRICE:

United States	\$110
All world	\$16
Mix with no Asia	\$30
Asia	\$8
Canada	\$100
Gb	\$150

Please contact with supports about prices for other countries

Botnets for sale.

Figure 2.2: Partial screenshot from the underground economy [71, p11]

⁴ Investigated undergrounds include www.antichat.ru, www.hackforums.net, www.madetrade.org, www.damagelab.org, www.blackhatworld.com, www.exploit.in, www.darkode.com and www.verified.cm.

2.3.1 Research into the Russian Underground

In 2012, Trend Micro ⁵ published a report on the Russian Underground [46]. In the report, they explain concepts and techniques that can be found on the Russian underground and they include prices that they found in their online investigation. The following paragraphs will note the most important findings.

Crypters

Trend Micro explains that a crypter is used to conceal malware from virus scanners fingerprinting for specific instruction sets. *“The more effective the technique used, the more expensive a file is”* [46, p1]. In Table 2.2, prices can be found for different crypters. Statistical crypters use different instruction sets for decrypting and running the malware for every client to avoid detection for those instruction sets. Polymorphic crypters use more advanced methods to encrypt parts of decryption algorithm and parts of the malware, putting more thought into detection avoidance. A joiner is a program that stitches different files together in a container before being crypted. Well known exploit packets can be bought crypted on forehand (stub crypter).

Table 2.2: Costs of a crypter according to Trend Micro [46]

Offering	Estimated price (US \$)
Basic statistical crypter	10 to 30
Joiner	10 to 30
Stub crypter with various add-ons	30 to 80
Polymorphic crypter	100+

Hosting

Having a server hosted somewhere on the internet is a powerful tool, as it can help distribute software, manage botnets, etc. Trend Micro splits the costs in dedicated servers and more powerful servers. Furthermore, they identified bulletproof-hosting services, which *“allow cybercriminals to host any kind of material on a site or page without worrying about it being taken down due to abuse complaints”* [46, p3]. To mitigate attacks on the server, one can also rent bulletproof-hosting with DDoS protection and a high capacity internet connection. Table 2.3 lists the estimated costs of hosting according to Trend Micro.

Pay-Per-Install (PPI)

Cybercriminals can buy installs for their malware in the underground as well. The malware gets distributed among existing botnets, but can

⁵ Trend Micro is a Japanese security software company.

Table 2.3: Costs of hosting according to Trend Micro [46]

Offering	Estimated price (US \$ per month)
Dedicated server	0.50 to 1
Powerful server	10 to 20
Bulletproof-hosting service (i.e., VPS/virtual dedicated server)	15 to 250
Bulletproof-hosting service with DDoS protection and a 1Gb Internet connection	2,000

also be advertised on websites (usually pornography) by exploiting browser weaknesses of the visitors of that website. A cybercriminal usually pays per 1,000 installs and can choose his target audience. This is no exact science, since infections could also occur on other website visitors, or the malware could be detected by several antivirus tools before installation, however the numbers usually are quite accurately matched. Table 2.4 shows the prices for different target audiences.

Table 2.4: Costs of Pay-Per-Install Services according to Trend Micro [46]

Offering	Estimated price per 1000 installs (US \$)
Global mix	12 to 15
European mix	80
Russia	100
United States	100 to 150
Spain, Germany, or France	170 to 250
New Zealand	200 to 250
Great Britain	220 to 300
Italy	200 to 350
Australia	300 to 550

DDoS

Prices for Distributed Denial of Service attacks (flooding servers with traffic to make them unavailable) differ quite a lot depending on the strength of the attack. One can choose to build the botnet code themselves, and distribute it via a PPI-infrastructure. For comfort, one can also buy / rent a DDoS botnet and attack a victim with that one. Usually, the botnet can be controlled via a web interface provided by the

so called botnet-herder. Prices found in the Russian underground by Trend Micro can be found in Table 2.5.

Table 2.5: Costs of DDoS according to Trend Micro [46]

Offering	Estimated price (US \$)
1-hour DDoS service	10
1-day DDoS service	30 to 70
1-week DDoS service	150
1-month DDoS service	1,200

Spam

The report by Trend Micro has a lot of prices for different types of spam, but for this research only the SMS spam is interesting. Table 2.6 shows the costs for spamming SMS messages. The originator name and number can be faked for convenience.

Table 2.6: Costs of spamming according to Trend Micro [46]

Offering	Estimated price for 1,000 texts (US \$)
SMS flooding service	15

2.3.2 Research on DDoSing Serbia

The results from Trend Micro [46] were used by Radunovic [86] to derive the costs for DDoSing the complete country of Serbia. These estimations can be used to have an insight into what components are necessary to fulfill a convincing large-scale DDoS attack. The author suggests to acquire the following:

- A backdoor trojan for 30 euros, crypted for 20 euros;
- Installing the trojan on 50,000 computers via a PPI-infrastructure on a European + Global mix for 2,000 euros;
- Remote DDoS-kit with source code to control the botnet using a web interface for 300 euros;
- Bulletproof Command&Control servers, 14 in total, to control the botnet, each for 150 euros, to total 2,100 euros;
- A master Bulletproof Command&Control server with DDoS protection for 1500 euros;

- A VPN-server to connect to the master Command&Control server anonymously for 30 euros.

This sums up to a total of about 6,000 euros for 40,000 online bots (out of 50,000). According to the author, this would be enough to bring the complete country of Serbia down.

2.3.3 Research on Crimeware-As-A-Service

In a research performed by Sood and Enbody [98], the emphasis was on the ease a non-expert could buy crimeware-as-a-service and use it in his advantage. They found that, besides the PPI-infrastructure (1 US \$ per 100 bots), other ways of installs could be found too, such as constructing a crimeware advertisement (crimevertisement), costing about US \$ 500 to US \$ 800 a month. Other schemes like 1 to 1.5 US \$ cents per click were also found.

Furthermore, they also noted some prices they found in their search through the underground economy, which can be found in the following paragraphs.

Crypters and obfuscators

Obfuscators can be used to turn any set of instructions into a hard to reverse-engineer set of instructions while preserving the inner working, of which crypters (using encryption) is a special kind. Sood and Enbody [98] identified different types of obfuscators and arranged them according to the input type of which the results can be found in Table 2.9.

Table 2.7: Cost of crypters and obfuscators according to Sood and Enbody [98] (source: www.styx-crypt.com)

Obfuscated files	Estimated price (US \$)
HTML Code	5 to 10
iFRAME code	5 to 10
JavaScript/DOM code	5 to 10
Executable (EXE)	20 to 25
Dynamic link library (DLL)	20 to 25
PDF	25 to 30
Flash (SWF)	25 to 30
PHP/JSP/ASP scripts	25 to 30

Botnet components

Botnets can be custom-built, of which the different components were identified and summarized in Table 2.8. A botnet starts with a basic

Build with C&C-server and can be extended with additional components. Usually, a botnet is ordered with the additional components at one vendor, but sometimes, different components (with APIs) are bought at different sellers. Sood and Enbody [98] does not explain the usage of all the components, but most speak for themselves.

Table 2.8: Estimated cost of botnet components according to Sood and Enbody [98] (source: www.verified.ms)

Botnet component	Estimated price (US \$)
Basic configuration Build (Builder+C&C+APIs)	2,500 to 3,000
Backconnect servers (SOCKS/FTP/RDP)	40 to 70
Credit card stealer (Luhn's verifier), screenshot stealer, DDoS module, certificate grabber, custom connector or bug reporter	50 to 100
Mini AV engine (bot detector)	150 to 250
Log parser module	250 to 350
Keylogger/form-grabber module	300 to 350
GEO IP protection module	300 to 400
VNC admin panel	300 to 500
FTP iFRAMER	800 to 1,000
Complete VIP package	5,500 to 6,000

Browser exploit packs (BEP)

To setup a crimeadvertisement yourself, the underground also provides (obfuscated) source code of so called browser exploit packs (BEPs). These obfuscated source codes can be used to run browser fingerprinting services, by which victims can be fingerprinted for vulnerabilities which can be leveraged to install the malware that was crafted. Table 2.9 shows the different options and packages that come along with BEPs.

2.3.4 Research on Cybercrime-As-A-Service

In an article from FortiGuard Labs [71]⁶, cybercrime-as-a-service was explained. Throughout the article, different prices for cybercrime

⁶ FortiGuard is a US-based security company, primarily focused on mobile.

Table 2.9: Estimated cost of browser exploit packs (BEP) according to Sood and Enbody [98] (source: www.exploit.in and www.madtrade.org)

BEP licenses/features	Estimated price (US \$)
One-day effectiveness test	50 to 100
PPI per 1000 infected hosts	50 to 150
Renting servers	50 to 200
Domain services	50 to 300
3 month license	500 to 800
Semi-annual license	1,000 to 1,200
Annual license	1,500 to 2,000

were mentioned according to their experience in the underground world. The prices can be found in Table 2.10.

Table 2.10: Estimated cost of cyber crime according to FortiGuard Labs [71]

Service	Estimated price (US \$)
Crypters, packers and binders	10 to 100
Simplified botnet code	50+
Infection / spreading services (1,000 installs)	100
Remote Access Trojans	250
Consulting services such as botnet setup	350 to 400
DDoS service for five hours per day for one week	535
Crimeware upgrade modules (e.g. SpyEye personalized)	500 to 10,000
Crimeware (Zeus/SpyEye)	700 to 3,000
Exploit kits (e.g. Black Hole, GPack, MPack, IcePack and Eleonor)	1,000 to 2,000

2.3.5 Research on Pay-Per-Install (PPI)

The costs for PPI-infrastructures noted in the aforementioned subsections were all from 2012 and 2013, but the same costs were already identified in 2011 by Caballero et al. [9]: “PPI providers profit from installation fees paid by the clients. PPI install rates vary from \$100 - \$180 for a thousand unique installs in the most demanded regions (often the US and

the UK, and more recently other European nations), down to \$7 - \$8 in the least popular ones (predominantly Asia).” Which could suggest that the prices can be considered very stable.

2.3.6 Zero days

Zero days (or o-days) are flaws found in applications or operating systems that were not yet found by any party except for the attacker. Therefore virusscanners, intrusion detection systems, firewalls, etc. will not be able to spot or mitigate the threat posed by the attack. These zero days are very important for highly sophisticated attacks, and were used in several very complex (considered state-sponsored) malware such as stuxnet [15].

A research from 2007 [74] suggests that vulnerabilities for operating systems cost about \$50,000 - \$250,000. Researchers from Forbes magazine also found similar prices in a 2012 study ⁷, averaging Mac OSX zero-day vulnerabilities to about \$20,000 - \$50,000, and Windows zero-days at \$60,000 - \$120,000. A study from 2014 [1] confirms these prices as being the most up-to-date. Table 2.11 presents these prices.

Table 2.11: Prices of zero-days in the field

Platform	Estimated price (US \$)
Mac OSX	20,000 to 50,000
Windows	60,000 to 120,000

2.3.7 Conclusions

To conclude this section about the underground economy, it can be seen that prices for services and tools offered in the underground economy are more or less shared between different researchers and firsthand research on those fora by the author of this thesis as well.

Table 2.12 summarizes the prices that will be used throughout this research, which are more or less averaged prices from different authors.

⁷ Forbes shopping for zero-days: an price list for hackers secret software exploits: <http://www.forbes.com/sites/andygreenberg/2012/03/23/shopping-for-zero-days-an-price-list-for-hackers-secret-software-exploits/>.

Table 2.12: Average prices in the underground economy

Service	Subtype	Estimated price (US \$)
Crypters	Static crypter	10 to 30
	Joiner	10 to 30
	Polymorphic crypter	100
PPI (1,000)	Global Mix	50 to 100
	USA	100 to 150
	New Zealand	200 to 250
	Australia	300 to 550
	Russia	100
	Asia Mix	7
	Europe Mix	100
	Specific European countries	170 to 350
Hosting per month	Powerful Server	10 to 20
	Bulletproof	40 to 250
	Bulletproof with DDoS protection	2,000
	Anonymous VPN only	30
DDoS	per day	50
	per week	150
	per month	1,200
SMS flooding	per 1,000	15
Botnet code	Simplified	50+
Remote Access Trojan	Standard	250
Consulting services	Botnet setup	350 to 400
Zeroday	Mac OSX	20,000 to 5,000
	Windows	60,000 to 120,000

To describe the formalized framework for assessing e-voting schemes, the prices will be used to “decorate” the steps of an attacker in traditional attack trees with actual effort to put into breaking election safeguards. This means that it possible to calculate the costs of breaking these safeguards for a number of votes. This will be addressed in chapter 5.

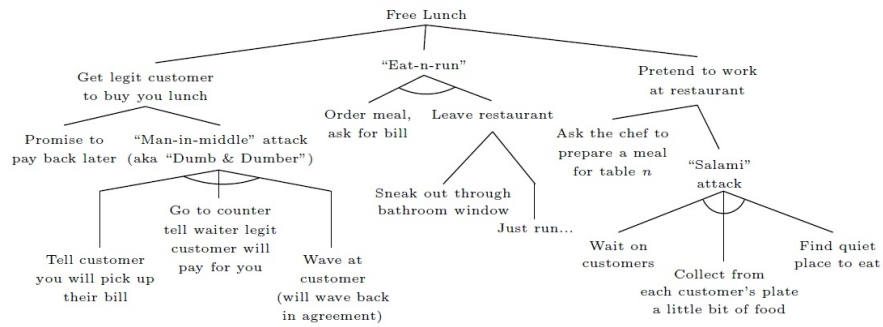


Figure 2.3: Example of an attack tree [73]

2.4 ATTACK TREES

Attack trees were first introduced by Schneier [93]. They are used to model the different attack paths that ultimately lead to a predefined attacker goal. Attack trees are described as trees with the attacker goal as the root node, each subsequent subnode represents a path to reach this ultimate goal. The leaf nodes are the actual steps that are required to reach the goal through the attack depicted by the nodes between the leaf and the root. Non-leaf nodes are by default "OR"-composed, they thereby define different independent attack paths, but can be extended with an "AND"-composition. An and-composition requires multiple paths (or leaves) to be combined in order to reach that (sub)goal and are visually depicted with an arc. Every leaf-node can be "decorated" with attributes that give additional information on those steps in the attack, they can e.g. depict the costs, the use of special equipment or the possibility of discovery of the attack. Together with the and- and or-compositions, each of the nodes higher up in the tree can be constructed according to these leaf-attributes to allow for further analysis of the attack path. Attack trees can be analyzed with questions such as the cheapest path without special equipment, or a path which won't be discovered. The set of all possible paths is called the "attack suite" [73]. Figure 2.3 shows an example of an attack tree for a free lunch from the paper of Mauw and Oostdijk [73].

Mauw and Oostdijk [73] show, among other contributions, where attack trees originated from, and how they relate to other such notions from prior literature. For a thorough understanding of attack trees, the reader is directed to this work.

Extended attack trees

According to Buldas et al. [7], regular attack trees are depicted too simplistic, when taking all attacker considerations into account. In their model, they also include some additional probabilities and costs to really determine the outcome of a certain attack, taken a reasonable

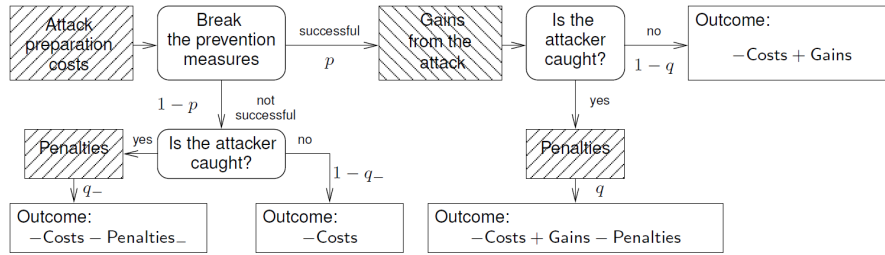


Figure 2.4: Event tree in the attacker game [7, Figure 2]

attacker in a game theoretic setting. For each of the different steps an attacker can/needs to take to achieve a certain higher level node in the attack tree, they define the following in costs: the attack has certain attack preparation costs (Costs); a probability of success in this step (p); a probability of getting caught in either situation (q); a penalty when getting caught (Penalty); and gains when the attack was successful (Gains). Put very simplistically, they depicted this in Figure 2.4, but they note that there are far more complex versions possible.

Buldas et al. use their model to show how to determine the actual benefits of substeps within the attack tree. This can for example be used to determine which of the or-clauses is most probable for an attacker to take.

Shortcomings

Attack trees are explicitly designed, not to contain repetitive steps. Attack paths can share different sub steps, but as [73] shows, they can ultimately be transformed into so called “rooted directed acyclic bundle graphs”. Furthermore, the attacker goals are defined according to a specific and determined goals, which do not allow for any quantification to be drilled down into the sub steps. In attack trees it is e.g. impossible to model an attacker goal with constants like “Break secret suffrage of X voters” where X can be used in the substeps, and in their attributes. Especially the formal modeling of attack trees such as Mauw and Oostdijk describe, does not allow for such a superset of functions on top of attack trees.

Election safeguards are requirements that must ensure that elections are fair and free and that the corresponding result is a representation of the will of the citizens [88]. This chapter will first present the Dutch safeguards, the information security approach to safeguards and a comparison between the Dutch approach to safeguards and the information security approach to safeguards. It ends with a conclusion. These topics can respectively be found in sections 3.1, 3.2, 3.3 and 3.4.

Besides these topics, different laws and regulations, both on a national and international level influence the requirements for elections, as do researchers and electoral councils of different countries. The background on these topics can be found in appendix B.

This chapter will thereby answer research question one: *“How do the safeguards by The Election Process Advisory Commission compare to other voting safeguards (from other scientific disciplines)?”*

3.1 DUTCH APPROACH TO SAFEGUARDS

To allow for extension of the currently employed voting methods, [The Election Process Advisory Commission](#) investigated the safeguards necessary for the Dutch democratic system. Their report [103] lists the eight safeguards for elections in the Netherlands, which can be found in section 2.1.

The Election Process Advisory Commission [103, p22] themselves remark that there is some tension between some of the safeguards. They notice that fairness together with equal suffrage has to be in balance with secret suffrage. This could lead to some tension because the secrecy of suffrage could prohibit the recording of the fact that someone voted at all, which could get into tension with the fairness and equal suffrage safeguards. Furthermore, free suffrage and accessibility also have a certain tension, the more accessible the elections, the more chance of having problems with free suffrage because of less trained staff at the voting booths. One of the most important remarks on the tension between safeguards, is the tension between accessibility on the one hand and free and secret suffrage on the other, especially in the case of disabled voters or voters from abroad being offered an alternative voting method.

The Electoral Council agreed on most safeguards from [The Election Process Advisory Commission](#), although they advise to add one

additional safeguard: Independence [65]. The Electoral Council advises to add this safeguard to show the importance of having an independent Council (themselves) which organizes the elections and publishes the results.

3.2 INFORMATION SECURITY APPROACH TO SAFEGUARDS

From a computer science perspective, information security practice offers different key principles to describe the safeguards of information systems. Key principles mostly include the CIA-triad (confidentiality, integrity and availability) and additionally authenticity, non-repudiation and accountability. These principles can be applied to e-voting systems too, to provide for an information security analysis on the process and the systems involved. The following subsections describe these principles in the case of e-voting, which allows for section 3.3, which relates these principles to the safeguards posed by [The Election Process Advisory Commission](#).

3.2.1 Confidentiality

Looking at confidentiality, there are several items to be aware of. Besides the obvious confidentiality of the vote, voting credentials and decryption keys of the voting authorities, there are some additional confidential data. As discussed in section 2.2.1, there are several arguments to keep the number of votes per user confidential, for example to fight vote-spoiling (coercion-resistance) or using the fact that you voted only once together with information on that vote as a receipt for vote-buying (receipt-freeness). Therefore, the number of votes casted per voter is considered confidential. As discussed in appendix B.2.4 the intermediate result should also be kept confidential during the elections.

3.2.2 Integrity

Integrity in an e-voting scheme immediately points to the correct results of an election, which demands integrity of the individual votes. To allow for a correct tally of the votes, the voting process should ensure that only *one* vote per voter is tallied, which can be considered as integrity of the *final* vote of the voter. In voting schemes in general, the list of parties and candidates needs to be correct, to allow for a fair chance to vote for your preferred candidate and party. Consider a scheme in which candidates and parties are encoded into a numerical representation, not only the list of candidates and parties should be correct, it should also contain the correct encoding (implicitly the order e.g.). To allow to vote in general, the election authorities need access to a correct list of eligible voters. Specifically for e-voting

schemes, the different software and hardware components need to function as specified (or need to *be* as specified). The integrity of the verification mechanism impacts the integrity of the voting scheme and should therefore be integer too.

3.2.3 *Availability*

Besides the availability of the voting system for voters, the digital ballot box should be available to the election authorities as well. Depending on the verification mechanisms provided to the voter, different additional system components should be available for verification. In general, the list of candidates and eligible voters should not only be integer, but also available to the voter and the election authorities respectively.

3.2.4 *Authenticity*

Authenticity in this discussion on e-voting ensures the origin of candidate and party list and the list of eligible voters. Even more specific for voting schemes, the voting process should ensure that votes that are cast, are cast by eligible voters. The origin of the voting application or voting website should be protected by authenticity as well, together with the origin of the verification mechanism.

3.2.5 *Non-repudiation*

Non-repudiation in the general information security sense ensures that both the sender and receiver can't deny having sent or received a particular message or a piece of data. For e-voting schemes, the receiver should (depending on the level of verification) not be able to deny having received a vote while on the other hand, a voter should *not* be able to *prove* (depending on the level of privacy) that he sent a particular (or his last) vote.

3.2.6 *Accountability*

Accountability in an e-voting perspective can be seen from different angles, it covers the accountability of the availability, the secrecy of cast votes or the integrity of the final results. Summarized, this leads to accountability of both a correct design and correct implementation of the e-voting scheme in general.

3.3 INFORMATION SECURITY COMPARED TO THE DUTCH SAFEGUARDS

The different information security safeguards from section 3.2 are compared to The Election Process Advisory Commission [103] safeguards in Table 3.1. A ✓ means that The Election Process Advisory Commission safeguard covers the information security safeguard, while the ● explains that it is only implicitly covered by it. The last column shows which of the information security safeguards are not met when sticking to the safeguards of The Election Process Advisory Commission. Of course, the Dutch safeguards were not meant to cover all the *nitty gritty details* of e-voting schemes, but some of these information security aspects are quite important. The following subsections will discuss some of the important findings from the information security analyses performed on the safeguards of The Election Process Advisory Commission.

3.3.1 *Login credentials*

The login credentials used for e-voting are not described in the safeguards of The Election Process Advisory Commission. The confidentiality of the voting credentials covers the type of registration and the way login credentials are distributed among the eligible voters. Furthermore, it may cover the transferability of the credentials. Take for example a smart card with more use than just e-voting, compared to a password belonging to a polling card. The easier these credentials are transferred, the easier it is to coerce someone into selling their voting credentials, thereby breaking the confidentiality.

3.3.2 *Number of cast votes per voter*

Both of the safeguards that explicitly mention casting (free and secret suffrage), explicitly explain how the voter should be protected (he “*must be able to choose how to vote in complete freedom*” and it “*must be impossible to connect the identity of a person casting a vote to the vote cast*”). Both of these protections miss the confidentiality of the number of cast votes. As explained in section 3.2, the confidentiality concerning the “*having voted*” (or “*having re-voted*”) is essential in an e-voting scheme to ensure any type of receipt-freeness or coercion-resistance.

3.3.3 *Voting systems*

In general, the safeguards by The Election Process Advisory Commission do not cover what safeguards should protect the voting systems. For example, the availability of the verification mechanism, the integrity of the hardware and software of the voting servers and the au-

thenticity of voting application are only implicitly mentioned in the safeguards. Such a system overview can be considered too detailed, but to leave them completely out-of-scope could make the implementation fall victim of overlooking to comply with such an implicit safeguard.

3.4 CONCLUSION

This chapter reviewed the safeguards posed by [The Election Process Advisory Commission](#). As can be found in appendix [B.1.3](#), the safeguards do comply with the international rules and regulations. Furthermore, the principles and procedural safeguards for e-voting made by the [Council of Europe](#) can all be found in the safeguards as well. Appendix [B.2.5](#) concludes that the safeguards found after a threat analysis on e-voting in general, map very well to the Dutch safeguards. Furthermore, this section shows that additional safeguards from literature can be found in the Dutch safeguards too.

Sections [3.2](#) and [3.3](#) show however that, taken a information security perspective, the safeguards do lack some important views. Especially interesting points missing from the Dutch safeguards are confidentiality of login credentials; the confidentiality of the number of cast votes per voter; and the availability, integrity and authenticity of components in the voting system.

What section [3.3](#) additionally shows, is a testable list of information security requirements for an e-voting scheme. In chapter [5](#) (proposed framework), these requirements will become a link between the attacker goals in attack trees and the safeguards from [The Election Process Advisory Commission](#).

Table 3.1: The Election Process Advisory Commission vs Information Security view on e-voting safeguards

	Transparency	Verifiability	Fairness	Eligibility	Free suffrage	Secret suffrage	Equal suffrage	Accessibility	Not included
Confidentiality of									
votes						✓			
login credentials				●					●
voting authorities decryption key						●			●
number of votes cast per voter					●	●			●
intermediate results			●						●
Integrity of									
results		✓	✓						
votes		✓	✓				✓		
the final vote		●					●		●
voter list			✓	✓			✓		
candidate list			✓		✓				
hardware and software			●						●
verification mechanism		✓							
Availability of									
the application								✓	
the ballot box									✓
verification mechanisms		●							●
voter list				✓			✓		
candidate list			✓		✓				
Authenticity of									
voter list			✓	✓			✓		
candidate list			✓						
voting application			●		●				●
verification mechanism		✓							
Non-repudiation of									
vote cast (inverse)						✓			
vote received		✓	✓				✓		
Accountability of									
correct design	✓	✓	●	●	✓	✓	✓	●	
correct implementation	✓	✓	✓	✓	●	✓	✓	✓	

In a research performed by VKA [110], eleven countries have been identified to have piloted with e-voting, of which four have stopped their pilot period. The remaining seven countries are Australia, Canada, Estonia, France, India, Norway and Switzerland.

Implementation motivations differ among countries, depending on the type of voters allowed to e-vote, France e.g. motivates that the turnout of expatriates needed to be increased [38], while Estonia motivates that only the turnout of citizens that sometimes vote should (and would) increase, not the turnout of voters that never vote [69]. Estonia also motivates that technological improvement of governmental projects stimulate technology [72].

The following sections will describe the different e-voting schemes implemented in Estonia (section 4.1), France (section 4.2), Norway (section 4.3) and briefly presents some work in other countries (section 4.4) followed by a set of findings on these schemes (section 4.5). The selection of countries that are described more elaborate was done on the basis of the available information on the schemes, the fact that they are currently used and the fact that they resemble properties that are favored for an implementation in the Netherlands as well. They furthermore represent the broader set of implementation choices and vendor dependencies that is seen with more countries.

This chapter will thereby answer research question three: “*What are the currently deployed e-voting schemes used in other countries?*”

4.1 ESTONIA

Estonia introduced e-voting back in 2005 and has since seen a steady growth of its use [69]. While e-voting was introduced in Estonia, no country in the world had piloted e-voting on this scale. Therefore, a lot of energy was put into the design of the scheme, e.g. to fight vote-buying. The Estonian voting system was the first to introduce online re-voting as well as offline re-voting, to make the scheme receipt-free. The Estonian voting period looks as follows: the first seven days of elections can be used to do online voting, online re-voting and offline re-voting; then three days of counting are organized; after that, on election day, everybody who didn't vote yet, has the possibility to vote. Re-voting online can be done an unlimited number of times, an offline (paper) re-vote can be done only once, but cancels all online votes [72, 69].

Outline

Estonia

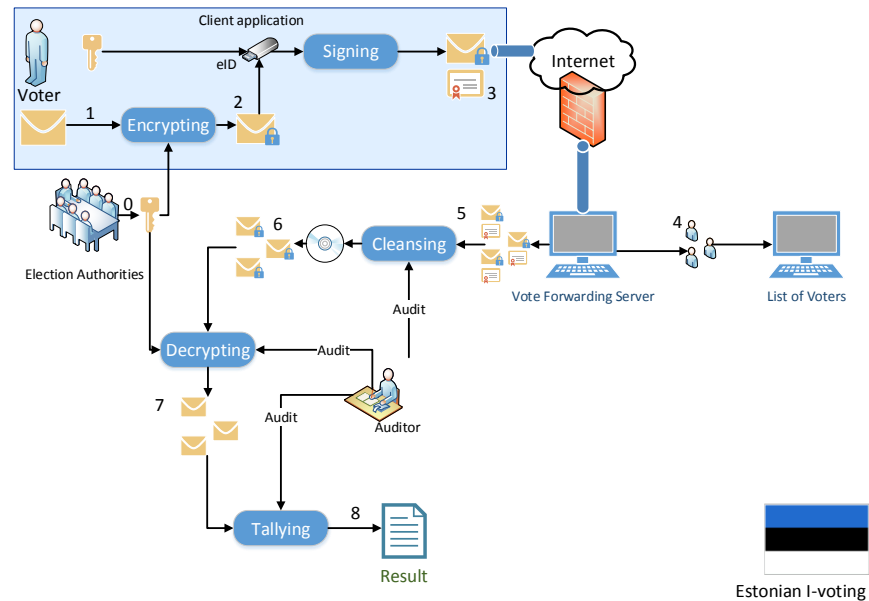


Figure 4.1: Estonian i-voting scheme (simplified)

A list of all online voters is sent to the polling stations the day after the online voting period ended, asking to mark all voters who re-voted offline. These votes are discarded before the online votes are being tallied. Furthermore, all online voters are removed from the voting lists for election day [72, 69].

4.1.1 Scheme

Estonian scheme

The Estonian system is built by the Estonian company Cybernetica and roughly works as follows. Every citizen of Estonia is obligated to own a new electronic ID (eID) which has a built-in smart card with encryption capabilities and signing keys ¹. Voters download a voting application to their computer, insert their smart card in a smart card reader and authenticate themselves to the smart card using their personal pin code (PIN₁). They then choose their party and candidate in the voting application and encrypt it with the public key of the voting authorities, after which they sign their encrypted vote after again authenticating to the smart card using another personal pin code (PIN₂). Figure 4.1 shows the procedure following the authentication phase, and after the voting applet has been downloaded by the voter. The use of the PINs is not displayed in the Figure ².

¹ On the 19th of May 2014, roughly 1.2 million cards were in active use according to the official website <http://www.id.ee/?lang=en>. This is about 98% of the Estonian inhabitants.

² Information retrieved mostly from [109] and [72]

- o. The voting application has been setup with the public key of the election authorities and the list of candidates;
1. The voter selects his preferred party and candidate in the application;
2. The voter encrypts his vote using the public key of the voting authorities;
3. The voter signs his encrypted vote using his private signing key on the smart card. The encrypted and signed vote is sent to the Vote Forwarding Server over an SSL-secured internet connection;
4. The identities of all e-voters is stored separately from the votes on the List of Voters Server, which can later be used to exclude them from voting on election day;
5. After the elections, the votes are stripped from their signature, and sent into the cleansing process to “wash off” all electronic and physical re-votes;
6. The (encrypted and) cleansed votes are burned on a CD-rom and sent to the decryption phase. The decryption phase is supplied with the private decryption key by the election authorities;
7. The decrypted votes are sent to the tallying phase;
8. The tallying phase produces the final result of the e-votes.

Different aspects of this server side platform are audited by an independent auditor: the cleansing, decrypting and tallying phases.

4.1.2 *Mobile e-voting*

In 2011, Estonia started experimenting with mobile e-voting, in which a secure SIM-card with pin codes and certificates serves as a replacement for the eID-card and the card reader. This means that the eID-card and the card reader are not necessary anymore, as their functions have moved to the specially crafted secure SIM-card. The process of voting is relatively similar. A voter downloads the voting applications to their computer, enter their telephone number and compare the code of the SMS that was sent to the one displayed in the voting application. Then the voter enters his PIN₁ to identify himself with the SIM card. The voter can then select their voting preferences on the computer, after which the mobile phone will again get an SMS with a code that needs to be compared. The voter enters his PIN₂ in the mobile phone to confirm (and sign) the vote [72], it is then sent to the voting authorities. Note that the SIM-card signed the vote using the certificates that are present on that SIM-card, instead of having the eID-card that would normally sign the vote.

*Estonian mobile
e-voting*

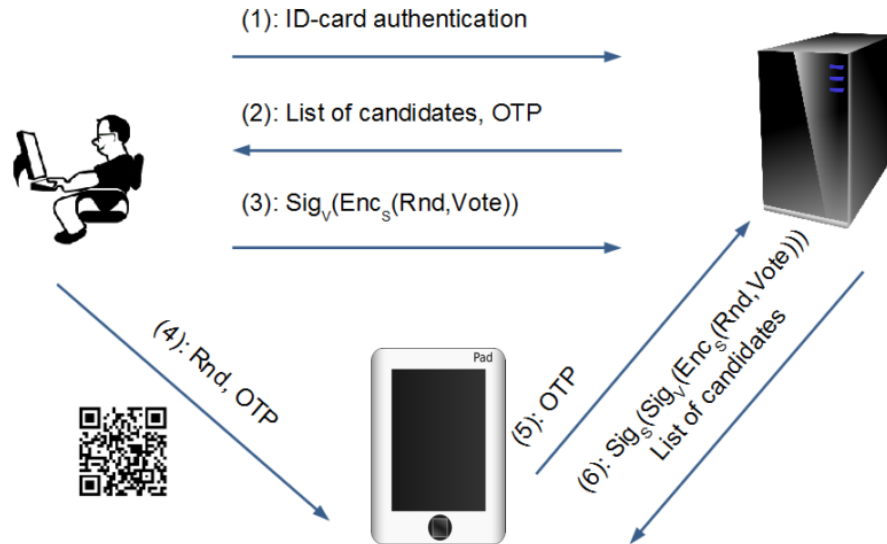


Figure 4.2: Estonian verification mechanism (simplified) Martens [72][p32]

4.1.3 Verification mechanism

Estonian verification mechanism

In 2013, Estonia started an experiment for vote verification for individual voters, using an Android-application, for their local elections. The verification works as follows, when the voter encrypts his vote with the public key of the election authority, the voter includes a *random number* in the encrypted part. The voting authorities reply with a *session code* corresponding to this specific vote (re-voting would give a new session code). The voting application then shows a QR-code containing both the random number included in the vote and the session code, which can be scanned by the Android-application. The Android-application then authenticates itself with the voting authorities using the session code, for which the voting authorities will return the encrypted vote included with the list of candidates. The Android-application can then compile an encrypted list of votes with the random number to check for the encrypted vote sent by the voting authorities. The one that matches is the one displayed by the Android-application and should match the vote casted by the voter [72]. Figures 4.2 and 4.3 visually display this process. Note that for every re-vote, a new corresponding session code is generated which can verify the corresponding vote, thereby fighting vote-buying³. Note that a voter can't check what their actual last vote was, they can only verify if a particular vote was correctly encrypted and received by the voting authorities, specifically checking whether their vote was correctly counted is not provided.

³ In a phone call with the Estonian National Election Committee it was confirmed that all votes casted by the voter have a verifiable vote returned by the voting authorities, of which only the last vote will be counted. This makes the verification worth nothing to a vote-buyer, because it doesn't say anything about it being the last vote or not.

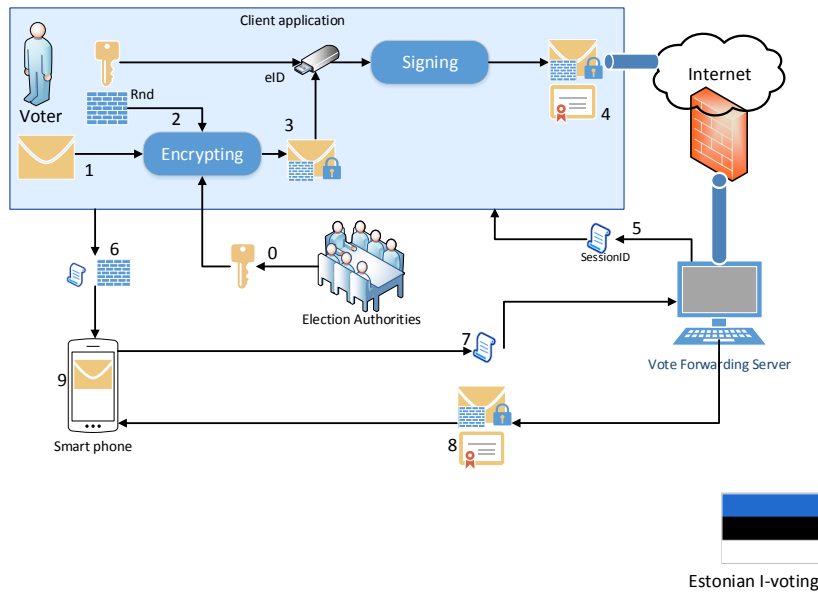


Figure 4.3: Estonian voting and verification mechanism (simplified)

4.1.4 Incidents

In the 2011 parliamentary elections, two incidents concerning e-voting occurred. The first incident occurred when, during the offline decryption and tallying phase, one vote could not be tallied, but was encrypted correctly. This could only occur in one of the following situations: a new voting application was built and used, which correctly communicated with the voting authorities but incorrectly formatted the vote; the currently employed voting application was misused; the employed voting application contained a bug. To be able to correctly address what happened, the vote needed to be decrypted, which would have broken the ballot secrecy for this particular voter. The court therefore decided not to do this for this individual case. Later examination came to one of the following two possible causes for this invalid vote: a hard to find memory bug in the application or an intentionally spoiled vote (using a man-in-the-middle HTTPS-proxy). The matter has not been resolved, and probably won't ever be resolved [49].

Incidents: Wrong vote

The second problem concerned a student who decompiled the voting application and built malware to (i) change a voter's vote when using the regular downloaded voting application and to (ii) record the voter's original intention for his vote. The malware recorded what the voter voted for, built a fake screenshot-like foreground for the application, faked mouse-movements of the voter to the preference of the student and changed the vote (without notification by the voter, because of the screenshot-like foreground). This took the student 4 to

Incidents: Proof of concept malware

5 days ⁴. When the student declared that he had built this malware as a proof of concept, the voting authorities responded saying that they had flagged his test-votes as being suspicious, however they had not actually acted upon this. The court ruled that there has not been any real attack that violated anyones ballot secrecy or integrity, therefore the election results were not discarded [49]. This is of course up to debate.

4.1.5 Statistics

Estonian statistics

The number of e-voters among eligible and actually participating voters has seen a steady growth from 0.9% and 1.9% respectively in 2005, until 15.4% and 24.3% in 2011 [38]. Also, the general turnout for elections grew in this period of time. The percentage of voter turnout for parliamentary elections grew from 61.9% in 2007 to 63.5% in 2011, and for local elections from 47.4% in 2005 until 60.6% in 2009 [38]. Besides other arguments like political situations or interesting debates to argue the higher turnout, Trechsel and Vassil [104] actually conclude, based on simulations, that the local elections in 2009 had a higher turnout of about 2.6% than similar elections without the e-voting option. Furthermore, factors that were found among e-voters compared to regular voters in the 2005 local elections did not include gender, income, education, type of settlement or age, though they were mostly influenced by the political background [69]. Other research shows similar results, but also mention the fact that e-voting is not offered in Russian, influencing the demography of e-voters [104] (which is the language of a big minority in Estonia, about 26%).

4.1.6 Conclusion

Summary of the Estonian e-voting scheme

To conclude the Estonian e-voting scheme, the privacy and verification related notions will be discussed. The elections from 2013 and onwards can be considered ballot-secret, since they provide for an environment in which an outside observer doesn't learn for whom a voter voted using only publicly available data and communication, considered that the public key of the voting authorities is transported to the voter through the voting application securely. Furthermore, according to Smyth and Bernhard [97, theorem 14, p15], this means this scheme also ensures ballot-independence. The encryption method used, does not support unconditional privacy. The definition of receipt-freeness in section 2.2.1 of this report mentions that only if a voter can prove what his *last* vote was, the voting scheme breaks receipt-freeness. Since the verification method provided by Estonia since 2013 only covers the verification of a particular vote (not the last vote), Estonia's e-voting scheme complies with receipt-freeness, but it

⁴ Source: <http://news.err.ee/v/politics/ed695579-af05-48ab-8cc0-3085e5f0c56c>

can't be considered coercion-resistance. A counterexample to show this can be the case in which a voter gives away his eID, corresponding PIN codes and the physical ballot used for physical re-voting. The coercer can now spoil the voter's vote or control all votes. Note that coercion-evidence is neither the case, since physical re-votes are not compared with e-votes, nor are e-votes compared among each other. Since the Estonian voting scheme does not offer a bulletin board like interface where all votes are published, it does not comply with the exact definition of universal verifiability as chosen in section 2.2.2, because not "anyone" can verify that the announced result is a correct accumulation of the individual votes. The individual verifiability in the 2013 and onwards elections is limited to cast-as-intended verifiability using the QR-code verification, ensuring that the content of the vote corresponds to the voter's intention. Recorded-as-cast is not the case, because there is no way a voter can be sure that the voting authorities actually stored the vote in the final database that is used for vote counting. The split of cast-as-intended into the cast-by-me and contains-correct-vote verifiability methods allow for a further analysis [67]. It is clear that both of these verifiability notions are met through the use of the self chosen random in the encryption and the sessionID returned by the voting authorities. The Estonian Electronic Voting Committee themselves claim that their verification method meets "accepted-as-cast", which is not in the list of official verification notions, but can intuitively and informally be accepted as a correct representation of what is going on.

4.2 FRANCE

France

France introduced e-voting for French living abroad in 2003 for the election of the minister president and other representatives representing the expatriates in the upper chamber [84] (using two voting rounds). In 2012, France expatriates could also vote for representatives in the lower chamber [38]. The voting application for the 2009 and 2012 elections was developed by the Spanish company Scytl. Voters who wanted to vote for the 2012 elections needed to be registered on a consular election board and were sent an ID to login, fifteen days prior to the elections by postal mail, valid for both rounds of the elections. Ten days before the elections, the ID to login was resent via SMS. A password to login to the voting application was sent via email five days prior to election day, which was different for both rounds.

4.2.1 Scheme

French scheme

The actual implementation details for this e-voting application are mostly in French, rather unclear and not well defined in literature, but in general, the elections of 2012 worked as follows, "To secure the

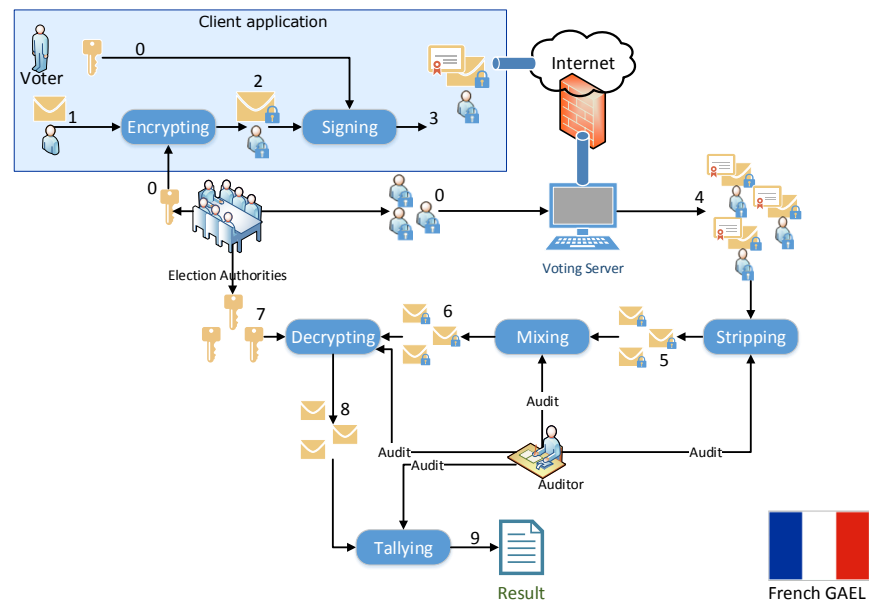


Figure 4.4: France e-voting scheme (simplified)

voter's computer, the connection to the e-voting website generates a secure electronic voting booth on the voter's machine. After he/she casts his/her vote, the voter is sent the receipt" [84, p194], being a Java-applet which is downloaded to the voter's computer [38]. From the 2009 elections and onwards, a voter could register his own password for the voting application prior to the elections, with the use of his consular identification number (NUMIC), name, birth date and passport number [38]. "The French system could be seen as the least secure in terms of voter authentication, with online registration for Internet voting. While this online registration requires information that is personal to the voter and may be difficult, but not impossible, for others to discover. However, the French system's prime motivation is encouraging the participation of expatriate voters and it is understandable that under such circumstances the balance between security and ease of access may be found in favor of accessibility" [38, p70]. The quote compares the French system to the Norwegian, Estonian and Geneva (Swiss) systems.

In both the 2009 and 2012 elections, the NUMIC is used as an identifier for a voter, which is sent encrypted, together with the encrypted and signed vote to the voting authorities. Before the decryption phase, the votes are mixed and the NUMIC is stripped off. Decryption works by threshold decryption with the shares of the private key held by the members of the election committee. After the voting period closes, the NUMIC of all voters is sent to the consulates around the world to prohibit voters from using the other voting channels for the elections [38].

Figure 4.4 shows the procedure following the authentication phase, and after the voting applet has been downloaded by the voter.

- o. The voting application has been setup with the public key of the election authorities, the voter's private signing key and the list of candidates. Furthermore, the voting server is supplied with all NUMICs of voters, encrypted with the election authorities' public key;
1. The voter selects his preferred party and candidate in the application;
2. The voter encrypts his vote and his NUMIC separately, using the public key of the voting authorities;
3. The voter signs his encrypted vote using his private signing key. The encrypted and signed vote and the encrypted NUMIC are sent to the Voting Server over an SSL-secured internet connection;
4. After the advance election days, the encrypted and signed votes and the encrypted NUMICs are sent to a stripping phase, which strips of the encrypted NUMICs and the signatures of the encrypted votes ⁵;
5. The encrypted votes are sent to a standard Scytl mixing service ⁶;
6. The mixed votes are sent to the decryption phase;
7. The decryption phase is fed with the different shares of the decryption key by the election authorities to reconstruct the decryption key;
8. The decrypted votes are sent to the tallying phase;
9. The tallying phase produces the final result of the e-votes.

Different aspects of this server side platform are audited by an independent auditor: the stripping, mixing, decrypting and tallying phases.

4.2.2 Verification mechanism

The 2009 elections were the first elections in which the French used a verification method for their voting system, "As with the Scytl system, voters get a message on their PC confirming that their vote has been stored in the ballot box at a given hour on a given day. It is possible to call this

French verification mechanism

⁵ Not included in the Figure: the Voting Server checks whether only votes casted by eligible voters are included in the collected votes by comparing the included encrypted NUMICs with the list gotten at the setup. Furthermore, the Voting Server sends all NUMICs to the physical voting locations, to prohibit them from re-voting

⁶ Additionally: it is unclear whether this complies with the Norwegian implementation of the mixnet or not.

message back anytime by again inserting the voting card number in the voting web site" [38, p60]. This verification only counts as an acceptance verification, since it does not contain any information about the vote casted. French expatriates are not offered the opportunity to re-vote.

4.2.3 Statistics

French statistics

These statistics are based on the expatriates votes. In 2003, 60.60% of all casted votes, were casted online, in 2006 this declined to 14% and in 2009 to 9%, however the setup of the election is somewhat complexer to draw figures from this. Depending on the year of the elections, different regions of the world in which the expatriates live are eligible to vote, because they renew only the representatives for their part of the world. In the 2003, expatriates living in North America could vote, in 2006, expatriates in Europe, Asia and Middle East could vote and in 2009, expatriates in Africa, North and Latin America could vote [38].

The costs for the latest (Scytl software) elections are unknown, the 2003 elections cost 61,000 euros and 2006 elections 2,000,000 euros [38].

4.2.4 Conclusion

Summary of the French e-voting scheme

To conclude the French e-voting scheme, the privacy and verification related notions will be discussed. The elections from 2009 and onwards can be considered ballot-secret, since they provide for an environment in which an outside observer doesn't learn for whom a voter voted using only publicly available data and communication, considered that the public key of the voting authorities is transported to the voter through the voting application securely. Furthermore, according to Smyth and Bernhard [97, theorem 14, p15], this means this scheme also ensures ballot-independence. The encryption method used, does not support unconditional privacy. Receipt-freeness is not supported by this voting scheme, since recording your monitor while voting can prove you voted for some candidate (and that can serve as a receipt). Since re-voting is not offered, this vote can be considered final. Therefore it also doesn't comply with coercion-resistance or coercion-evident. Since the French voting scheme does not offer a bulletin board like interface where all votes are published, it does not comply with the exact definition of universal verifiability as chosen in section 2.2.2, because not "anyone" can verify that the announced result is a correct accumulation of the individual votes. Since there is no additional feedback to the voter about the content of his vote, any form of individual verifiability is lacking from this voting scheme, including the cast-as-intended, recorded-as-cast or tallied-as-recorded, neither the additional cast-by-me and contains-correct-vote or the Es-

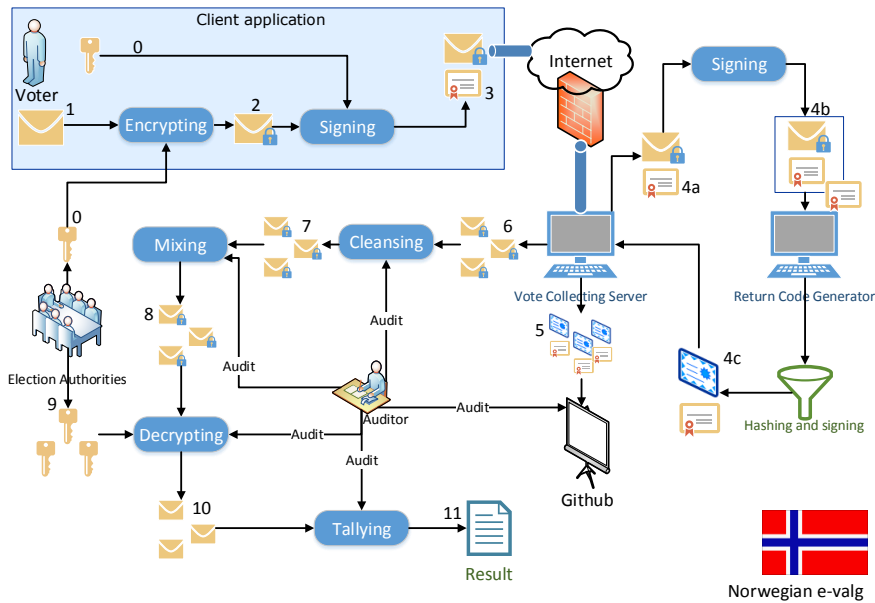


Figure 4.5: Norwegian e-voting scheme (simplified)

tonian accepted-as-cast since for all of them, at least the content of the encrypted vote should be verifiable for the voter.

4.3 NORWAY

Norway piloted e-voting for local government elections in a few (10 of 429) municipalities in 2011 [38], using a system provided by the Spanish company Scytl.

4.3.1 Scheme

The Norwegian voting scheme uses authentication by means of an electronic ID (MinID⁷) and a password to authentication to the MinID. The MinID authenticates the user to the voting website, which returns the voter a voting application and voting credentials, including the private signing key of the voter, the public key of the voting authorities and a list of the parties and candidates.

After the voting application has been downloaded to the voter's computer, the actual voting process will start. Figure 4.5 shows the procedure following this download, and will be explained accordingly⁸.

⁷ MinID is used for various governmental services already.

⁸ Information retrieved from Spycher et al. [99],

<http://www.regjeringen.no/en/dep/kmd/prosjekter/e-vote-trial/about-the-e-vote-project/github-eng.html?id=733356>

and <https://brukerveiledning.valg.no/Dokumentasjon/Dokumentasjon/Forms/AllItems.aspx> (both visited on the 11th of February 2014).

Norway

Norwegian scheme

0. The voting application has been setup with the public key of the election authorities, the voter's private signing key and the list of candidates;
1. The voter selects his preferred party and candidate in the application;
2. The voter encrypts his vote using the public key of the election authorities;
3. The voter signs his encrypted vote using his private signing key. The encrypted and signed vote is sent to the Vote Collecting Server (VCS) over an SSL-secured internet connection ⁹;
4. The signature and the eligibility are checked by the Vote Collecting Server (VCS) ¹⁰;
 - a) The VCS partially re-encrypts the signed and encrypted vote;
 - b) The VCS signs the encrypted and signed vote to proof it has seen the vote, and then sends it to the Return Code Generator (RCG);
 - c) The RCG calculates the hash of the encrypted and double signed vote and signs this before he sends it to the VCS ¹¹.
5. Every hour, all newly received hashes + signatures from the RCG are put on a Github repository. Every vote is a line in the Github file. This proves that that vote has been seen by the RCG ¹²;
6. After the elections, the votes are stripped from their signature, and sent into the cleansing process by the VCS, to "wash off" all electronic and physical re-votes ¹³;
7. The final set of votes is sent to the Mixnet, which mixes the votes and re-encrypts them ¹⁴;

⁹ Not included in the Figure: the voter computes a zero-knowledge proof, to proof the content of the encrypted vote and sends this along with the encrypted and signed vote.

¹⁰ Not included in the Figure: The zero-knowledge proof is also checked.

¹¹ Not included in the Figure: The RCG also generates a return code, which is sent to the voter via SMS, who can then check whether the return code conforms with the return codes he had earlier received via postal mail, prior to the elections.

¹² Not included in the Figure: The hash and the signature of the hash from the RCG and the signature of the vote by the VCS are also sent to the voting application to show to the voter. The entire list of these hashes and signatures is signed by the VCS and put on the Github as a separate file.

¹³ Not included in the Figure: The VCS proofs that the votes that go into the Cleansing phase correspond to the Github repository, using a zero-knowledge proof. Furthermore, the cleansing phase reports and logs every removed vote.

¹⁴ Not included in the Figure: The mixnet provides a zero-knowledge proof of correct re-encryption and mixing. Additionally: The mixnet is designed by Scytl [85] as a air-

8. The mixed votes are sent to the decryption phase;
9. The political parties put together the different shares of their key ¹⁵;
10. The decrypted votes are sent to the tallying phase;
11. The tallying phase produces the final result.

Different aspects of this server side platform are audited by an independent auditor: the cleansing, mixing, decrypting and tallying phases. All items that are put on the Github repository are also checked against these results, and are checked for consistency between the VCS and the RCG.

4.3.2 Verification mechanism

The RCG calculates a personal return code, which, on forehand was provided to the voter by postal mail. The return codes were intended to include both a return code for the candidate and the party, but later it was decided to only include the party (reasons for this are unclear) [99].

*Norwegian
verification
mechanism*

Besides a very extensive supervision of the systems included in the voting scheme, an external auditor checks whether the different phases of the e-voting schemes are performed correctly. The auditor checks (among other logs) the cleansing phase, the mixnet, the correct decryption and the tallying. The auditor furthermore checks whether the Github contains the same votes as the VCS and the RCG have.

Bulletin board

Interestingly, the Norwegian voting scheme before 2013 did not have a public bulletin board (Github), “*aiming at circumventing vote buying by individuals from the public*” [99, p8]. The elections of 2013 also introduced another feature into the e-voting scheme, namely, the number of already casted e-votes as part of the verification SMS that is sent by the RCG [110]. The exact benefit of this included number is rather unclear as this still does not tell the voter whether someone revoted on his behalf.

The protocol described in Figure 4.5 has been proven, in a simplified version, to be secure under certain conditions [44], but as Koenig et al. [63] show, it is not realistic to assume e.g. a secure SMS channel between the RCG and the mobile phone, or to ignore recent malware on smart phones to attack banking applications.

gapped LAN network, some improvements to this mix-net were found by Demirel et al. [29].

¹⁵ Not included in the Figure: The decryption phase provides a zero-knowledge proof of correct decryption. Additionally: A threshold of six out of ten political parties should cooperate to reconstruct the decryption key.

As with the Estonian system, also the Norwegian system accepts physical re-voting to overwrite an electronic vote, but in contrast with Estonian system, re-votes can also be casted on election day (after the advance voting period). During the advance voting period, online re-voting is allowed unlimited number of times [99]. The advance voting period is relatively long, compared to other countries, having 30 days of online advance voting and somewhat longer for physical advance voting. For both voting channels in the advance voting period, there is no preregistration needed.

For transparency reasons, not only the documentation of the Norwegian voting system is available, but the complete source code can be reviewed by everyone [38]¹⁶.

4.3.3 Statistics

Norwegian statistics

In the 2011 municipal voting pilot, of 168,000 eligible voters, 26.40% voted online of 104,374 votes casted. In 2013, in the municipal voting pilot, of 196,172 eligible voters, 36% voted online (70,622 voters). In total 72,969 votes were casted online (thus a total of 2,347 re-votes were casted).

4.3.4 Conclusions

Summary of the Norwegian e-voting scheme

To conclude the Norwegian e-voting scheme, the privacy and verification related notions will be discussed. The elections from 2011 and onwards can be considered ballot-secret, since they provide for an environment in which an outside observer doesn't learn for whom a voter voted using only publicly available data and communication, considered that the public key of the voting authorities is transported to the voter through the voting application securely. Furthermore, according to Smyth and Bernhard [97, theorem 14, p15], this means this scheme also ensures ballot-independence. The encryption method used, does not support unconditional privacy. The definition of receipt-freeness in section 2.2.1 of this report mentions that only if a voter can prove what his *last* vote was, the voting scheme breaks receipt-freeness. Since the verification method provided by Norway only covers the verification of a particular vote (the latest vote casted, with a possibility to re-vote), Norway's e-voting scheme complies with receipt-freeness, but it can't be considered coercion-resistance. A counterexample to show it doesn't comply with coercion-resistance is the case in which a voter gives away his MinID, corresponding password and SIM card for both authentication and verification SMS messages. The coercer can now control all votes casted by this voter. Note that coercion-evidence is neither the case, since physical re-votes are

¹⁶ Source code and design documents can be found here: <http://www.regjeringen.no/en/dep/kmd/prosjekter/e-vote-trial/source-code.html?id=645239>

not compared with e-votes, nor are e-votes compared among each other. Since the Norwegian voting scheme does not offer a bulletin board like interface where all votes (either encrypted or decrypted) are published, it does not comply with the exact definition of universal verifiability as chosen in section 2.2.2, because not “*anyone*” can verify that the announced result is a correct accumulation of the individual votes. The individual verifiability in the 2013 and onwards elections is, according to the Norwegians, cast-as-intended, recorded-as-cast and tallied-as-recorded. The cast-as-intended and recorded-as-cast can be considered correct for a particular vote by having specific return codes via SMS for the cast-as-intended property and a public bulletin board with hashes of all the votes for the recorded-as-cast. This is furthermore secured by having the MinID as a part of the authentication method: it is hard to cast a vote without the voter’s awareness, especially since the voter receives an SMS with the return code for the newly casted vote. The tallied-as-recorded verification property, claimed by the Norwegian voting authorities is not met though, because a voter can’t verify if no other votes are casted that supposedly were from him. The fact that he can’t confirm this, makes for the lack of tallied-as-recorded on an individual level. The Norwegian voting scheme offers a cast-by-me functionality, because everybody can find his particular vote among all other votes using the hash and the signature from the RCG, furthermore, the SMS from the RCG allows for trust in the correct-contains-vote property.

4.4 OTHER COUNTRIES PILOTING E-VOTING

Other countries

Some of the internationally used e-voting schemes are not considered as a plausible alternative for e-voting in the Netherlands, mostly because of the lack of verification methods, pilots stopped too long ago or for the lack of a thorough description of the voting scheme in literature.

4.4.1 *Australia*

Australia

Australia has an e-voting system in which all votes are collected electronically, decrypted using threshold cryptography and directly printed to be put into the regular voting process. The verification consists of a very simple mechanism, in which a website outputs a verification code that could be confirmed with the election authorities. This mechanism doesn’t offer any of the official verification properties [4].

4.4.2 *Canada*

Canada

Canada doesn’t use any verification method for their voting scheme [56] and publicly available literature on the Canadian e-voting scheme is

rather scarce, therefore it is considered not comparable to any plausible alternative for the Netherlands.

4.4.3 *India*

India

In 2010, India piloted e-voting in six municipalities for local body elections [38], for which 387 voters were registered. Before a voter could actually e-vote, he needed not only to be registered, but also had his identity checked with the local authorities at his home. Only 182 voters were checked and thereby approved to e-vote, of which 124 did actually cast an e-vote. The system is built by the Spanish company Scytl. After registering, the voter is offered his credentials via two different media: the username was sent via e-mail, and the password via SMS. From literature, it is unclear what the verification method is [110].

4.4.4 *Mexico*

Mexico

Mexico was the first county in Latin America to pilot e-voting in their 2012 elections for a new governor of Mexico-City. The system provided by the Spanish company Scytl recorded 2639 votes of a total of 7911 votes casted by Mexicans living abroad, which is about 33%. To cast a vote, one had to register and use the e-mailed link to login to a password page with their user-id, e-mail address and *personal information*, after which they received a 16-digit password on their screen (which they either had to print or remember). After voting, the voter receives a return code, which later was published online [110]. Besides the lack of literature on the Mexican voting experience, the offered verification method is considered to be less than cast-as-intended, and therefore this scheme is not considered as a plausible alternative for the Netherlands.

4.4.5 *United Kingdom*

United Kingdom

The United Kingdom also piloted e-voting, but the pilots stopped in 2007 after evaluation reports reported a lack of vision and strategy [110]. Since these pilots are more than six years ago, the voting schemes used in the United Kingdom are not considered as a plausible alternative for the Netherlands.

4.4.6 *Switzerland*

Switzerland

Switzerland currently has three pilot programs for e-voting, all for different regions (Cantons) of Switzerland, namely in Zürich, Geneva and Neuchâtel. The e-voting schemes are used for referendums and

elections of special bodies within the Cantons. E-voting for referendums is widely used in Switzerland, because “*In Switzerland, there is a long tradition of postal ballots*” [83, p81]. For example 95% of the voters in the last ten years have chosen “*postal voting rather than going to the ballot box*” [16, p56]. The idea of advanced voting periods is widely accepted. According to Esteve et al. [38], the Geneva system is the most advanced and well described in literature of the different Cantons, therefore, the Geneva system will be used for comparison. The Geneva system uses scratchcard like surfaces on the regular polling card send by postal mail to the voter, which can be scratched away for e-voting usage, displaying a 6-digit number. If that area is not scratched away, the polling card can be used for either voting on voting day, or voting during the advance voting period. Note that voting in Switzerland is restricted to one vote only (no re-(e-)voting) [38]. The polling card also has another 16-digit number attached to it, which is necessary for authentication using e-voting besides information about the voter (place and date of birth and password). This 16-digit number can be used to verify if the election authorities received the vote, which was encrypted with the public key of the election authorities [16]. Because the lack of real verification methods, this scheme is not considered as a plausible alternative for the Netherlands.

4.5 FINDINGS AND CONCLUSION

The above sections are concluded in the following paragraphs in words, and summarized in the Tables 4.1 and 4.2. For relevance, only Estonia, France and Norway are included.

*Summary of the
Estonian, French
and Norwegian
e-voting schemes*

Estonia

The Estonian e-voting scheme is ballot-secret, ballot-independent, receipt-free and complies with cast-as-intended, cast-by-me and contains-correct-vote verifiability properties. Voter authentication happens through a two-factor authentication method of both the eID card (or compatible SIM card) and the corresponding PIN codes. Both e-voting and (physical) re-voting is due in the advance voting period, after which e-voters can no longer vote on voting day. E-voting is offered to all eligible voters, living either in Estonia or abroad without any registration upfront.

France

The French voting system is ballot-secret, ballot-independent but does not comply with any of the verifiability properties. Voter registration happens through the knowledge of the consular identification number (NUMIC), name, birth date and passport number, after which a password can be chosen. Voter verification consists of a check for the NUMIC and the self chosen password. Voters can either vote by postal mail or e-voting during the advance voting period (depend-

ing on registration), or vote on voting day at an embassy or consulate if they didn't use the advance voting period. Re-voting is not offered. E-voting is offered to all eligible voters who live abroad.

Norway

The Norwegian e-voting scheme is ballot-secret, ballot-independence, receipt-free and complies with cast-as-intended, cast-by-me, contains-correct-vote and recorded-as-cast verifiability properties. Voter registration for e-voting prior to the elections is not necessary. Voter authentication happens through a two-factor authentication method of both the MinID card and a password. E-voting is offered to eligible voters of selected municipalities for the trial period.

Table 4.1: Privacy and verification properties for international e-voting schemes.

Country	ballot-secret	ballot-independent	receipt-free	coercion-evident	coercion-resistant	cast-as-intended	-cast-by-me	-contains-correct-vote	recorded-as-cast	tallied-as-recorded	universal verifiability	no registration needed	re-vote using e-votes	re-vote using physical votes	re-voting on election day
Estonia	✓	✓	✓			✓	✓	✓				✓	✓	✓	
France	✓	✓													
Norway	✓	✓	✓			✓	✓	✓	✓			✓	✓	✓	✓

Table 4.2: Voter specification for international e-voting schemes

Country	Eligible voters	Authentication method
Estonia	All voters living in Estonia or abroad	Two-factor: eID card / SIM card and corresponding PIN codes
France	All voters abroad	One-factor: Consular ID number, name, birth date and passport number
Norway	Voters in pilot municipalities	Two-factor: MinID card and corresponding password

PROPOSED FRAMEWORK

This chapter will describe the framework that was created for this thesis to operationalize the safeguards presented by [The Election Process Advisory Commission](#), allowing for the quantification of effort needed for an attacker to break any of these safeguards within an e-voting scheme. The effort needed for an attacker contains two main categories: time and financial investments. Time investments can be either active time (actually doing something) or passive time (waiting on someone else). Furthermore, time can also be split into time *before* the start of the voting period, and the time after that start. In this chapter it will become clear why this is an important difference. Financial investments contain both fixed and variable costs, of which the latter depends on the the attacker goal. The results of using this framework is not only a quantification of the current state of the e-voting scheme, but also a comparison to other schemes or a predefined baseline and a mitigation strategy to redesign the scheme.

This chapter will thereby answer research question two: *“How can these safeguards be operationalized to allow for quantitative comparison of the risks of different e-voting schemes?”*

First, section 5.1 will describe the different actors that play a role in the framework. After that, the capabilities of the attacker will be discussed in section 5.2, where after the actual framework will be described in section 5.3. This chapter ends with the conclusion in section 5.4.

5.1 ACTORS

In the case of an attack, several actors play a role. This section describes the informal modeling of these actors. Five actors are distinguished: attackers, voters, the election authorities, IT staff and auditors.

5.1.1 Attackers

In a study performed by Storer and Duncan [100], three types of attackers were identified from literature.

- A malicious election authority, intent on either denying an elector a vote, or observing how they voted. Such an attacker would be comparable to an abusive regime or government, intent upon thwarting the democratic process;

Outline

Actors

Attackers

- A malicious elector intent upon causing disruption to the election process in which they are entitled to vote;
- A malicious external attacker intent upon undermining the democratic process. They are identified as foreign espionage agents, criminal organizations, protests groups or even investigative journalists.

Note that the malicious election authority (bullet one) could also change the votes, which is not included in the study of [Storer and Duncan](#). For this research, this will be taken into account though.

These different attackers have different motivations, resources and funds to perform an attack on the democratic process. Furthermore, some of them wish their attack not to be detected by logging / auditing mechanisms, while other attacks are specifically designed to get caught and cause rumor of a rogue e-voting system.

To determine the attacks that can be performed by an attacker, it is necessary to describe his capabilities, and more generally, an attacker model. This is outlined in section [5.2](#).

5.1.2 Voters

Voters

Voters in e-voting schemes use their PC (or some other device like a mobile phone or tablet) at home, at work or somewhere else to cast their vote. A study performed by Trechsel and Vassil [[104](#), p27] showed a decline in online voting from other places than from home in an Estonian survey covering e-voters in 2005, 2007 and 2009 elections. This can partly be explained by the rise in internet connectivity among Estonian citizens, which could have offered them the opportunity to vote from home in later elections. For this research, every voter will be modeled as an e-voter voting from home or work/educational institution, using the most up-to-date statistics from that country. In Estonia, in the 2009 election, 76.6% of the e-voters voted from home and 21.7% voted from work or educational institution, for simplicity reasons, they will be rounded to 80% from home and 20% from companies/educational institutes.

Their device could be infected with malware, or their communication to the internet could be under the control of an attacker. For simplicity reasons, it will be assumed that a piece of malware is designed to work on all the devices citizens use to vote and is custom chosen per possible voter. If the framework is applied to countries that have completely different statistics on operating system usage ¹, this could be up for debate.

¹ China e.g. has a very high usage of Windows XP.

5.1.3 Election authorities

Election authorities

The election authorities are modeled as citizens being impregnable to coercion on a scale larger than one of them. This means that, when e.g. distributing shares of a cryptographic key, an attacker can get a hold of a maximum of one of the shares. Furthermore, only one member of the election authorities can be coerced into changing data in the systems or violating any form of confidentiality. More on this in section 5.1.6.

5.1.4 IT Staff

IT Staff

IT staff concerned with the implementation, testing, maintaining and operating of the e-voting systems have a crucial position within the e-voting scheme since they have both knowledge and access to certain parts of the system that are considered "secure" or "confidential". When modeling an attack, it could be that either the knowledge or the access of an IT staff members needs to be leveraged to pull the attack off. As with the election authorities, the IT staff will be modeled as coercion resistant except for one employee, furthermore, social engineering attacks or other types of attacks that involve the stealing of credentials e.g., are all maximized to one staff member only. More on this in section 5.1.6.

5.1.5 Auditors (or other external observers)

Auditors

As was shown in chapter 4, all of the examined voting schemes that are currently used for binding elections have a form of external observers, mostly called the auditor. In the proposed framework, the auditor has a special role as he/she observes the integrity of different crucial steps in the process of e-voting. These steps include checking who has physical access to critical servers, to checking who can digitally access them, to checking zero knowledge proofs on correct mixing and to validate the generating of the public-private keypair and much more ². In the recommendations by the Council of Europe [23, p19 with background on p58], special attention is paid to the audit functionality of e-voting systems:

- I. General
 - 100. The audit system shall be designed and implemented as part of the e-voting system. Audit facilities shall be present on different levels of the system: logical, technical and application.

² An interesting example of the tasks of the auditor in the Norwegian e-voting case can be found in Esteve et al. [39, p43-p45].

- 101. End-to-end auditing of an e-voting system shall include recording, providing monitoring facilities and providing verification facilities. Audit systems with the features set out in sections II – V below shall therefore be used to meet these requirements.
- II. Recording
 - 102. The audit system shall be open and comprehensive, and actively report on potential issues and threats.
 - 103. The audit system shall record times, events and actions, including:
 - * a. all voting-related information, including the number of eligible voters, the number of votes cast, the number of invalid votes, the counts and recounts, etc.;
 - * b. any attacks on the operation of the e-voting system and its communications infrastructure;
 - * c. system failures, malfunctions and other threats to the system.
- III. Monitoring
 - 104. The audit system shall provide the ability to oversee the election or referendum and to verify that the results and procedures are in accordance with the applicable legal provisions.
 - 105. Disclosure of the audit information to unauthorised persons shall be prevented.
 - 106. The audit system shall maintain voter anonymity at all times.
- IV. Verifiability
 - 107. The audit system shall provide the ability to cross-check and verify the correct operation of the e-voting system and the accuracy of the result, to detect voter fraud and to prove that all counted votes are authentic and that all votes have been counted.
 - 108. The audit system shall provide the ability to verify that an e-election or e-referendum has complied with the applicable legal provisions, the aim being to verify that the results are an accurate representation of the authentic votes.
- V. Other
 - 109. The audit system shall be protected against attacks which may corrupt, alter or lose records in the audit system.

- 110. Member states shall take adequate steps to ensure that the confidentiality of any information obtained by any person while carrying out auditing functions is guaranteed.

In a research performed by an independent institute for the Norwegian government, Esteve et al. [39] concluded that the Norwegian e-voting project complied with 85 of the 102 relevant recommendations, including those for the audit mechanism (except for recommendation 108³).

From this research, it can be assumed that the auditor does its job as he is supposed to do. For this framework it is assumed that for all the processes the auditor examines, it is impossible to forge, steal, delete or change data without being noticed by the auditor. As for the IT staff and the election authorities, it is possible to coerce or bribe at most one auditor, and not the set of auditors (if there are multiple involved). More on this in section 5.1.6.

5.1.6 *Bribing or coercing in general*

Bribes and coercion

As has been stated in the previous subsections, for this framework it is assumed impossible to bribe or coerce more than one member of either the election authorities, IT staff or auditors. This has been chosen on the basis of the chance of getting caught, which exponentially increases with every try of bribing or coercing someone concerned with the elections. This doesn't mean that it is impossible to achieve, and the model can allow for more with a few modifications. Though, a furthermore motivation to cap this number to one, is that this means that every process in the e-voting scheme that is looked at or operated by two or more members of these groups, can be considered a secure process. The modification that is concerned with this framework to allow for more coercion or bribes, would be an increase on the number of professionals that are involved with a process before it is considered secure.

5.2 ATTACKER MODEL

Attacker model

As discussed in section 5.1, there are restrictions on the number of actors involved that can be coerced or bribed by an attacker. Besides these restrictions, there are additional measures that an attacker has to take into account when planning his attack. For this framework, an attacker model was developed, which is mostly based on the threat model introduced by Dolev and Yao [30]: the attacker has full access to the network, he can see, analyze, replay, discard and edit every message, furthermore, he can add messages as well. The attacker is only bound by cryptographic constructs. For the attacker model in

³ Partial compliant because there was no open access to the audit logs [39, p22-p23].

this framework, a few changes were made. The attacker can do whatever he wants, except for the following:

1. Break non-trivial cryptography, though he can break non-trivial cryptography in the following cases:
 - a) Brute-forcing unsalted hashes from user-chosen passwords;
 - b) Brute-forcing per user generated passwords that are made up of relatively easily guessable properties and known method;
 - c) Comparing, altering or brute-forcing non-probabilistic public key cryptography with known public key;
2. Break secure connections between components;
3. Break any connection within an air-gapped environment;
4. Get hold of (parts of) the decryption key of the voting authorities;
5. Bypass the task of an auditor;
6. Coerce or bribe more than one member of the election authorities, IT staff or auditors;
7. Extract confidential information of or bypass PIN-code protection from smart cards;
8. Hacking secured non-user components (e.g. servers) and alter, delete or insert information or malware.

This is a somewhat more practical version of the original [Dolev and Yao](#) threat model, as it is not limited to network traffic only, but includes some of the system architecture and key actors as well. The restrictions for the attacker are reasonable, in the sense that most of them correspond to the ideas of the [Dolev and Yao](#) threats model (restrictions 1, 2 and 3), otherwise would have involve an unreasonable number of robbed, coerced or bribed individuals (restrictions 4, 5 and 6) or would include hacks into extremely hardened components (restrictions 7 and 8). The restrictions on the non-trivial cryptography have been relaxed to account for a more practical analysis of implementations of e-voting protocols.

5.2.1 *Adaptive attacker model*

The attacks that are found are quantified with cost and time efforts an attacker puts into successfully performing the attack. Depending on the skill level of the attacker, the number of attackers, the botnet the attacker already has, etc., these efforts could be either lowered or made higher. The attack trees are decorated with the help of tables

that list these attacker resources and capabilities. This framework allows for an *adaptive attacker model*, due to the fact that these tables can easily be changed to model the same scheme in this risk framework with other attacker resources and capabilities.

5.3 FRAMEWORK

Now that the attacker has been described, this section will address the framework and the steps involved.

To be able to find all possible attacks on e-voting schemes, to quantify them according to the effort for the attacker, to group them according to the [The Election Process Advisory Commission](#) safeguards, to compare them with other e-voting schemes or a baseline and to mitigate the risks of the e-voting scheme, this section will describe the framework. Within the framework, the described steps have to be carried out in sequence. After having implemented the mitigation strategies in a new design for an e-voting scheme, the process of finding, quantifying, categorizing, comparing and mitigating can be restarted to allow for new adjustments to the scheme to be evaluated. This circular process is shown in Figure 5.1.



Figure 5.1: Proposed framework

The different subsections that follow will each be concerned with a step in the framework. First section 5.3.1 will describe how to find attack vectors, then section 5.3.2 helps quantify these with the help of attack trees, section 5.3.3 shows how to group the quantified attacks according to the safeguards by [The Election Process Advisory](#)

*Outline of the
framework*

Commission, section 5.3.4 shows how to compare them with other e-voting schemes or a baseline on these safeguards and finally, section 5.3.5 will briefly discuss how to mitigate these attacks.

Note that all these attacks do not look for fundamental protocol level vulnerabilities or cryptographic weaknesses. These are all considered to be found using standard assessment of the protocol using formal verification, risk assessments on those levels, audits by cryptographers, etc.⁴. The attacks that are mentioned here, exploit the environments these e-voting schemes operate in.

5.3.1 Find

Find-step

Let alone that for a regular IT project it is considered impossible to do an exhaustive search for all possible risks that could occur, an e-voting project is much more complex and has much more dependencies. All of these e-voting projects evolve around both IT infrastructure as well as procedures and guidelines for physical actions. Weldemariam and Villafiorita [113] recognized this when they designed a methodological approach to do procedural security analysis. They have designed a method to formally analyze a workflow (i.e. a subprocess of an e-voting project) involving assets (e.g. votes, results, etc.) under the restriction of procedures (e.g. an auditor checking integrity or procedures to clean digital votes if they are overwritten with a physical one). Their goal is to identify potential attack vectors in such workflows in an automated and formal fashion and thereby approach an exhaustive search. Note that in their paper, they use processes around voting computers as their case study, that is why some figures will refer to those cases. After having used their method, numerous attack vectors will be identified. Before going on to the next step in the framework, it is necessary to sieve these attacks on the basis of feasibility. This will involve the use of domain experts.

Substeps

⁴ This is not considered unrealistic, especially since countries switching to e-voting do a very thorough and careful analysis of the considered schemes, and involve many experts of the academic as well as the business domain. The claim of the author of this thesis, is that the Dutch Government will also involve such an analysis, demonstrated by the various committees that have been involved with the previous voting projects:

<http://www.rijksoverheid.nl/documenten-en-publicaties/rapporten/2007/10/29/reactie-van-de-kiesraad-op-het-rapport-van-de-adviescommissie-inrichting-verkiezingsproces.html>,

<http://www.rijksoverheid.nl/documenten-en-publicaties/rapporten/2007/04/17/stemmachines-een-verweesd-dossier.html>,

<http://www.rijksoverheid.nl/documenten-en-publicaties/rapporten/2013/12/18/persbericht-cie-onderzoek-elektronisch-stemmen-in-het-stemlokaal-elke-stem-telt.html>

and <https://www.kiesraad.nl/nieuws/onderzoek-naar-internetstemmen-brengt-mogelijkheden-en-risico's>

The different substeps in this step of the framework are shown in Figure 5.8. They will be elaborated upon in this section. First, the [Weldemariam and Villafiorita](#) method is shown in sections 5.3.1.1, 5.3.1.2 and 5.3.1.3, where after section 5.3.1.4 explains how this can be applied in this step of the framework.

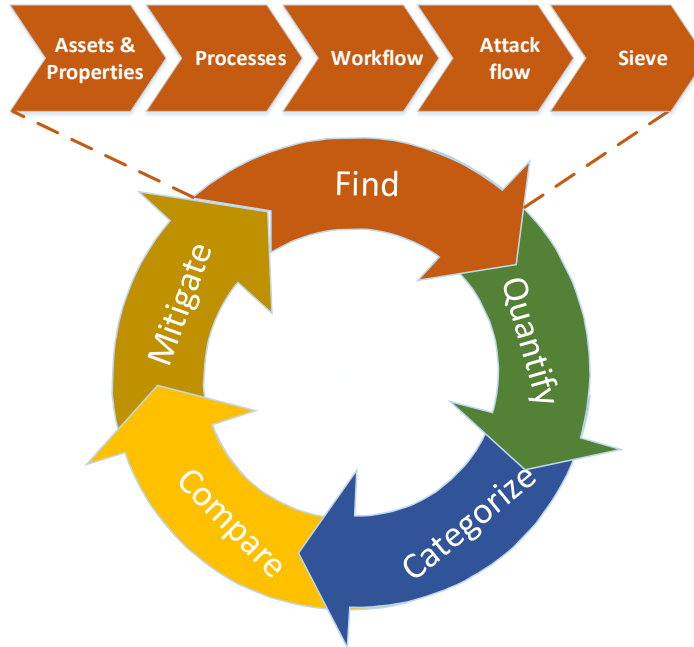


Figure 5.2: The framework, with the detailed steps for Find

5.3.1.1 Concept

In their method, they first describe every *asset* type belonging to a certain digital / manual process according to some properties (e.g. assets like a ballot box, a ballot, a vote, a piece of software, etc.). These properties are free of choice, and can later be used to model the threat model, examples of these properties could include IT Security features as “plain text”, “encrypted” or “signed”, they could also resemble real world properties such as “closed” (e.g. for a safe) or “sealed” and there are predefined properties “location”, “value” and “content”. Except for the predefined properties, all properties are free of choice and can be made up by domain analysts.

There can be multiple instances of each of these assets (e.g. multiple ballots of the asset type ballot). Every asset itself is either a *primitive asset* or a *container asset*. Container assets can carry other (primitive or container) assets. A container asset thus is both a location and an asset (e.g. a ballot box containing ballots). For this research, the “value” property is discarded since it is of no additional use.

Processes in a workflow create, delete or modify assets (e.g. signing a ballot (changing its “signed” property to true), or deleting an invalid one). Processes can have multiple assets as input and out-

Assets

Processes

put. At all moments in time, all assets have a certain “state” (namely their properties). Since those can only be changed by a process, the transformation of those assets throughout the run of several of those processes can be followed very carefully (called the *asset-flow*, which is different for each specific instance of an asset). Finally, each of these processes has one or multiple actors who can operate it.

Figure 5.3 visually shows the concepts described above. The concept of asset-flows is shown in Figure 5.4, in which the process on the left triggers the asset on the right to change state.

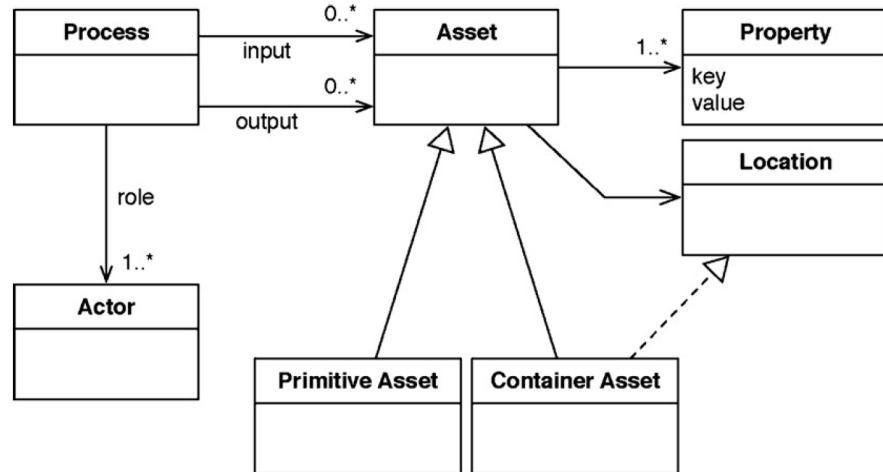


Figure 5.3: Concepts of the methodological approach to procedural security analysis [113, Figure 2]

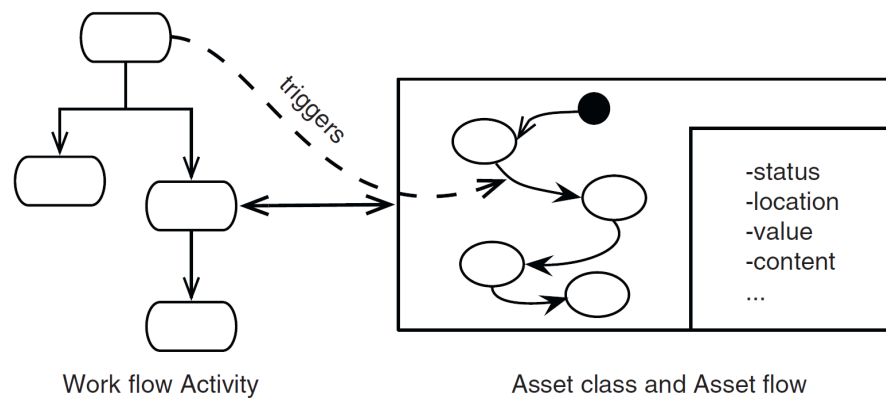


Figure 5.4: Asset-flow in a procedural security analysis [113, Figure 4]

According to [Weldemariam and Villafiorita](#), this description should be enough to model simplistic attacks on single assets: their contents are known, their locations are known and one can simply describe how to change the state of these assets in an undesired way. However, as the authors note, this is not enough to describe complex attacks in which there is a composition of threats against a set of assets. They propose to model attackers as actors that can operate newly created

malicious processes or use current processes by which they transform assets into a not-intended states. These (malicious) processes are then called *threat-actions* and are built from the basic threat-actions delete, read, write (or update) and create. E.g., a replace threat-action is composed with the basic threat-actions as follows: the original asset was deleted and a new one was created, the threat-action *replace* thus consists of two inputs and one output asset, see Figure 5.5. Put simply, by exhaustively adding all possible basic threat-actions to every asset recursively, all compound threat-actions per asset can be found.

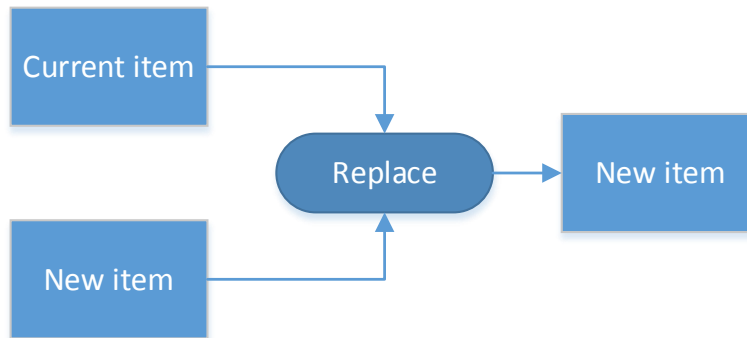


Figure 5.5: Replace process

Malicious assets

Assets that are under the control of the attacker (e.g. he can read the content or change properties) are called *malicious assets*. The attacker can reuse these assets in parts of his compound threat-actions, e.g. when the attacker knows some private signing key (since he read it) and he can control the content of some message (since he can write to it or can create it), he can effectively operate the compound threat-action of signing that message with the stolen private signing key. The attacker can thus also operate some known processes of the workflow, or he could try to get the knowledge on how those processes work before he can operate them. Note that these compound threat-actions can thus become very complex very soon.

5.3.1.2 NuSMV

NuSMV

[Weldemariam and Villafiorita](#) describe how to model both the workflow and the threat-actions in the symbolic model checking language NuSMV [18]. In their methodology, they first describe the regular workflow including the processes and assets in NuSMV, and have domain analysts decide on the properties of the assets and the working of the processes. They model the assets as finite state-machines⁵ that can change state whenever some preconditions are met, this nominal case is called “M-1” and is modeled without the presence of an attacker. The workflow is modeled similarly, having a “program counter” which decides what the next process is, what assets belong

⁵ They are called AFMs: Asset-flow models.

to it and checking if all preconditions are met, including those of actors and their roles. The combination of the workflow and the individual state-machines for assets work together and influences each other. M-1 forms the basis of the nominal situation, and is extended to form the basis of the extended M-2 model, which does include an attacker.

M1 vs M2

Transactions in the AFMs are modeled according to preconditions as well as information deduced from the program counter (thus, including the processes from the workflow model). Both M-1 and M-2 are modeled in NuSMV, in accordance with their semantic (which is based on the Kripke transition system⁶). Each of the assets is modeled as a finite state-machine with preconditions which can be drawn from the properties of the asset as well as on the roles of the actors and the program counter of the workflow. The threat-actions in M-2 are modeled as similar transitions, but include variables that specify what is happening (e.g. that the voting application is fake). The formal verification of the NuSMV tool will produce an output trace for each of the attacks, which thus includes these variables. This allows for both analysis of the necessary steps in the attack, and the way these attacks could be mitigated (i.e. by “turning off” this threat-action). A snippet of NuSMV source code from the paper of [Weldemariam and Villafiorita](#) is listed in Figure 5.6. It shows a transition from the voting machines software (their case description) into either encrypted software, encrypted software put in an envelop or plaintext software. All of these depend on the current state of the program counter (`pc.pc`) and some other boolean constraints (such as the current state of the content of the software (`content`) or location (`loc`) or active actors (`POfficerActive`) or random other constraints (`!fakekey`)).

⁶ A Kripke structure is a variation of nondeterministic automaton proposed by Saul Kripke, used in model checking to represent the behavior of a system. It is a simple abstract machine (a mathematical object) to capture the idea of a computing machine, without adding unnecessary complexities. It is basically a graph whose nodes represent the reachable states of the system and whose edges represent state transitions. A labeling function maps each node to a set of properties that hold in the corresponding state. Source: https://en.wikipedia.org/wiki/Kripke_structure

```

[...]
next(sw.content) := case
  (pc.pc = encrypt && content = PlainSW
   && loc = ElectoralOffice)
  || (content = EncryptedEnvSW
     && pc.pc = openEnvelope && POfficersActive
     && loc = PollingPlace)
: EncryptedSW;
(pc.pc = loadEnvelope && (TechnicianTwoActive
  || ElectoralServiceActive)
 && content = EncryptedSW)
: EncryptedEnvSW;
pc.pc = decrypt && POfficersActive
 && (status = Encrypted
   || content = EncryptedSW) && !fakeKey
: plainSW;
[...]

```

Figure 5.6: Snippet of NuSMV source code from the paper of [Weldemariam and Villafiorita](#)

5.3.1.3 Model Checking

After having described all necessary components of both M-1 and malicious threat-actions M-2, the authors go on by defining the formalized security properties using LTL or CTL ⁷, which works well with the Kripke structures in which the M-2 is defined. The NuSMV will then model check the complete specification and generate traces to failed security properties. The paper suggests a few of the security properties, including undetected attacks; denial of services; and reachability analysis of certain asset-threats (unwanted properties such as plaintext votes outside of the voting application [113, p1120]).

Model checking

The results of the model check form a trace which can be analyzed by the domain analysts for feasibility (both by domain experts from the (e-)voting domain as well as IT security experts). Certain threat-actions or asset-threats that were used in the attack could then be cut out to look for other threats, this is called *sieving*.

Results

Note that the authors do not claim that all possible attack vectors could be found using their technique, though besides getting a broader view on the examined workflow by thinking about the work-

⁷ LTL stands for Linear Temporal Logic and is used to check if certain conditions ever became true. CTL stands for Computational Logic Tree and is used to check whether a condition could ever become true.

flow in a structured way, it does provide complex traces to attacks that would probably not have been found without this extensive method. Furthermore, it allows for analysis of the mitigation strategies with only minor adjustments to the NuSMV code. In Figure 5.7, an example from the paper shows three possible attacks (in pink) in the workflow of getting the source code of a voting machine from the vendor to the voting machine, that were found using this structured approach. Attack one shows what happens if the attacker encrypts fake election software using the plaintext password he found; attack two shows a deny of service attack against the pincode with which the envelope was loaded; and attack three a denial of service attack by throwing away the envelope⁸.

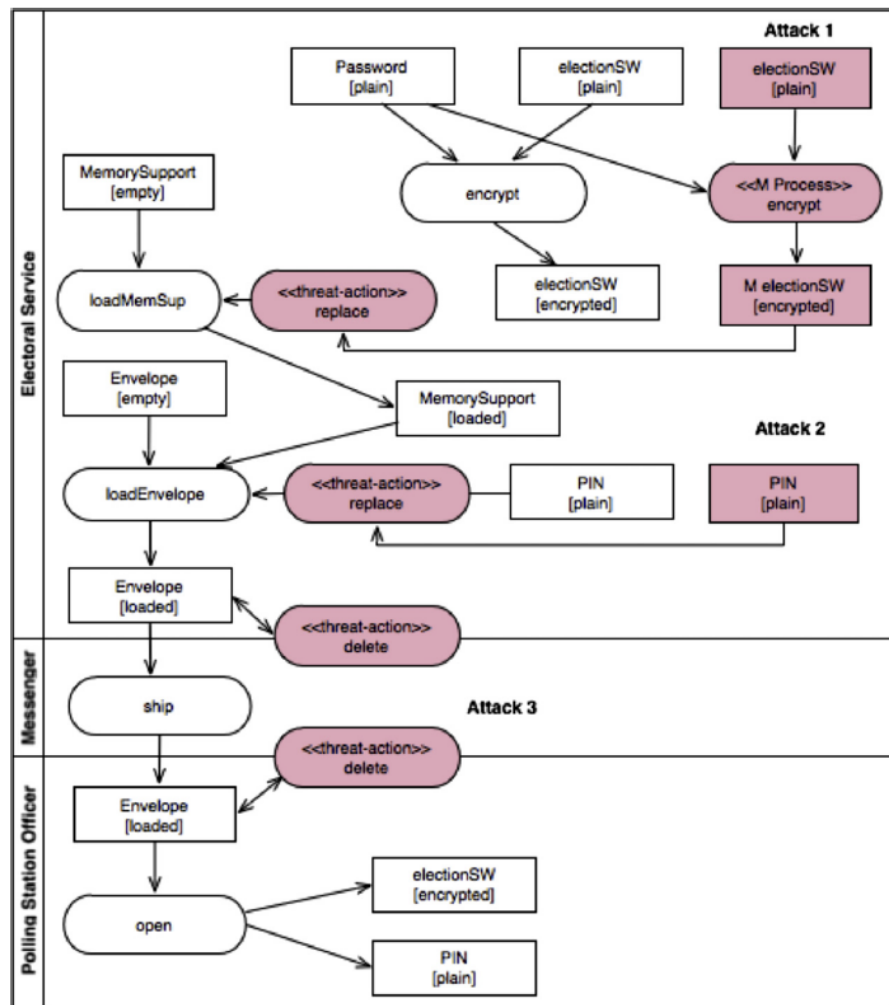


Figure 5.7: Example of found attacks in a workflow Weldemariam and Villafiorita [113, p1123]

⁸ For a full case description, the reader is referred to the original paper: Weldemariam and Villafiorita [113].

5.3.1.4 Application in this framework

Application

In this section, the method by [Weldemariam and Villafiorita](#) was presented to “exhaustively” find attacks in a workflow. Not only does this method show a more formal, model checking approach to find possible vulnerabilities, it does also allow for a more relaxed and informal analysis of what is actually happening in a particular workflow. The techniques described to absorb the information and knowledge of the domain analysts and output it in a structured and clear model including activity-diagram-like workflows and state-machine-like asset-flows allow for a much more intuitive way of searching for vulnerabilities.

The strength of this approach is not only to find all possible attacks using the model checking, but also to integrate the approaches of the domain analysts and the security analysts by providing a “talk book” to discuss the roles and actors associated with the assets and to see how the security properties evolve throughout the e-voting scheme.



Figure 5.8: The framework, with the detailed steps for Find

Substeps

To use this technique in the framework, first all assets, their properties and initial values need to be identified in accordance with domain specialist (step 1). Then, the processes to manipulate these assets need to be identified, as well as their actors (step 2), followed by the construction of the workflow. This is visually put together and then programmed in NuSMV code as a complete workflow (step 3). In the final step, the model checker is run to check for attack vectors (step 4), which are then carefully filtered by domain specialists (step 5).

5.3.2 Quantify

Quantify

Now that all realistic attack vectors of the e-voting scheme have been identified using the technique from [Weldemariam and Villafiorita](#), it is time to quantify *each* of them. As discussed in chapter 1, the quantification of the risks is based on the effort for an attacker to pull off an attack. This effort will be expressed in active and passive time efforts during advance and real voting days and financial investments, split into variable and fixed costs.

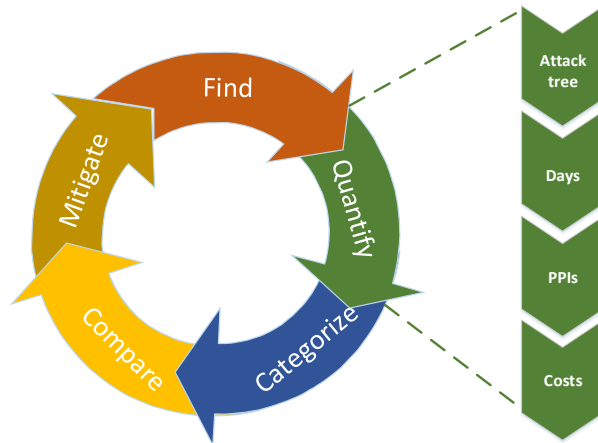


Figure 5.9: The framework, with the detailed steps for Quantify

To do so, the substeps depicted in Figure 5.9 for the quantify step need to be carried out for each of the attack vectors found in the previous step in the framework.

The following subsections will discuss these substeps in more detail. First it will be explained how to build attack trees from these attack vectors in section 5.3.2.1, then section 5.3.2.2 shows how to calculate the number of days involved, after that section 5.3.2.3 will show how to determine the number of victims necessary to successfully perform the attack and finally the cost calculations will be presented in section 5.3.2.4.

5.3.2.1 Attack tree

Building the attack tree

The technique described in section 5.3.1 to exhaustively find attack vectors helps to identify the different asset manipulations necessary to carry out the attack. These manipulations can be categorized into these two categories ⁹:

- Actors misbehaving;
- Processes misuse (either processes created by the attacker or misuse of existing processes).

⁹ Asset manipulation can involve creating, copying, changing or deleting assets.

This subsection explains how to build attack trees from these attack vectors found by the previous step in the framework. Attack trees provide for a structured and clear way of organizing different parts of an attack, they for example allow for reuse of parts of an attack, or provide for different attacker paths (OR-subparts). Furthermore, they are studied very well, and can thus, in large schemes, benefit from these studies ¹⁰.

Each of the corresponding asset manipulations could be noticed by voters, candidates, IT staff, election authorities, auditors or any other one. This is important to keep in mind when building the attack, especially if the goal of the attack is to stay unnoticed. But even if that is not an explicit goal, the moment in time the attack is discovered could be an important factor of the success of the attack. If, e.g., the attacker wants to show that the e-voting system is insecure by publishing the link between voters and their votes, and he gets caught before he published any of them, his desired impact is much lower.

ACTORS MISBEHAVING If misbehaving actors are a necessary step in the attack, they need to be “motivated” to misbehave. This could be done in multiple ways, but they are generalized into bribing and coercion (leaving the politically and emotionally motivated arguments out-of-scope). As explained in section 5.2, it is considered impossible to either coerce or bribe more than one actor to misbehave, but the framework could be extended to allow this.

PROCESS MISUSE If an attack vector includes asset manipulation due to process misuse, it is not immediately evident how this should be modeled in an attack tree. The attack vector that was in the output of the previous step of the framework, does not explicitly state the particular steps an attacker needs to take. Since domain experts from the IT Security domain were already actively involved in determining the possibility of certain attack vectors in that previous step of the framework, they should be actively involved into building the attack tree as well.

As shown in Figure 5.7, both the digital/physical location and the timing in the workflow where this threat-action should take place is output of the analysis of the previous step. It is for example impossible to sign a faked vote on the server side, if the server nor the attacker have a copy of the signing key of a particular voter, this should therefore be done at a place where the attacker can either copy the signing key, or misuse a process by which he let his fake vote be signed.

The physical/digital locations an attack manipulation resides can be split into three main domains, the client domain (voting application or the complete environment of the voter, including e.g. his out-of-band mobile phone), the server domain (all servers, internal

¹⁰ See section 2.4 for some background on attack trees.

networks, air-gapped media and auditor environments) and the connection between these two domains (mainly the internet or telephone networks). They are depicted in Figure 5.10.

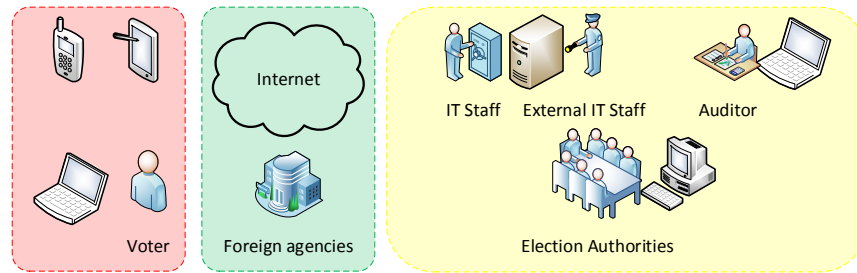


Figure 5.10: Three domains of locations in the voting process

Client domain

Attacks that involve asset manipulation on the *client domain* need an attacker to actually manipulate assets on either the clients voting medium (computer, tablet, phone, television, whatever will become a voting media) or on his verification medium (mostly phone, but in some schemes it could also be some other device connected to the internet such as computer, tablet, television, or whatever is connected to the internet). This could be done by either manually hacking these clients, or by distributing malware to already hacked computers. Due to the scale of many of these attacks (more on this in subsection 5.3.2.3), it is highly expected that pay-per-install services, as described in section 2.3, are used to get malware distributed on already hacked computers to avoid having to manually hack all of them. To infect computers with this custom malware, they need to be bought at those PPI-services.

For this thesis it is assumed that domain experts will help determine the number of victims necessary to successfully pull off the attack in the following section. The author of this thesis is sure that attackers will first distribute a placeholder piece of malware to the bought computers, which awaits the install of the custom-built malware, instead of having to wait for the actual malware to be built (which sometimes will take up until halfway into the voting days). Depending on the type of attack, it could be necessary to actually inject the voting application / website on the client computer or to just passively monitor either the screen or network traffic. Some of the attacks will involve having a C&C-infrastructure to either send commands from, votes or other info to or need some other form of control.

Not all attacks (e.g. an attack that deletes all votes) need such an infrastructure. In general, getting this specific piece of malware onto the computers used for e-voting during voting days will take an approach like the following:

- Possibly: buy C&C-infrastructure;

- Possibly: build C&C-infrastructure;
- Build asset manipulating code;
- Buy computers using a PPI-service;
- Distribute placeholder malware;
- Possibly: decompile voting application;
- Build fake voting application / malware;
- Distribute the malware to the placeholder malware;
- Execute the malware on the victims' computer.

This is visually depicted in an exemplary attack tree in Figure 5.11, with the “possible” steps shown in transparent attack node. All nodes are considered “and”-nodes.

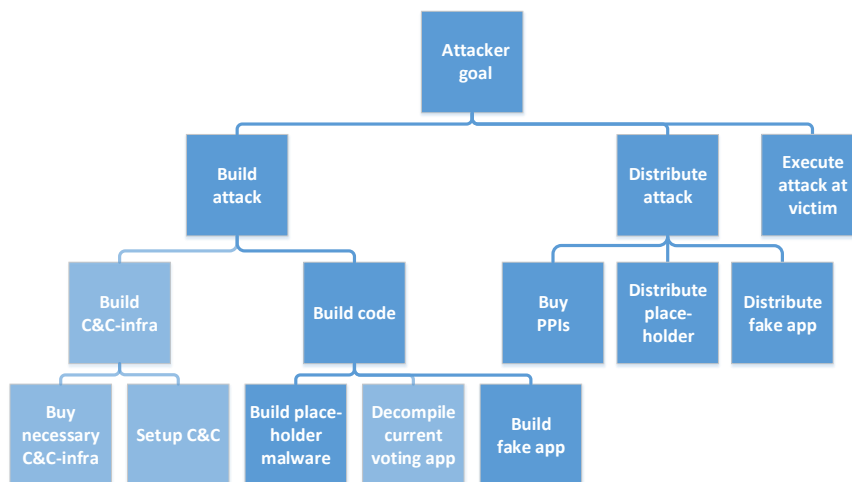


Figure 5.11: Example of an attack tree

For an attacker to do similar data manipulation on a mobile device, this is currently a bit harder. This has to go through a process of letting mobile phone users install a fake application from their application store [2], or to let them visit an infected website which tries to do a drive-by-download attack ¹¹. Awaiting what will happen in the world of mobile devices on PPI-services, and the ever changing number of rooted/jail broken devices which are much more vulnerable to attacks, the exact steps for getting the malware on these devices is left out-of-scope. Schemes in which PPI-services are offered are expected to be the most fertile for mobile devices as well ¹², because

¹¹ A drive-by-download attack is an attack which tries to exploit a weakness in the browser to install some malware on the device while just visiting a regular website.

¹² Blogs suggest that this is upcoming. Source: <http://www.webroot.com/blog/2014/06/23/peek-inside-commercially-available-android-based-botnet-hire/>

the attacker does not need to infect all these devices himself, and can easily select the citizens of the country at stake. In such a scheme, the steps are exactly the same as those for a computer.

Server domain

Attacks that involve asset manipulation on the *server side* of the spectrum need an attacker to actually manually hack into the servers. Considering that the servers for e-voting are only used for this service, they can be monitored for security incidents very efficiently: they do not offer any services other than e.g. receiving votes, sending voter lists, confirming identities. This makes for a very easy to manage monitoring system for both the IT staff and the auditors. Any regular hacking attempt, beginning with reconnaissance scanning for all services and ports is easily spotted due to its abnormalities in behavior and limited timeframe the monitoring needs to be done. This also counts for so called *advanced persistent threats* [52]. All traffic towards the server infrastructure could therefore be whitelisted, making monitoring even easier and hacking attempts a lot harder. This means that it is safe to assume that hacking this server infrastructure without being noticed can only be done using a zero-day¹³ that can be used to get remote access on the server or to retrieve information on credentials to get access. From there on, all the steps that are taken to manipulate the assets on the server side need to be: (i) done without the auditor noticing (if the attack should go unnoticed); (ii) redundant (if the infrastructure has some redundancy built-in); (iii) should also effect all verification methods (if those need to be effected as well); (iv) and should be persistent (they should not be easily reversible). The exact order in which steps should be taken, are very much dependent on the attack itself: in some attacks there need to be persistent communication between the voting servers and some C&C-infrastructure; some attacks need to have a persistent piece of malware doing repetitive tasks like changing votes and some attacks only need the deletion of votes to be effective; etc.

Between client and server domain

Attacks that involve asset manipulation *between these two domains of server and client*, need to have an attacker behaving as a man in the middle or man-next-to-the-middle (being a second party that fakes being a man-in-the-middle). While most of the traffic on the internet between the client and server is cryptographically protected using TLS, it is unrealistic to assume a real classical man-in-the-middle in this environment¹⁴. All man-in-the-middle attacks that focus on either the client or the server have been described earlier, leaving the man-next-to-the-middle. As shown in section 2.3, an attacker can fake SMS-messages coming from the voting authorities. Thereby, he lets

¹³ For information on zero-days, see section 2.3.6.

¹⁴ It is safe to assume this, given the earlier described carefully designed protocol and audited cryptography. Any other man-in-the-middle attack would be unrealistic.

the voter believe that the SMS was genuine with which he could fake some out-of-band verification method. This is an example of asset manipulation (in this case creation) in the connection between the client and server domain, which in this thesis is a so called man-next-to-the-middle.

5.3.2.2 Days

The previous substep of the quantify step in the framework explained how to transform the attack vector into an attack tree, which could look like one displayed in Figure 5.11. This substep will help decorate this attack tree with the number of days the attacker needs to put into performing this step. This is a vital substep to eventually denote the costs of the attack, consider e.g. an attack in which votes are changed: the longer it takes to setup the attack, the less probable it is to successfully infect a computer that is used for e-voting in the remaining days of the elections, and therefore more computers need to be bought with a PPI-service, thus increasing the costs.

First, the attack tree needs to be transformed into a time frame, to visually show the dependencies of different steps in the attack. An example is shown in Figure 5.12, which depicts an attack on the availability of the voting system (mostly based on the Serbian DDoS attack found in section 2.3.2).

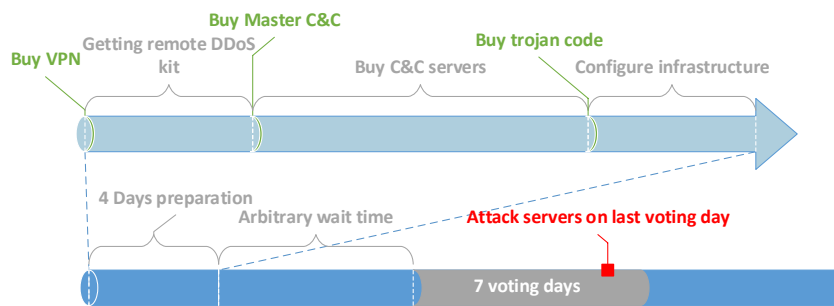


Figure 5.12: Time frame example

Each of the steps in this time frame can be decorated with a time investment with the help of Table 5.1, which separates both active and passive time, and separates advance and actual voting days, as explained in section 5.

The table is constructed with time investments from a case study and from estimated amounts. The case study was described in section 4.1.4, it showed that a voting application can be decompiled in a matter of 4 to 5 working days, together with building a fake version of the same application. This will be the standard that is used as part of the attacker model. An exception is made for malware based on

Days

Time frame

Time decoration

decompiling the voting application and building in a simple confidentiality stealer, which will take about 2 working days.

The other time investments are estimated from the research conducted in section 2.3. It is for example estimated that it will take an attacker a few days to buy a large amount of PPIs. For this research it will be assumed that an attacker can obtain about 30,000 a day, however as most of the use cases in which an attacker will buy these PPIs will show, the attacker buys them way before the voting period and therefore is not really restricted by this time constraint. The time constraint to deploy the custom-built malware to the placeholder malware on the PPIs is estimated at about a day (since most PCs will be online every day), which mostly does happen during the voting period and thus is considered a definite time constraint. Botnet consultation with a professional will take a day to arrange, but this will again take place in the advance period, before the actual voting period and therefore is not really time restricted. Similarly, refining standard botnet code is done in the advance period before the actual voting period, and will take about two days.

Table 5.1: Time constraints for the attacker model, based on research from section 2.3

Task	Advance or actual voting days	Active time in days	Passive time in days
Decompiling voting app and building fake version	Actual	4 to 5	0
Decompiling voting app and inserting vote listener	Actual	2	0
Buy 30,000 PPIs in one country	Advance	0	1
Deploy the actual malware to placeholder malware	Actual	0	1
Botnet consultation	Advance	1	0
Refining standard botnet code	Advance	2	0

5.3.2.3 PPI

Pay-per-install services

Not all attack trees include a step that involves pay-per-install services, but those that do need to be further equipped with the exact number of PPIs to be bought before the costs can be determined. First, the target number of victims should be known (e.g. the number of votes that need to be changed in order to gain one seat in parliament, the strength of a DDoS attack, etc.), this can be added to the root

node of the attack, from where on the underlying number of victims can be determined. To achieve this, statistics on numerous of ratio's are needed, e.g. the total number of computers in a particular country, the number of e-voters, the percentage of computers used for e-voting, the number of e-voters in the voting days that are actually left after the setup time, etc. Especially this last statistic is very important, take for example an attack that takes a setup time of 5 days in a total of 7 voting days: the number of votes that will still vote will probably have dropped enormously. An example of the number of voters per hour in Estonia is shown in Figure 5.13. It is thus important to determine the number of victims that that vote on different days during the voting days, and to consider the percentage of voters already "lost" due to this setup time.

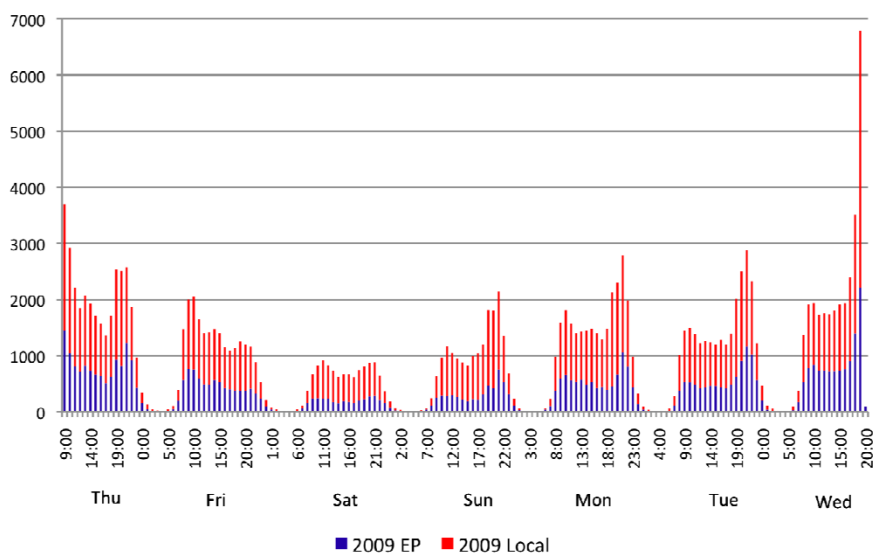


Figure 5.13: Estonian votes per hour during voting days

Additionally, consider the fact that every bought computer has a certain success-probability of having the placeholder malware installed, getting online during the remaining days of the voting days after the malware is ready and the chance that this particular computer will be used for e-voting at all, especially during the remaining days. Quite quickly, taken this into consideration, the number of PPIs rises to tens of times the chosen minimal number of victims impacted at the root node.

Note that some of the actions that take place in the voting days could be moved towards the advanced voting days depending on the publication date of the voting application or some other events. These events need to be taken into account when constructing the time efforts.

In the next chapter, two attacks that are heavily reliant on the number of PPIs bought will be discussed at length, whereas the DDoS ex-

ample in this chapter is rather straightforward and does not involve any setup time during voting days.

5.3.2.4 *Costs*

Costs

All components of the attack can now be put together in the attack tree, including the number of PPIs to buy. This step of the framework will decorate the attack tree with the costs involved with the attack. All costs need to be split into variable and fixed costs, this allows for the construction of a function from the number of the to be affected votes towards the costs.

Coercion

Coercion is considered to involve either a lot of time or high chance of getting caught, and is left to specific domain analysts to determine an effort in terms of time and money that is involved into coercing someone. For the Dutch case it goes that bribes are very rare in the Netherlands [19, p59]¹⁵, therefore the actual amount of effort necessary to bribe someone concerned with the voting process seems extremely hard to estimate, and will be left to the individual case of the people being bribed. It is left out-of-scope how to give some basic figures on the money and time investments to bribe someone, but the framework allows for such modeling.

Note that the chance that an attack is caught extremely increases when bribes or coercion is involved in the attack, since they could step up to the police or other authorities. Furthermore note that processes that are under the supervision or under the control of two or more people from the IT staff, election authorities or auditors, can be considered safe, due to the fact that in the current model, only one of them could be bribed or coerced.

Digital underground

The different steps in the attack need to be carried out with a particular precision. Fortunately, the digital underground offers consultancy services for many different aspects of attacks, and it is therefore considered smart to involve those while setting up the attack. Furthermore, the attacker could want to stay anonymous, and fortunately the digital underground also offers services for that, like proxies or virtual private servers that offer bulletproof-like anonymity. To setup a C&C-infrastructure, the maximum number of connections to one of the C&C-servers should be carefully managed, an estimated 600 computers could be managed by one of them. The set of these C&C-servers is managed by one mothership C&C-server who only communicates with the other servers and is protected using a bulletproof hosting provider with DDoS-protection to ensure that this

¹⁵ Research over the period of 2013 shows that the Netherlands is the 8th best ranking country in the world considering the Corruption Perception Index (CPI). Source: <http://cpi.transparency.org/cpi2013/results/#myAnchor1>

server won't be taken down [86]¹⁶. An example of how such a C&C-infrastructure could look like can be found in Figure 5.14.

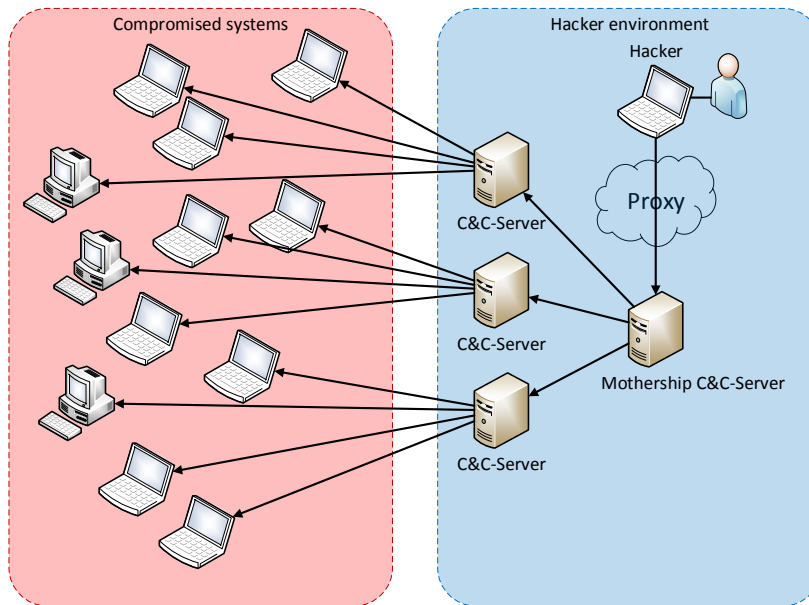


Figure 5.14: An example of a C&C-infrastructure.

Furthermore, extra care needs to be put into distributing the first placeholder malware to fool any honeypots setup by the election authorities or CERT-teams [110, 49]¹⁷, therefore all malware samples (e.g. the placeholder malware or the fake voting app) need to be encrypted before they are sent to the victims.

Cost decoration

With all these considerations and details for implementing the steps of the attack tree, and quantifying the number of victims in mind, the different steps in the attack tree can be decorated with the costs that were given in Tables 2.12 to come to a decorated attack tree for each of the attack vectors found in the “find”-step of the framework. A part of the tree could look like the example in Figure 5.15. The costs are in the nodes, together with the time effort (active advanced | passive advanced | active actual | passive actual voting days). The red node depicts the variable costs, while all other nodes are fixed costs.

¹⁶ Radunovic [86] used a similar setup with the same figures for their DDoS of the complete Serbian IT-infrastructure exemplary attack.

¹⁷ It is known that e.g. the Estonian CERT and election authorities setup honeypots to try to sense what is going on during the elections [110, 49].

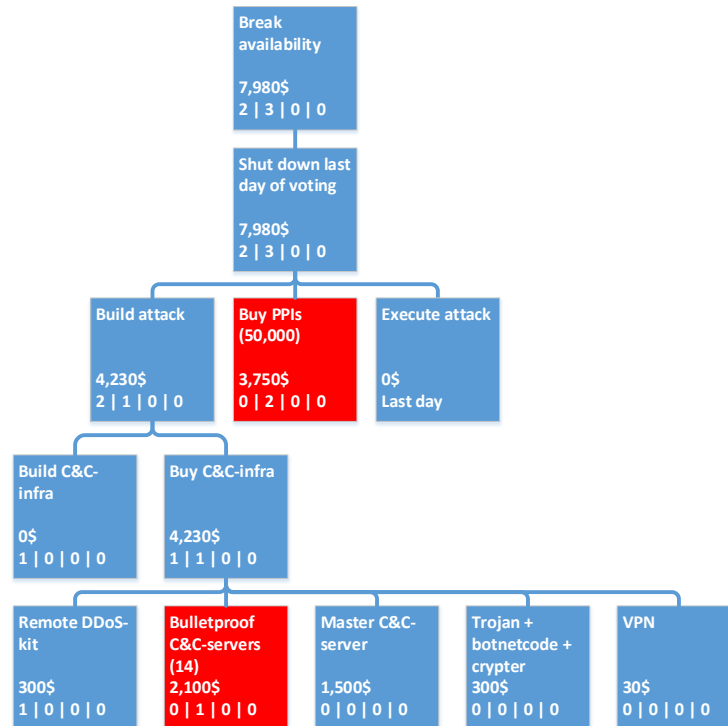


Figure 5.15: An example of a quantified attack tree

Cost function

Using this decorated attack tree, a total cost function can now be generated. This function can have several types of input like “the infrastructure’s network capacity” (like in this example), “the number of votes changed” (which will be shown in the next chapter), etc. Based on this input, the total cost can be calculated. In some cases, this will be rather straightforward, but it could also be the case that it forms a challenge to define such a function. In the DDoS example, the number of bulletproof C&C-servers and the number of PPIs are concerned with variable costs, while the other costs are fixed (note that the PPIs concerned with this attack can be bought globally whereas in most attacks they need to originate from a particular country). A total cost function would thus include those fixed costs and a multiplication of the input and a fraction of the variable costs. If the infrastructure to be DDoSed would be twice as resilient, the number of PPIs must be doubled, and thus the number of bulletproof C&C-servers as well. In the examples in the next chapter, two examples will show how to incorporate the number of affected votes in such a function.

Note however, that when the number of active voting days is affected by an increased number of victims, this has consequences in the statistics on the actual victim rate (chance that a proposed victim will actually fall victim in the left time frame). This thus heavily influences such a cost function.

5.3.3 Categorize

The goal of this step in the framework (see figure 5.16 for reference) is to categorize all the attacks that were found and quantified using the previous two steps in the framework, this allows for an objective comparison of this e-voting scheme with either a baseline or another e-voting scheme in the following step of the framework.

Categorize



Figure 5.16: Proposed framework

As discussed at length in chapter 3, the Dutch approach to safeguards conforms with the international standards (specifically with respect to law, ethics, literature and approaches by other countries, as shown in appendix B). In section 3.3 though, it was established that, from an information security perspective, the safeguards do not fulfill all the necessary requirements. A few of them were mentioned as being out-of-scope or too detailed, but three main topics were found to be very important, namely login credentials; number of cast votes per voter; and the voting system in general.

In this section, two methods will be presented to determine the safeguard of each attack. The latter of the two will be used for this framework, while the first can be used as a reference to show that in fact a lot of safeguards are broken when an attack takes place. The choice for that latter is based on the fact that it gives more guidance to the actual problem caused by the attack in comparison to the first categorization method, which focuses on showing all weaknesses exploited by the attack.

5.3.3.1 Information Security Requirements Approach

To group the different quantified attacks according to the safeguards as presented by [The Election Process Advisory Commission](#), a quick reference to Table 3.1 could index most of the steps in the attack, and find their corresponding safeguard(s). This could however mean that,

Information security approach

when taking e.g. an attack on the integrity of the results, both the verifiability and fairness safeguards are violated. However one can easily see that the real problem with the integrity of the results is the fairness safeguard, not the fact that the verifiability is broken, since verifiability stems from fairness and should push forward the fact that elections are fair. But since all safeguards help the fairness property, as shown in the dependency tree of safeguards in appendix B.2.3, it doesn't provide a lot of information if all attacks would be categorized as breaking the fairness safeguard.

This shows that it is not very straightforward to categorize attacks according to these safeguards. Even more problematic are attacks that are complex and involve the breaking of multiple of the information security requirements. Take for example the attack on the integrity of the results in the Estonian e-voting scheme which will be discussed in section 6.2.1. The attack involves breaking the integrity of the votes, the software, the verification mechanism and the results, the authenticity of the voting application and the verification mechanism and it could be considered to also break the accountability of both the correct design and implementation of this scheme. This would mean that the safeguards verifiability, fairness, free suffrage, secret suffrage and equal suffrage would be broken.

*Need for another
method*

For this research, the approach is to categorize each attack in exactly one safeguard. This approach was chosen on the basis of the actual attacker goal: what will he be able to do. Arguing that all steps he took also need to be categorized fails at addressing the real issue of the attack with the right safeguard. The Estonian integrity example shows that attacks on such a fundamental point of democracy: integrity of the results, would be miscategorized into multiple safeguards that are not really at stake (such as verifiability). The following subsection will discuss another method of categorizing attacks, which has the preference over overcategorizing.

5.3.3.2 *Grouping attacks according to the attacker goals*

*Attacker goal
approach*

Another approach to group the attacks, is to consider the goal of the attacker. As discussed in section 5.1.1, there are several attackers with different goals. The attacker goals can be defined as to change votes/the results; break the confidentiality of votes; breaking the availability of voting system; or to break the faith in the system. The attacker can have the motivation to have this attack be detected or not. The different attacks are grouped according to this principle, taken into account the description of the Dutch safeguards. The following enumeration of these safeguards includes a rough and informal description of the attacks that fall into this safeguard.

- Transparency: attacks that try to break faith in the e-voting system, but limited to those that actually focus on the transparent

way the system works (e.g. news that mistakenly claims the system is not safe);

- Verifiability: attacks that focus solely on verifiability, trying to break faith in the e-voting system by means of the verifiability mechanism (e.g. sending fake SMS messages after someone voted);
- Fairness: attacks that *modify* votes to modify the results (e.g. changing unwanted votes);
- Eligibility: attacks that *insert* votes (done by *outsiders*) to modify the results (e.g. adding additional external votes to the ballot box);
- Free suffrage: attacks that narrow the freedom of voting for the voter, either conscious or unconscious (e.g. removing some candidates or parties from the candidate list or by publishing early results during elections already, aimed at persuading the voter to vote otherwise);
- Secret suffrage: attacks that create any form of linkability between the voter and his vote or attacks that enable the voter to get a receipt (e.g. a vote “stealer” that wiretaps the voter);
- Equal suffrage: attacks that *insert* votes (additional votes from *eligible voters* (letting the same vote count more than once)) to modify the results (e.g. starting with a non-empty ballot box);
- Accessibility: attacks that limit the accessibility of the e-voting system for voters, either conscious or unconscious (e.g. DoS attacks or deleting unwanted votes).

All attacks are appointed to one of these safeguards, as compared to the previous method, which allowed for categorizing an attack into multiple of the safeguards. Note that some of these safeguards tend to share some attacks, although these descriptions try to separate these as much as possible. It may be noted that this categorization is definitely not perfect, and new attack surfaces could emerge that can't be fit into any of these safeguards on the basis of the above description. On the other hand, the idea behind this categorization is clear, and links the description of the Dutch safeguards to the goal of the attacker. New attacks that emerge and that can't be placed in any of these safeguards on the basis of this description should therefore force the broadening of these descriptions given above.

5.3.4 Compare

In the previous steps of the framework, all the attacks on an e-voting scheme have been found, quantified and categorized according to

Compare

The Election Process Advisory Commission [103] safeguards. In this step of the framework (see figure 5.17 for reference), the e-voting scheme that is up for inspection will be compared to similar e-voting schemes. This can either be done by going through the same process of finding, quantifying and categorizing attacks within other schemes or by going through the same process on a subpart of another e-voting scheme. What can come out of the earlier steps in the process might indicate that a particular piece of the e-voting scheme is more vulnerable than others, facilitating most of the attacks. It makes more sense to compare the categories of attacks to e-voting schemes from The Election Process Advisory Commission [103] on those more vulnerable parts of the e-voting scheme up for inspection.



Figure 5.17: Proposed framework

Baseline comparison

Besides comparing multiple e-voting schemes with each other, one e-voting schemes minimal efforts can also be compared to some baseline set by government bodies. Imagine that the government decides that the Fairness safeguard should not be broken under a certain amount of financial effort. Then this compare-step in the framework enables the comparison of the e-voting scheme with this baseline.

The comparison-step outputs not only a factual comparison based on the safeguards, but also identifies the facilitators of the attack. When the e-voting scheme is compared to other schemes, this identification process seems to make more sense. In general though, when plotting the attacks that involve the least attacker effort into the workflow diagrams of step one again, these common attack facilitators are easily found.

When comparing to the baseline, all attacks that stay below the baseline effort need to be mitigated in the next step. Therefore, all the attack facilitators for these attacks should be identified and used for the following step.

5.3.5 Mitigate

Mitigation

Based on the comparison of the e-voting scheme with either the baseline or other e-voting schemes in the previous step (including the identified attack facilitators in the workflow), this step in the framework (see figure 5.18 for reference) helps to identify mitigation strategies to mitigate these attacks.



Figure 5.18: Proposed framework

Attack facilitators can be very specific processes that need small readjustment, or missing checks at either client or server side which can be fixed with relative ease (e.g. only publish the voting application on the first voting day instead of prior to that). In other occasions the attack facilitators are very fundamental process flows or parameters that came together after thorough analyses as well-reasoned design choices. Take for example attacks that decompile the voting application and rebuild it with malicious code in it, they benefit from having a longer voting period because this decompiling and rebuilding needs time and can only be done after the start of the voting period. Note that there is a huge difference in voting days between Norway and Estonia (see section 4.5), but that choice for the number of voting days was a rather essential one in these countries. It came together after thorough discussions between politicians and the voting authorities. This shows that some of the attack facilitators do not allow for quick fixes.

This step in the framework sometimes requires to go back to the drawing table and to rethink certain decisions. This step benefits from having the scheme been compared to other (similar) schemes in the previous step. Since some of the decisions that have been taken were highly motivated by politics, it could involve workarounds for choices like the number of voting days or the level of transparency given about the e-voting scheme.

In total, the different steps in the framework are as displayed in Figure 5.19.

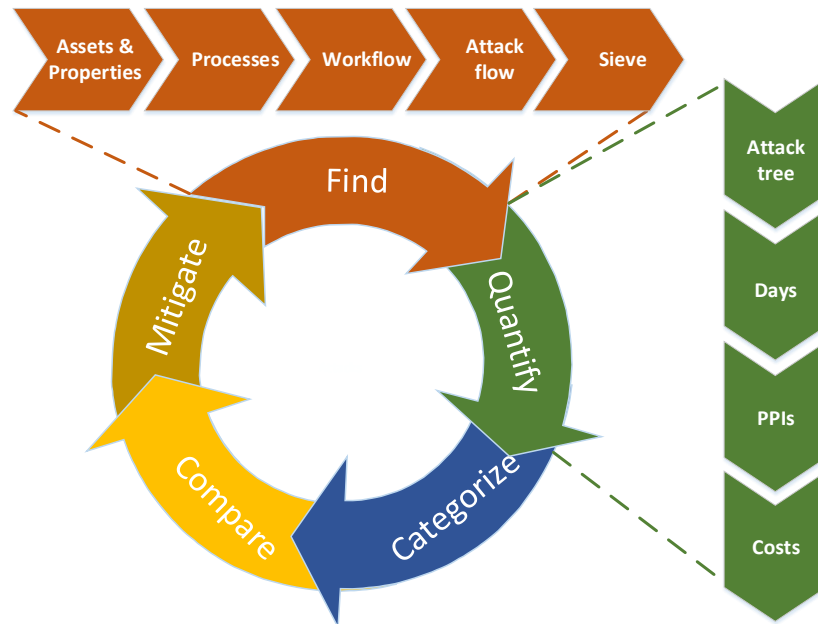


Figure 5.19: Complete framework including substeps

5.4 CONCLUSION

Conclusion

This chapter presented the framework that was the result of this research, consisting of five steps: finding, quantifying, categorizing, comparing and mitigating attacks in e-voting schemes. The steps in the framework are circular: after having mitigated the different attacks found in the e-voting scheme, the same framework can be applied to again find, quantify, categorize, compare and mitigate attacks. This chapter also showed the different actors concerned with e-voting schemes, and in particular the attacker was extended with an attacker model.

The proposed framework answers the research question 2: “*how the safeguards by The Election Process Advisory Commission can be operationalized to allow for quantitative comparison of the risks of different e-voting schemes*”.

This seemingly theoretic framework will be used in practice in the next chapter to examine some of the features of the Estonian e-voting scheme.

This chapter will present how the framework can be applied to the real case of e-voting in Estonia ¹. Estonia was chosen based on the elaborate statistics that are published, and the well described scheme. First, section 6.1 will briefly show what the *find* step of the framework looks like in practice. After that, section 6.2 will address the *quantify* step for two attacks that were found with the first step of the framework. The last three steps of the framework (*categorize,compare* and *mitigate*) are briefly covered in section 6.3.

Exemplary attack

The two attacks that will be quantified using the quantify step are (1) an attack on the integrity of the votes, how can a seat in parliament be “bought” as an attacker? This attack should go unnoticed by the authorities in order to function correctly; and (2) an attack that by definition does not go unnoticed. The attack focuses on the secrecy of votes for certain e-voters. Their votes will be published online, to attack the faith the citizens have in the e-voting system.

This chapter will partly help answering research questions four: “Apply the formalized safeguards to an e-voting scheme and verify the results and the framework.”

6.1 FIND ATTACKS

To show how the framework works, the *find* step of the framework will briefly be shown in this section. This section only covers how to model a part of the Estonian voting scheme graphically, but will not cover the whole scheme, nor will it provide the NuSMV syntax to test this voting scheme in a model checker. The part of the voting scheme that will be covered by this section, is from downloading the voting application to casting a vote and verifying this vote. To visualize this process in a workflow as described in section 5.3.1, first the different *assets* and *processes* need to be defined, including their properties. Thereafter, the workflow needs to be constructed with corresponding actors. The model checker can then output several attacks, of which two will displayed in the workflow. Last, the attacks need to be sieved for feasibility. These steps can be found in Figure 6.1.

Find

¹ For information on the Estonian voting scheme, see section 4.1.

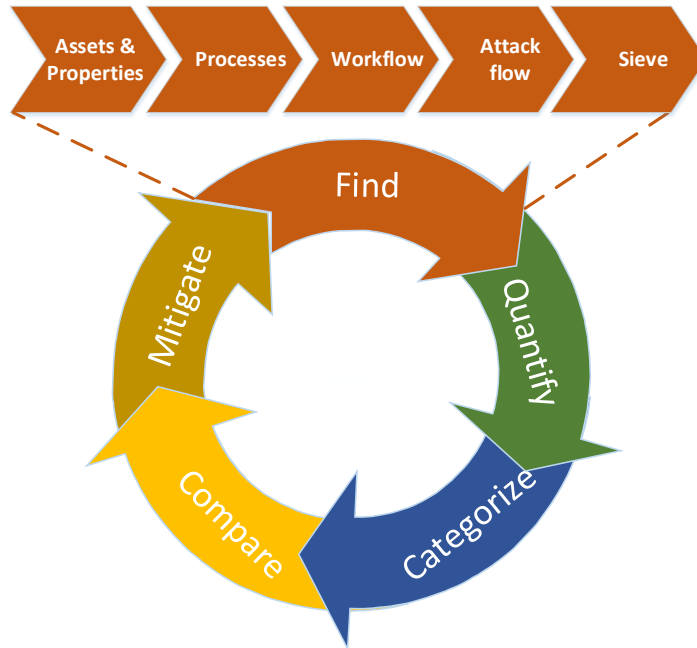


Figure 6.1: The framework, with the detailed steps for Find

6.1.1 Assets & Properties

Assets

When trying to define the different assets and properties concerned with the Estonian e-voting scheme, it is important to have a simplified view of the process as depicted in Figure 6.2². What can easily be distinguished as assets in the steps 1 to 3 of this Figure are the *voting app*, the *vote*, the *public key of the voting authorities*, the *private signing key* and the *eID*. When taking a closer look at the working of the eID, it becomes apparent that those need a *PIN₁* and *PIN₂* to function. The verification mechanism, as explained in Figure 4.2, additionally needs a *session ID* (OTP in the Figure), a *random* (Rnd in the Figure) and outputs a *QR-code* to a *mobile device*.

While identifying these from the descriptions is a relatively straightforward procedure of almost looking for any noun in these Figures, and deciding whether or not it would suit to be an asset in this part of the voting process, it seems harder to check whether all assets have been identified. As it turns out however, when starting to build the actual workflow, it can be easily seen whether or not all assets have been identified, as these assets would be missing from a correctly functioning e-voting scheme.

Properties

Now that a few assets have been found, covering the most presumable ones, the properties of these assets have yet to be defined. As discussed in section 5.3.1.1, these can be constructed by domain experts but should at least include *location* and *content*. In the initial phase, before the workflow starts, these need to be defined (or explic-

² This is a copy of Figure 4.1.

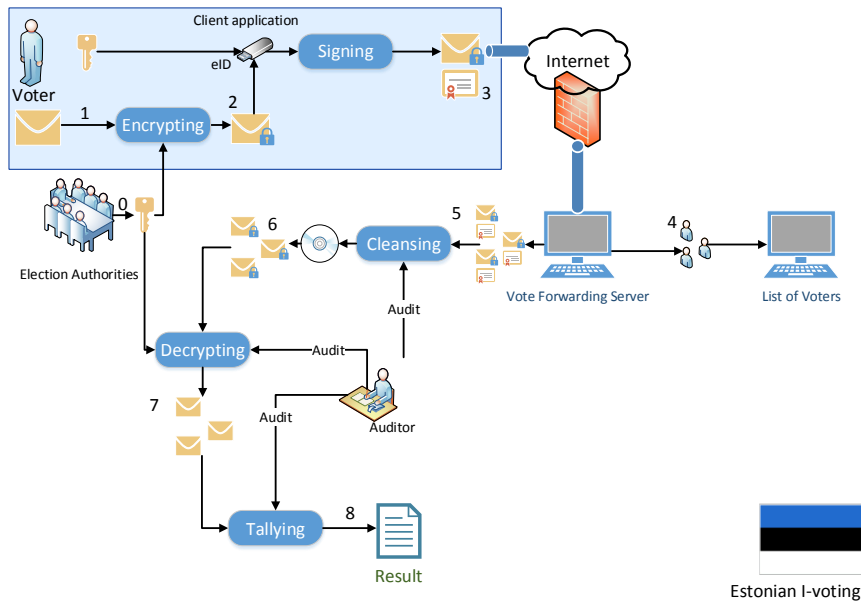


Figure 6.2: Estonian i-voting scheme (simplified)

itly be mentioned as empty), as well as the other properties identified by domain experts. Some of the assets have no other function than a passive one, for example the PIN_1 and PIN_2 assets are only entered into the eID and are not manipulated (by a process) in any way.

The assets that are not manipulated by any process are identified as being the *public key of the voting authorities*, *private signing key*, PIN_1 , PIN_2 , *SessionID*, *Random*, *QR-code* and *mobile*. Note that these assets can however be faked, or contain fake data. This inherent property will be attached to these assets in a later stage, when modeling the extended M-2 model that also includes attack vectors. Furthermore, to allow for the scope of this example, the *mobile* is modeled as an asset with no manipulative properties, which in a real world situation is not true of course.

The assets that have properties that are manipulated by processes are the *vote*, *voting application* and *eID*.

VOTE The vote in this model can either be encrypted or not and signed or not. Through all the stages of Figure 6.2, the vote is in either of these stages. No other properties belong to this asset.

VOTING APPLICATION Figure 6.2 clearly shows the role of the voting application. It goes through a few stages, from first having entered the vote into it, building the random, then encrypting the vote, letting it be signed by the eID, send the vote to the voting authorities and finally building the QR-code. Note that all these steps include other assets (the *vote*, *public key of the voting authorities*, *random*, *sessionID* and the *QR-code*). None of these properties actually belongs to

the voting application itself. The voting application will therefore be modeled as a container-asset with no additional properties. This is visually depicted in Figure 6.3.

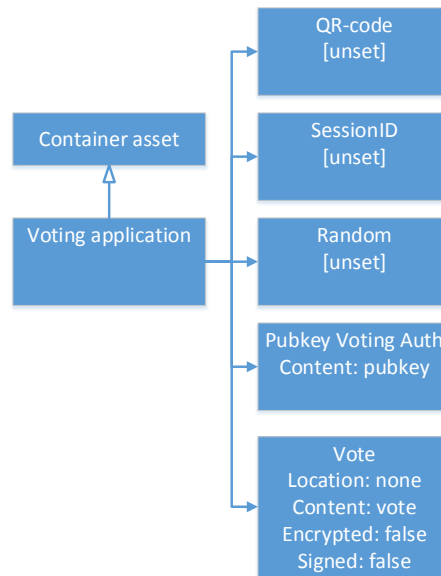


Figure 6.3: Conceptual visualization of the voting application

eID The eID is used for signing the encrypted vote, after first authenticating with PIN₁ and then with PIN₂ to actually validate the signing. To model the eID, both the vote and the private signing key are assets that can have their location set to be the eID. The PIN₁ and PIN₂ are modeled similarly. The eID itself doesn't hold any manipulative properties, but does use a location property to denote whether it is in the card reader or not. The eID is visually depicted in Figure 6.4.

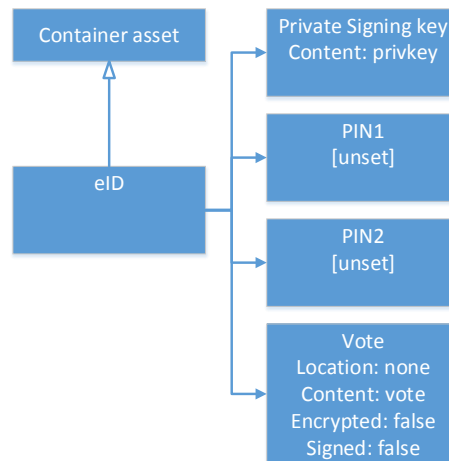


Figure 6.4: Conceptual visualization of the eID

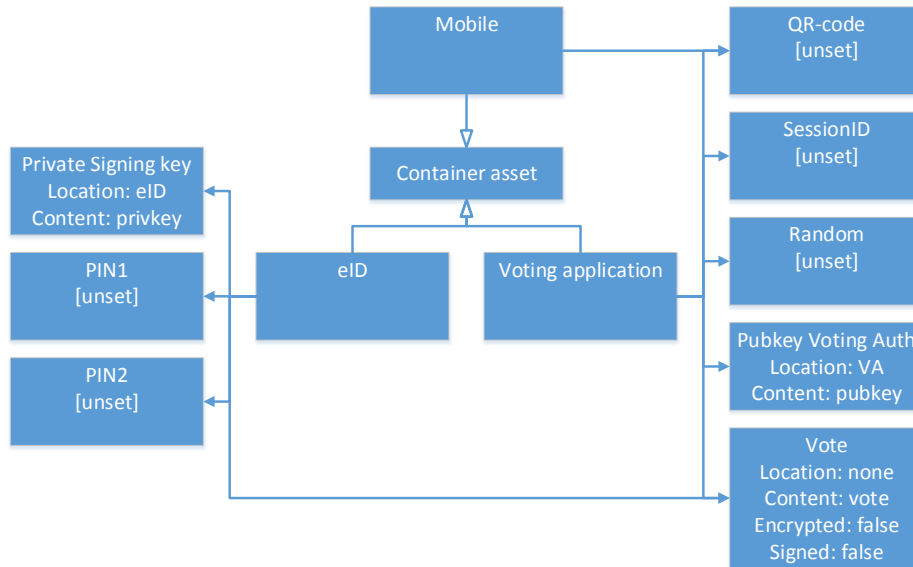


Figure 6.5: Conceptual visualization of the assets

COMPLETE CONCEPTUAL OVERVIEW Now that all assets have their properties being chosen, they will need to have an initial setting. As for most of them, they will not exist initially or have their content set to what they are (e.g. a key). The location of most of these assets is unset and will be changed throughout the workflow. A complete overview of the initial setting of the assets and their properties can be found in Figure 6.5. Note that when the location is not yet set, this means that that particular asset is not yet in a container asset.

6.1.2 Processes

The assets, their properties and the initial setting of properties have been identified. This subsection will show how to identify the processes that modify these assets. Again, this step of the framework starts with the simplified model depicted in Figure 6.2. A quick inspection of this model reveals the processes *input vote into voting application*, *encrypt vote*, *send vote to eID*, *sign vote* and *send vote to vote forwarding server*. Because the focus of this case study is a bit broader, the *requesting vote application* and *send voting app* precede these steps.

The specification of the eID reveals that it works with two PIN-codes, which are respectively entered to authenticate the holder of eID and signing a vote. Furthermore, the eID needs to be put into the card reader to function and the eID itself can't send the vote to the vote forwarding server, that has to be done by the voting application. This introduces four additional steps: *place eID on card reader*, *enter PIN₁*, *enter PIN₂* and *send vote to voting application*.

As presented in section 4.1, the Estonian voting scheme uses QR-codes for verification. For the proper function of this verification

Processes

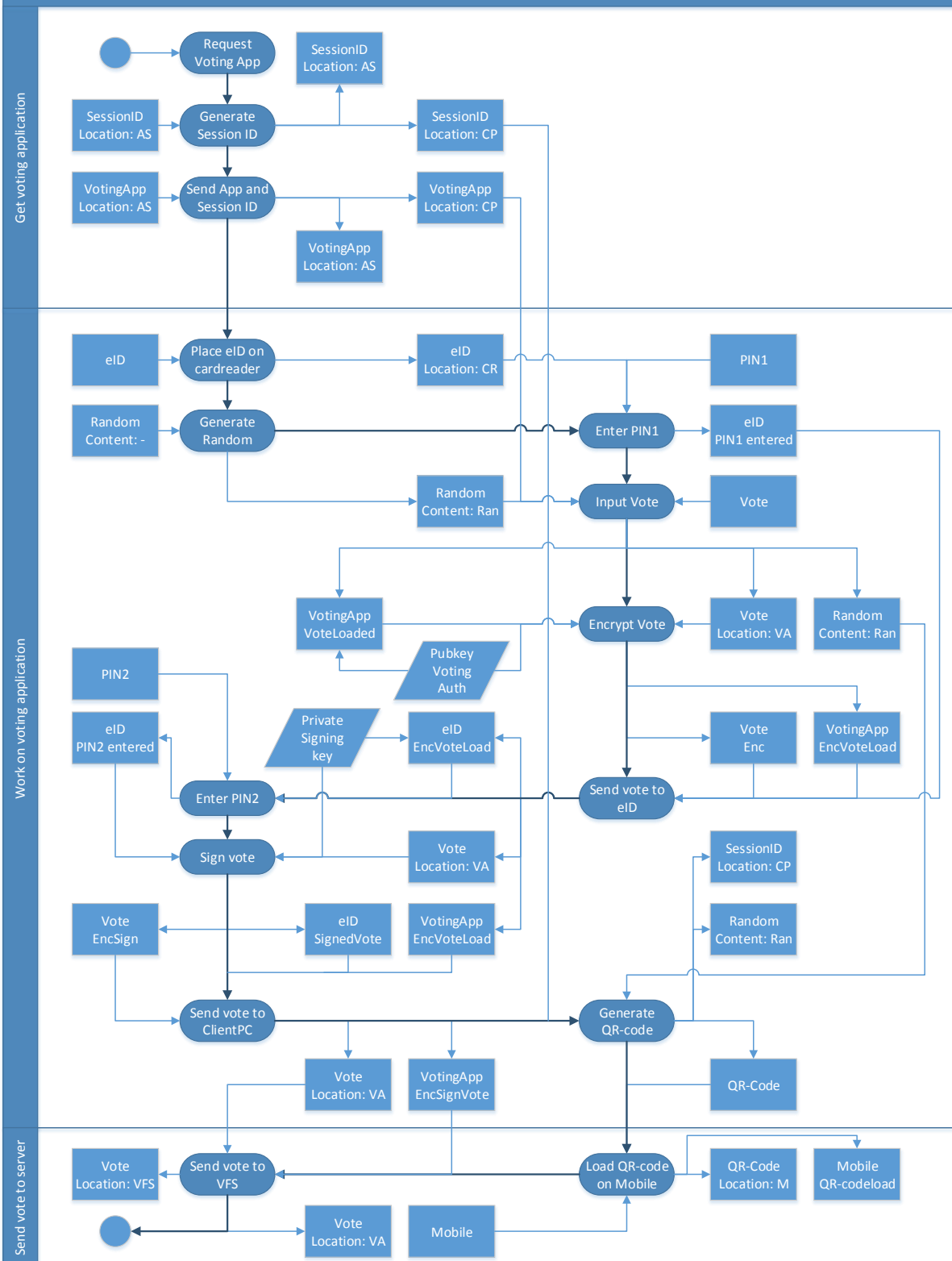
method, the server needs to *generate a session ID*, send it along with the voting application, the voting application needs to *generate a random, generate a QR-code* and *load the QR-code on the mobile phone*.

6.1.3 *Workflow*

Workflow

Now that these processes have been identified, the corresponding input and output assets and their changing properties can easily be defined. The visual representation of all these processes is depicted on the next page. Note that the rectangles with the rounded corners (and slightly darker background) are the processes, while the other rectangles are assets. Because of clarity, only the changing properties are shown. The parallelograms are keys that are extracted from their container asset. Every processes guides towards the new process with the slightly darker arrow, defining a workflow from the starting point (the circle in the top left corner) to the end (the circle in the bottom left corner).

Estonian voting system



6.1.4 Attack flow

Attack flow

Attack vectors can both be found using the model checker - which can find a lot of them - or by analyzing the constructed workflow manually. In this section, an attack vector found by hand will be shown. The goal of this attack vector is to change a vote the user wants to cast. There could be different starting points for this attack vector, but for now the focus is on the *Input Vote*-process. This process will be swapped for a fake process which accepts another vote as input. The fake vote will then be used in the *Encrypt Vote*-process, and so on, until it reaches the end of the workflow, being duplicated and having the locations *Vote Forwarding Server* and *Voting Application*. When a voter would now check whether his vote was stored by the Vote Forwarding Server correctly, using the QR-code, he would find out that there was something wrong. To cope with this, an additional attack vector is necessary, namely a faked QR-code. This can be achieved by swapping the *Generate QR-code*-process with a fake one that accepts another *SessionID* and another *Random* to allow for a QR-code that would apply to the faked vote. On the next page, the same workflow can be found with the attack vectors present. The red processes are those that were swapped with a fake process. Both of these faked processes included the acceptance of fake assets, which are depicted in this workflow in red as well. The input asset that is gone has a red input arrow to show that it disappeared, so does the new input arrow of the fake asset. Fake assets stay depicted in red from the moment they were faked. This eventually leads to a fake vote in the vote forwarding server, and a QR-code that displays a correct recording of the vote at the vote forwarding server.

How to actually get to these attack vectors, and how to make sure the QR-code correctly outputs the initial vote of the voter is part of the next step in the framework.

Note that an attack that would abuse these two attack vectors would achieve the goal that was formulated for attack 1 in this chapter. Note furthermore that when the vote was inserted into the voting application, it could be copied by a malicious process to allow for attack 2 of this chapter, as depicted in Figure 6.6. The original output of *Input Vote* is discarded and the malicious process *Steal vote* will now output the expected *Vote* with location *Voting Application*, meanwhile, it stole the vote by duplicating it and setting its location to *Attacker*.

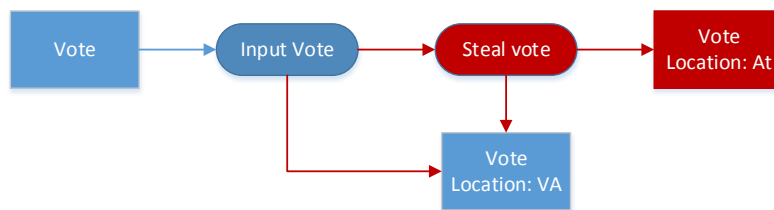
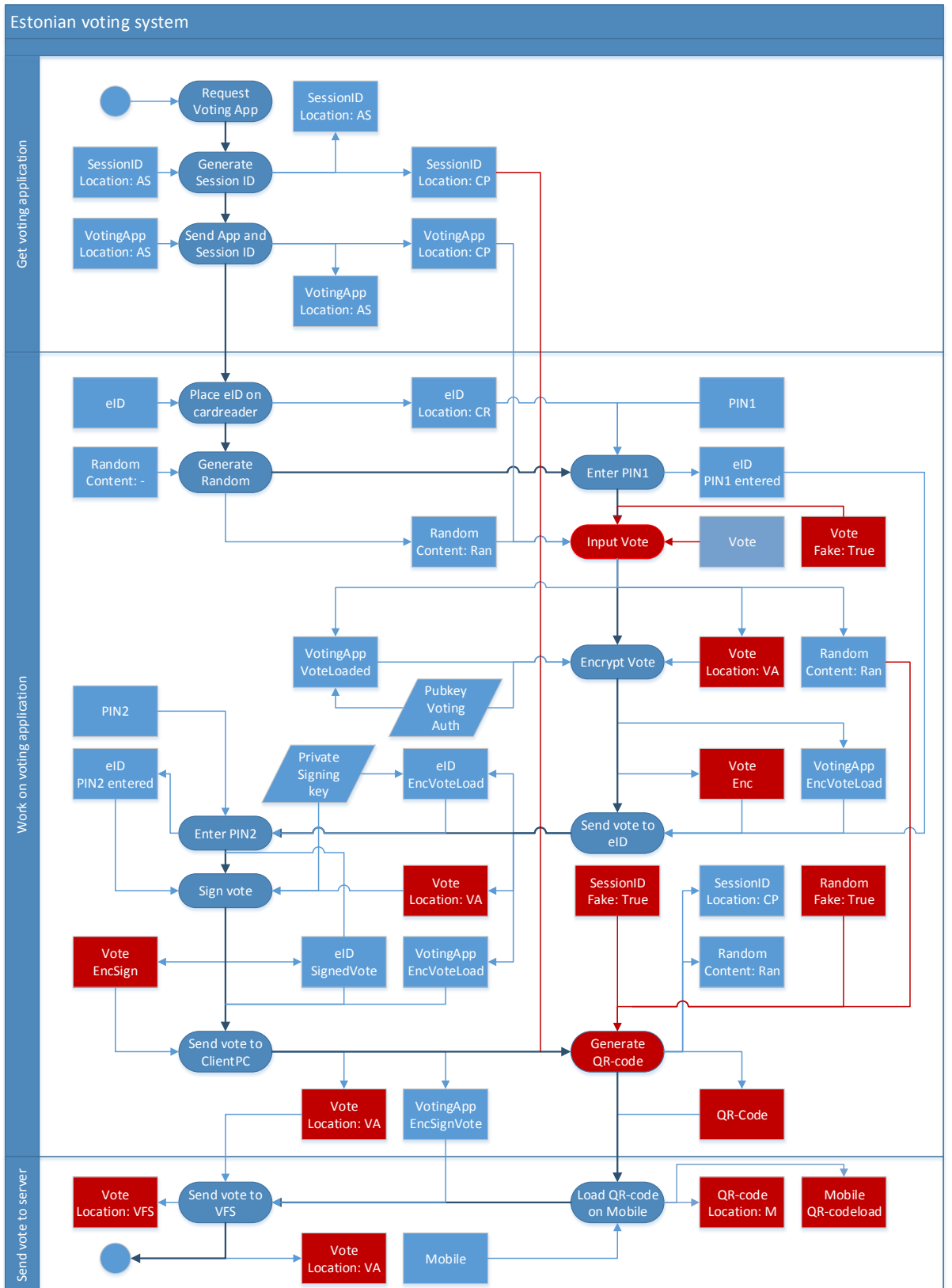


Figure 6.6: Attacker flow of confidentiality attack



6.2 QUANTIFY

Quantify

The two attack vectors that were found using the previous step in the framework, will now be used in this step. Both of the attacks will be quantified using the substeps of the framework. First the attack tree corresponding to the attacks will be built, then the number of days necessary for the attack will be calculated, where after the number of PPIs to be bought is determined, and finally the costs will be calculated. The steps can be found in Figure 6.7.

The structure of this section is as follows, first the attack vectors on vote integrity will be quantified in section 6.2.1 and then the same process will be conducted for the attack vector concerning the vote confidentiality in section 6.2.2.



Figure 6.7: The framework, with the detailed steps for Quantify

6.2.1 Attack on vote integrity

Vote integrity attack

The attack vectors that were found in the previous step belonging to this attack are: replacing a vote with a malicious vote; replacing sessionID and random with malicious ones to fake a correct verification. In the following substeps the attack will be built.

Note that a similar attack was developed by a student in Estonia as a proof-of-concept, see section 4.1.4, another example of a similar attack can be found in [37].

6.2.1.1 Attack tree

Attack tree

To be able to swap two processes from the workflow, the location of these processes must be examined. Both of them (*input vote* and *generate QR-code*) happen within the voting application run on the client computer. To be able to swap them, malware that injects the voting application could do the job. Therefore, an attack tree as blueprinted in section 5.3.2.1 will be used. To have a clear attacker goal, the goal will

be set to gain exactly one seat in parliament. This can later be changed due to the split between variable and fixed costs, but it helps in setting up the attack. The attack tree itself can be found in Figure 6.8.

To generate a valid QR-code for the faked vote. The C&C-server lets the first vote that it wants to fake through, and collects the sessionID and random. Those are then distributed to all voting applications that will change the vote to the attackers preferred one. This will allow them to verify their vote, by having their mobile phone read the same QR-code as the first vote that was let through.

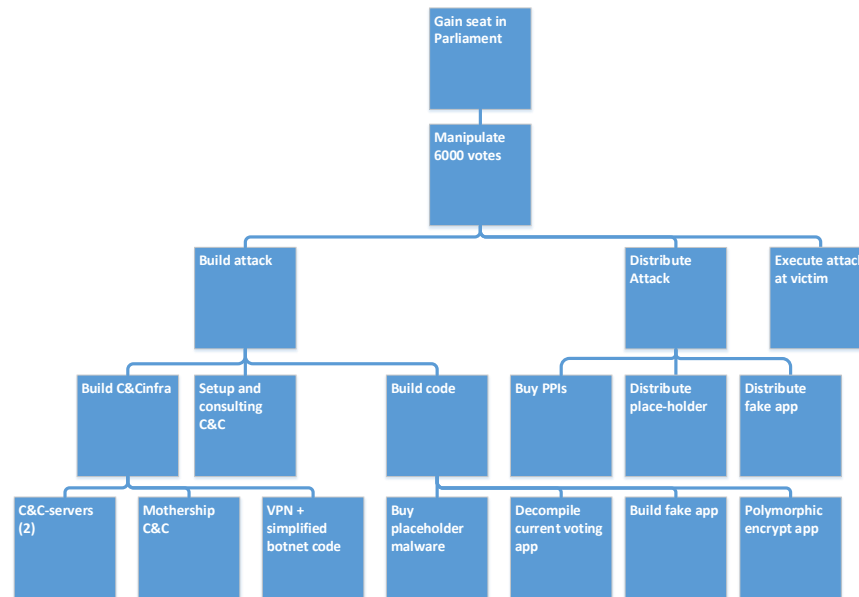


Figure 6.8: The attack tree of the integrity attack

C&C-infrastructure

The C&C-infrastructure that will be used for this attack, consists of the following:

- Bulletproof hosting for the C&C-servers (2): one server for maximum 3,500 active voters;
- Bulletproof hosting with DDoS-protection for the back-end server;
- An anonymous VPN to connect to the back-end server;
- Simplified botnet code;
- Consultation for the botnet code.

Placeholder malware

The placeholder malware is built using a RAT, with some custom refinements:

- Placeholder malware (RAT);
- Custom refinement of the placeholder malware code.

The actual malware is built by first decompiling the current voting application and then building a fake application. This one is encrypted 10 times using a polymorphic crypter to fool regular malware detection mechanisms:

- Decompile current voting app;
- Build fake voting app;
- Join botnet code and the fake application;
- Polymorphically encrypt it to get only 1,000 computers infected with one sample.

A simplified version of this attack is depicted in Figure 6.9. In red, the commands from the command&control server are shown. They change the vote in the fake voting application and the QR-code that is sent to the mobile phone for verification is changed to that of a real vote.

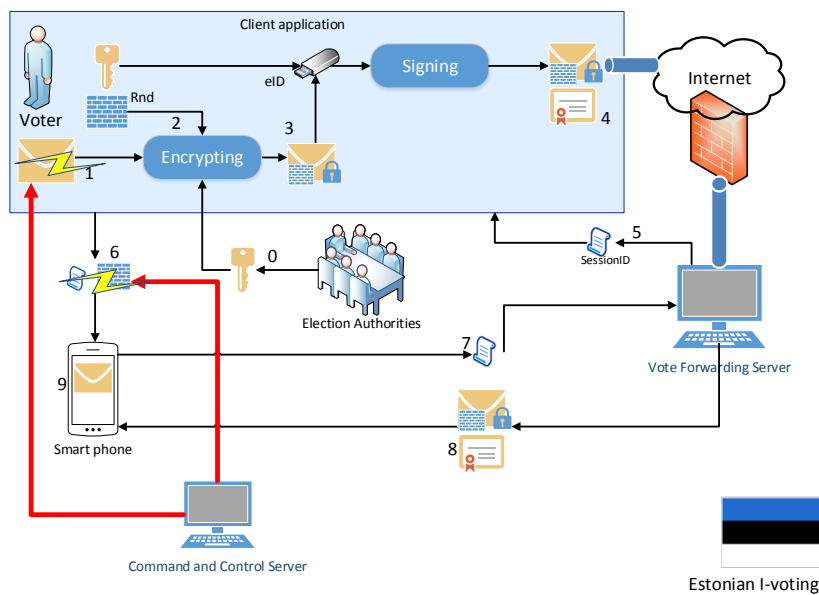


Figure 6.9: Attack on the Estonian e-voting scheme (simplified)

To avoid suspicion at the verification back-end caused by having thousands of people verifying the vote of one voter, one could choose to change only a portion of the votes, and distribute the verification randoms and sessionIDs of the other voters let through to the voters of which the vote was changed.

6.2.1.2 Days

Days

The different steps in the attack tree can now be decorated using the attacker model presented in section 5.3.2.2. This will lead to the attack tree displayed in Figure 6.10.

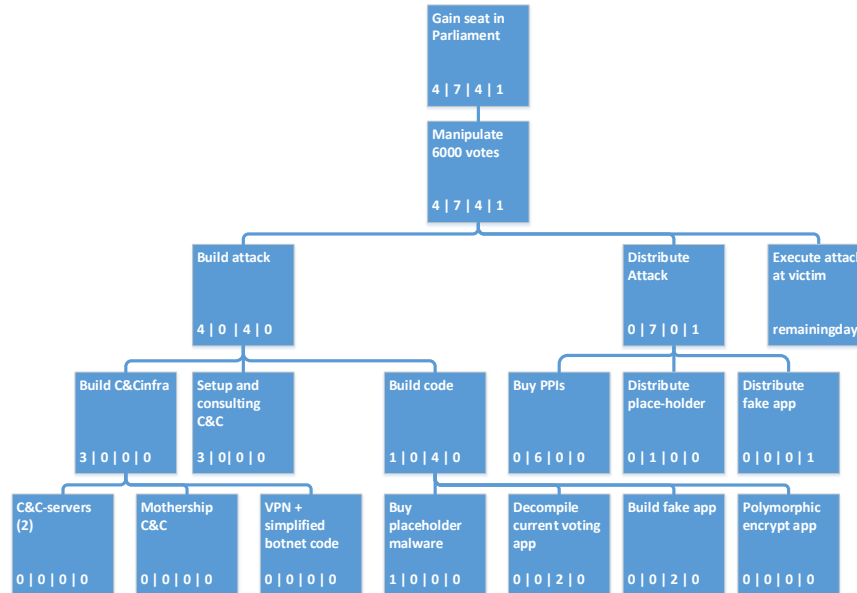


Figure 6.10: Days decorated attack tree for the integrity attack

The time investments are split in active and passive voting days, and in advanced and actual voting days in following order: active advanced | passive advanced | active actual | passive actual.

Now that these time investments are known, a time frame can be constructed that shows when certain time investments are made, and in what order. The constructed time frame can be found in Figure 6.11.

6.2.1.3 PPIs

Pay-per-install services

To make a difference of at least one seat in parliament, a hacker needs to change about 6,000 votes on average (and the party should at least have 5% of the votes in total, but that is considered to be the case already in this example)³. To estimate how many PPI's should be bought to get to 6,000 computers used for e-voting, the total number of desktop computers in Estonia will be roughly estimated, as well as the number of computers that is used for e-voting.

Total number of computers

Within Estonian companies, 47.5% of the people in active labor use a computer. This means that, with 621,300 people in active labor,

³ http://en.wikipedia.org/wiki/Estonian_parliamentary_election,_2011

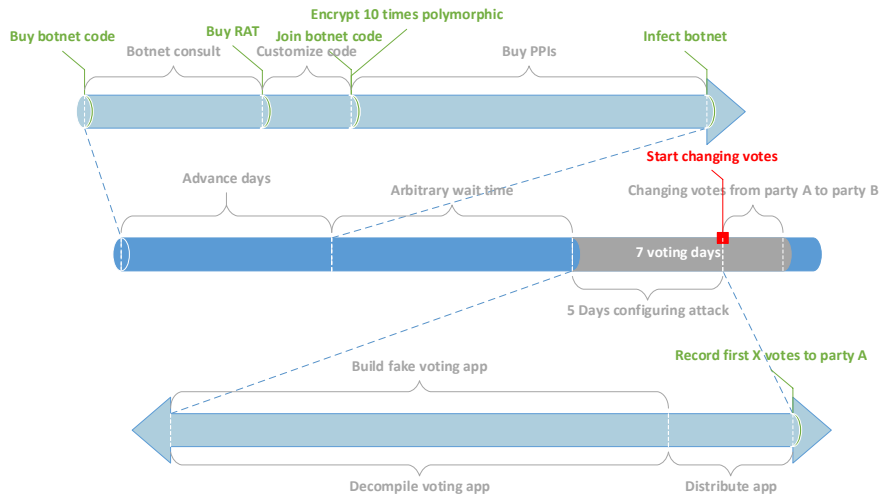


Figure 6.11: Timeframe for the integrity attack

295,118 computers are found in companies. However, the statistics are missing on the total number of computers per capita or per household, the number of internet connections and the number of mobile subscriptions compares relatively well with neighboring country Finland ⁴, which has about 1,6 computers per household ⁵. This suggests that, with about 600,000 households, the number of computers in households is estimated at about 960,000. This brings the total of computers in Estonia to a very rough estimate of 1,255 million.

Average e-voters per computer

Of the total of eligible voters, about 15% voted using e-voting for the national election of 2011, totaling to 163,035 voters ⁶, of which this research estimates 80% voted from home and 20% from the office:

- Home e-voters represent $163,035 * 0.80 = 130,428$ citizens using one the 960,000 computers in household;
- Office e-voters represent $163,035 * 0.20 = 32,607$ citizens using one the 490,000 computers in companies;

This means that on average,

$$\frac{130,428}{960,000} * \frac{960,000}{1,255,000} * 100\% + \frac{32,607}{490,000} * \frac{295,118}{1,255,000} * 100\% = 12,99\%$$

of all computers in Estonia are used for e-voting.

This means that, to reach the 6,000 votes needed for one seat difference in parliament, 46,189 Estonian computers should be bought. To

⁴ For example, the Worldbank on internet users <http://data.worldbank.org/indicator/IT.NET.USER.P2>

⁵ <http://www.generatorresearch.com/tekcarta/databank/personal-computers-per-household/>

⁶ <http://www.vvk.ee/voting-methods-in-estonia/engindex/statistics>

be relatively sure, and to deal with the fact that the spread of votes per computers is somewhat large (if one of the household members votes using e-voting, chances are bigger that some additional household member will also e-vote (average: 15% e-voters), a 20% margin will be used to total the number of PPI-computers to 56,000.

Due to the fact that the advance voting period for e-voters is only seven days, and decompiling the voting application, building a fake one and distributing it takes about five, only the voters of the last two voting days are affected. In 2011, of the 140,000 total e-voters only 51,000 voted in the last two days. This means that only 36,43% of the e-voters vote in the last two days. Therefore, the attacker needs $\frac{56,000}{36,43\%} * 100\% = 153,719$ computers to be infected, as depicted in Figures 6.12 and 6.13. A 10% margin makes this 170,000 computers.

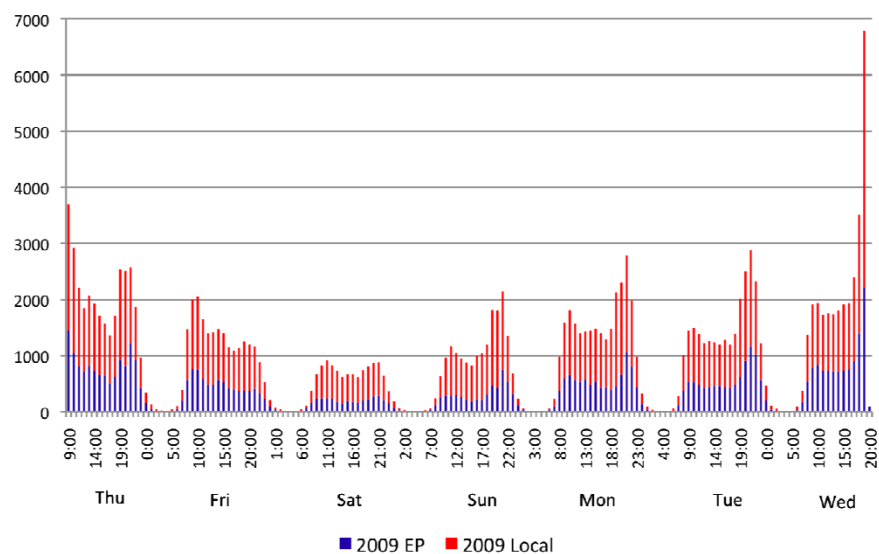


Figure 6.12: Estonian votes received per hour [104, Figure 5 on p30]

Note that there is both a limited number of people that actually e-vote and a limited number of Estonian computers that can be bought using a PPI-service. Therefore, there is an imaginable limit to the number of seats in parliament that can be reached using this attack.

6.2.1.4 Costs

Costs

The attack will be carried out with the following components and services bought in the underground, considering a moderately good programmer who did not built any malware before and needs some consults on how to setup his botnet to stay undetected. The list of costs can be found in Table 6.1.

A completely decorated attack tree can be found in Figure 6.14.

Since none of the variable cost steps in the attack tree influence the number of voting days left for the attack, the cost function is

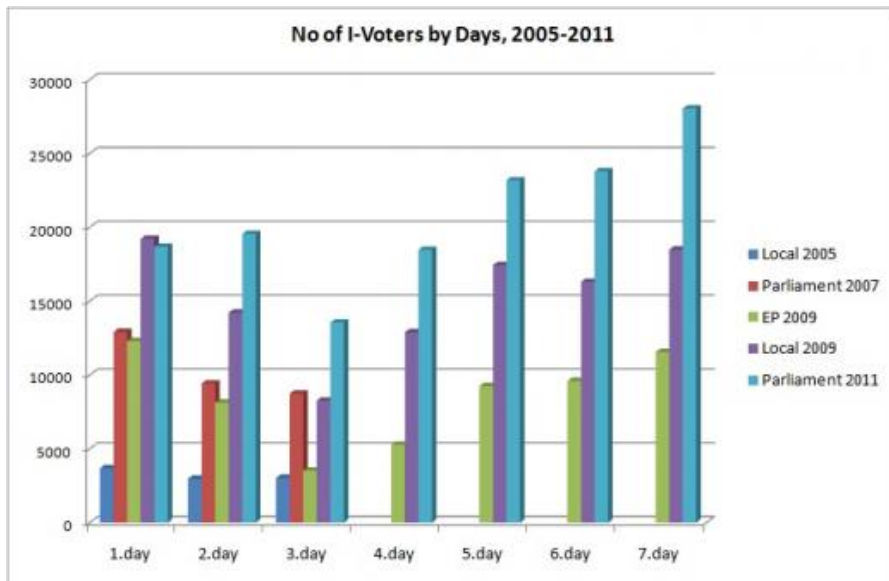


Figure 6.13: Estonian votes received per day according to the Estonian Voting Authorities: <http://www.vvk.ee/voting-methods-in-estonia/engindex/statistics>

Table 6.1: Items from the underground economy needed for the integrity attack

Items	Variable / Fixed	Costs (US \$)
Bulletproof hosting for the C&C-servers (2)	Variable	250
Bulletproof hosting with DDoS-protection	Fixed	2,000
Anonymous VPN	Fixed	30
Simplified botnet code	Fixed	70
Consulting for botnet setup	Fixed	350
Buying placeholder malware (RAT)	Fixed	250
Join the botnet code and the malware	Fixed	30
Polymorphic crypter (170 times)	Variable	17,000
170,000 PPIs	Variable	17,000

constructed rather straightforward. As explained earlier (in the PPI substep), there is a clear dependency between the number of victims and the number of PPIs to be bought. In this case, it turns out that 170,000 PPIs need to be bought to ensure 6,000 actively attacked voters. For every 3,500 actively attacked voters, this attacks needs one

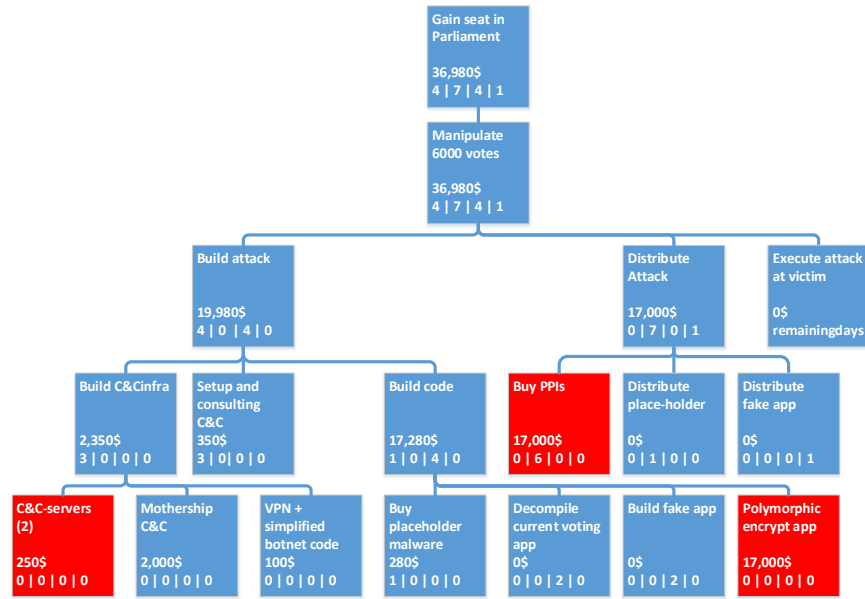


Figure 6.14: Completely decorated attack tree for the integrity attack

C&C-server. Furthermore, every 1,000 bought PPIs get their own polymorphically encrypted malware sample. All other costs are fixed. The following function is expressed in figure 6.15.

$$\begin{aligned}
 \text{Costs} &= 2,730 + 125 * \left\lceil \frac{\text{votes}}{3,500} \right\rceil + \frac{17,000}{6,000} * \text{votes} + \frac{17,000}{6,000} * \text{votes} \\
 &= 2,730 + 125 * \left\lceil \frac{\text{votes}}{3,500} \right\rceil + 5\frac{2}{3} * \text{votes}
 \end{aligned}$$

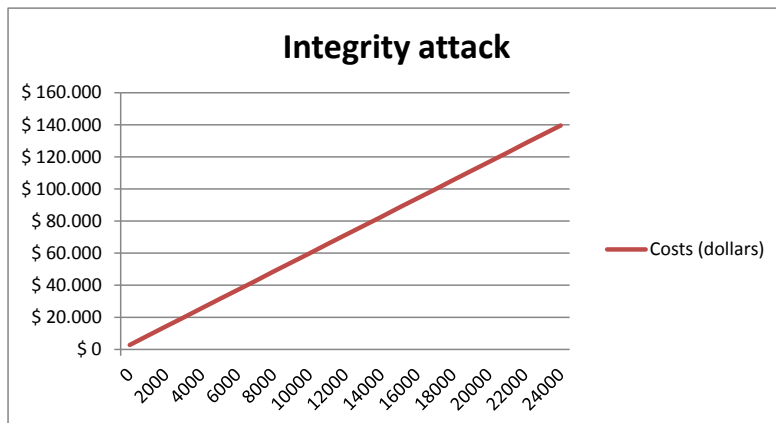


Figure 6.15: Effort as a function of the number of votes (x-axis) in an Estonian integrity hacking example. Every 6,000 votes means 1 seat in parliament.

6.2.1.5 Conclusion

As can be seen, the financial investments for such an attack are relatively low. The costs for setting up the attack add up to only 2,730\$, which can then be deployed for roughly 34,250\$ per seat in parliament. Furthermore, the time investments are relatively low and the preparation time can even be spread over numerous of days, since there is an arbitrary waiting time.

This attack is feasible, all be it that the Estonian government is closely guarding the internet connection of the Estonian citizens during the voting period. Furthermore, it is infeasible to buy more than a certain amount of seats in parliament using this method, due to the fact that not every Estonian computer can be bought in the underground, the fact that already 170,000 computers are needed for just one seat in parliament makes this attack not very scalable.

Conclusion

6.2.2 Attack on secret suffrage

One of the attack vectors that was found in the previous step belongs to an attack on vote confidentiality (secret suffrage). It copies a vote and sends it together with some identifier of the voter to the attacker. This attack uses the same point of entry as the previous attack, namely the *input vote* process.

For this attack, a rather alternative method has been established to get the attention of the media about this attack, with the attacker goal in mind that faith in the e-voting scheme is lost by the general public. The attacker does this by adding newly stolen votes every certain timeframe, like in a hostage situation where an attacker starts killing hostages every hour.

Secret suffrage
attack

6.2.2.1 Attack tree

The attack tree of this attack is exactly the same as the integrity attack: the goal of the attacker is rather similar, namely, to inject the voting application with some custom functions. In this case the C&C-structure is used to store all stolen votes instead of managing all sessionIDs and randoms, but again, this is a similar setup. The only additional step in this attack is to publish the names with their corresponding votes publicly, this is included in the execute step of the attack tree, depicted in Figure 6.16.

Attack tree

6.2.2.2 Days

As the attack tree of this attack is almost similar to that of previous attack, there is not much change in the number of assigned days. The difference lies in the fact that a vote changer is complexer than a vote stealer. It will be assumed that building the vote stealer will only take

Days

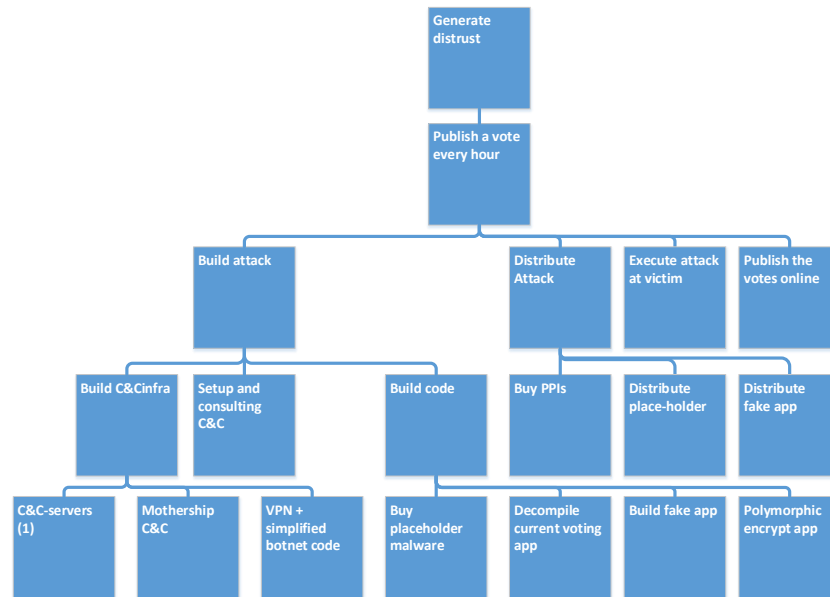


Figure 6.16: The attack tree of the confidentiality attack

1 day, instead of 2 for a vote changer. The attack tree decorated with days can be found in Figure 6.17.

6.2.2.3 PPIs

*Pay-per-install
services*

Depending on the impact the attacker wants to make, he needs to spread the placeholder malware to a set of computers. If for example, the attacker wants to add a new vote to the public website or service every hour, he needs to collect a certain base of votes from where he can select them, and needs to fill this base periodically.

Although the voting period in Estonia is relatively short (seven days), Figures 6.12 and 6.13 shows that the distribution of the votes is not uniformly distributed over time. To be sure to be able to publish a vote every hour, the attacker will need an estimated advantage of 24 votes, so that in the case none of his infected computers is used for e-voting in 24 hours, it still looks like he has the control over a lot of computers. Furthermore, the attacker can only start collecting votes from the moment he spreads the fake e-voting application. From that exact moment on, he must suspect his malware to be under the investigation of the Estonian CERT ⁷, because they can identify and examine the computer of that voter immediately.

Remember from the decorated attack tree with the number of days, that the attacker has two additional days during the voting period, compared to the integrity attack.

Attack information

⁷ Cyber emergency response team.

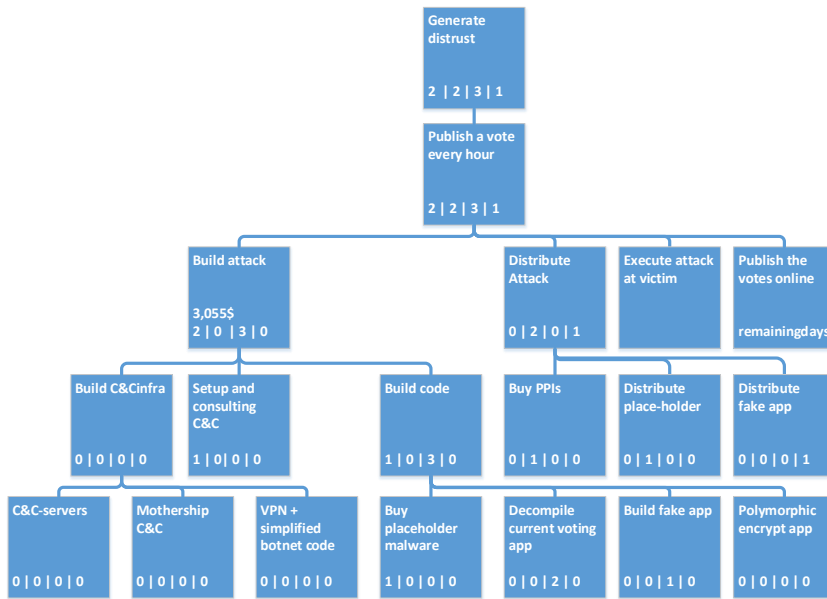


Figure 6.17: Days decorated attack tree for the confidentiality attack

The attacker needs to harvest 24 votes on the first day of launching the attack, and then wait one day before he starts publishing the votes. In total, the attacker already has enough captured votes if he can publish one vote every hour the remaining three days. This means that he needs 24 voters from day four, 24 voters from day five and 24 voters from day six. Since day four is the least voted-on day, randomly infecting computers of voters until 24 have been infected that vote on the fourth day, means that the attacker found at least 24 voters for day five and six. To infect 24 voters from the total e-voters list, he needs to infect $\frac{24}{\frac{18}{142}} \approx 190$ e-voters⁸. To infect one e-voter randomly, the calculations from the previous attack can be used. For that attack (which only targeted voters in the last two days, being 36,43% of the voters), $\frac{170,000}{6,000}$ Estonian computers needed to be infected. This means that, for this attack to work, $190 * \frac{170,000}{6,000} * \frac{36,43\%}{100\%} \approx 1,961$ computers need to be infected.

6.2.2.4 Costs

The exact same prices account for this attack as they did for the previous attack, with the difference being the number of PPIs to buy, the number of times the malware needs to be polymorphically encrypted

Costs

⁸ 18,000 e-voters of 142,000 e-voters voted on the fourth day in 2011 according to the voting authorities: <http://www.vvk.ee/voting-methods-in-estonia/engindex/statistics>

and the number of C&C-servers needed to handle all votes. The total decorated attack tree can be found in Figure 6.18.

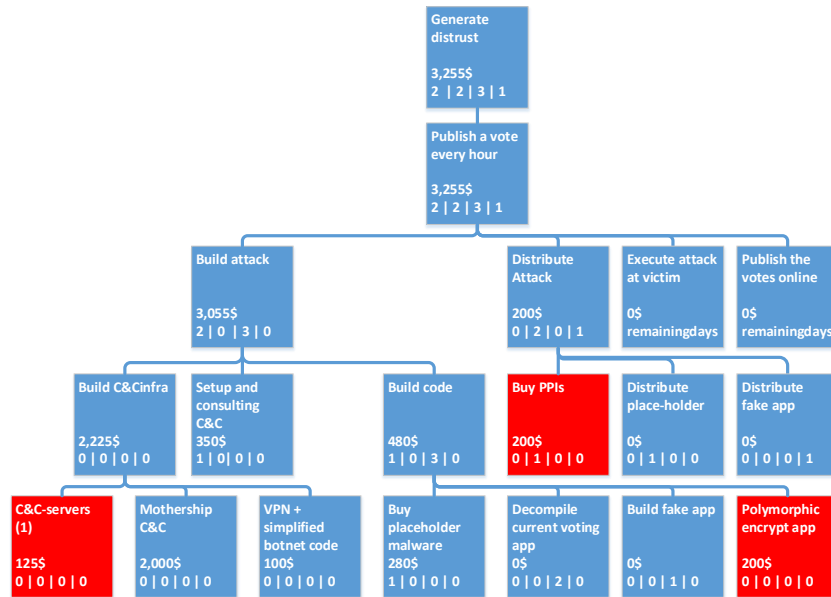


Figure 6.18: Completely decorated attack tree for the confidentiality attack

Since none of the variable cost steps in the attack tree influence the number of voting days left for the attack, the cost function is constructed rather straightforward. As explained earlier (in the PPI sub-step), there is a clear dependency between the number of victims and the number of PPIs to be bought. In this case, it turns out that 2,000 PPIs need to be bought to ensure a published vote every hour. For every 3,500 PPIs, this attack needs one C&C-server. Furthermore, every 1,000 bought PPIs get their own polymorphically encrypted malware sample. All other costs are fixed. The following function is expressed in figure 6.19, with votes being the number of votes published every hour.

$$\begin{aligned}
 \text{Costs} &= 2,730 + 125 * \left\lceil \frac{\text{votes} * 2,000}{3,500} \right\rceil + 200 * \text{votes} + 200 * \text{votes} \\
 &= 2,730 + 125 * \left\lceil \frac{\text{votes} * 2,000}{3,500} \right\rceil + 400 * \text{votes}
 \end{aligned}$$

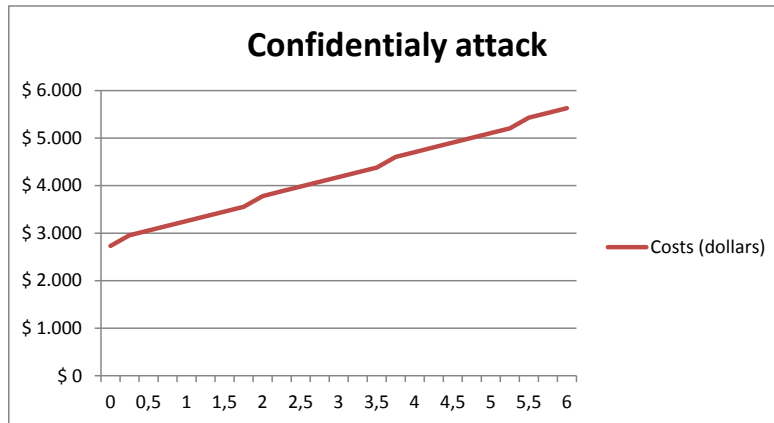


Figure 6.19: Effort as a function of the number of votes published per hour (x-axis) in an Estonian confidentiality hacking example.

6.2.2.5 Conclusion

As can be seen, the financial investments for such an attack are extremely low. The costs for setting up the attack add up to only 3,225\$, which can already be used to cause major disruption in the trust of the e-voting scheme. Furthermore, the time investments are relatively low and the preparation time can even be spread over numerous of days, since there is an arbitrary waiting time.

This attack is feasible, all be it that the Estonian government is closely guarding the internet connection of the Estonian citizens during the voting period.

Conclusion

6.3 CATEGORIZE, COMPARE AND MITIGATE

The remaining steps of the framework rely on a complete overview of all attack that can be found within an e-voting scheme. However, the two attacks that were addressed in this chapter do form an interesting case to zoom into some of the features of the framework.

6.3.1 Categorize

To be able to categorize these two attacks, the framework provides the researcher with two options. The researcher can choose to include the attack in all the safeguards that were violated with the attack, or the researcher could choose to base its category on the attacker goal. The latter case would of course categorize the attacks in the fairness and secret suffrage safeguards respectively. The former case would place both of these attacks in multiple categories based on Table 3.1.

The integrity attack brakes the confidentiality of (some) votes; the integrity of the results; the integrity of the votes; the integrity of the final vote; the integrity of the software; the integrity of the verifica-

Categorize

Integrity

Secrecy tion mechanism; the authenticity of the voting application; and the authenticity of the verification mechanism. This leads the integrity attack to break the safeguards verifiability, fairness, free suffrage, secret suffrage and equal suffrage.

The confidentiality attack breaks the confidentiality of votes; the confidentiality of the number of votes cast per votes; the confidentiality of the intermediate results; the integrity of the software; the authenticity of the voting application; and the non-repudiation of the vote cast. This leads the confidentiality attack to break the safeguards fairness, free suffrage and secret suffrage.

Depending on the choice of the policy makers, either of the two categorize methods can be used.

6.3.2 Compare

Compare

The choice of categorization method does influence the compare step. If the categorization method allowed attacks to reside in multiple safeguards categories, then it could be interesting how to compare different attacks among each other before being able to compare complete safeguards of an e-voting scheme with another e-voting scheme. Take for example the two attacks from this chapter. If both would be placed in the fairness safeguard category, as one of the categorization methods allows, then how to compare the following two cost functions. Which one is the cheapest attack and should be used for comparison? That leaves interesting questions for policy makers, as that would influence decision making a lot.

$$\text{Costs of integrity attack} = 2,730 + 125 * \lceil \frac{\text{votes}}{3,500} \rceil + 5\frac{2}{3} * \text{votes}$$

$$\text{Costs of secrecy attack} = 2,730 + 125 * \lceil \frac{\text{votesp/h} * 2,000}{3,500} \rceil + 400 * \text{votesp/h}$$

6.3.3 Mitigate

Mitigate

Interesting from this case study is the mitigation strategy that would try to mitigate the integrity attack. One of the factors that could be changed to try to mitigate this attack is the verification method, which could be enhanced with some more sophisticated protocols. An example could include the use of a mobile phone that gets a text message containing a code that corresponds to the voters individual polling card, like in the Norwegian voting scheme. Politics has to decide whether or not to include such verification methods, knowing that there are two very easy to spot downsides:

- That costs a lot of money, especially since the SMS-service must be very reliable and quick. If citizens are allowed to revoke, these costs could increase dramatically.

- The government must have a list containing all mobile phone number from their citizens. Privacy advocates will argue that the government shouldn't be keeping such a list. But besides, very practically issues will arise, like: what to do when someone orders a number change during election time? Also, the citizens should accept this method of verification, do they trust that the link between their vote and their mobile phone number doesn't give away their secret suffrage?

This shows that mitigation strategies can be very troublesome to actually implement in real life situations.

6.4 CONCLUSION AND REFLECTION

This chapter showed how the proposed framework can be used to find, quantify, categorize, compare and mitigate attacks within an e-voting scheme. Most emphases was put into quantifying attacks found using the graphically found attacks in the Estonian e-voting scheme.

After having applied the framework to the case of Estonia, it became apparent that actually using the technique provided by [Weldemariam and Villafiorita](#) needs the involvement of different stakeholders and experts concerned with the e-voting project. A simplistic setup as provided in section 6.1 can easily be constructed, though, finding the nitty gritty details involves a complete deep dive into the documentation and the consultation of the people involved. After that step, the graphically representation still need to be built and run in the model checker, which, additionally needs a rather large time investment. For the sake of this research, this step was not carried out.

The quantify-step in the framework was found to be relatively easy to use, after having found all the necessary statistics to approach the actual *days* and *costs*. The cost functions could be made with relative ease, and the results were quite frightening. Most of the attacks look-a-like, and can be therefore be constructed fast. It is expected that different attacks with the same attacker goal can share parts of the attack setup, and therefore benefit from the use of decorated attack trees.

The categorize, compare and mitigate steps were only touched upon but already signaled some important point. Depending on the choice of the policy makers, the categorize step will fall apart into a multi-category list of attacks or a list of single categories per attack. The compare step is heavily influenced by this categorization approach, as the different cost functions could have different input types. This could make comparing the attacks impossible or at least hard. The mitigation step of the framework was found to need a number of experts on the matter to both find mitigation strategies and to test

Conclusion and reflection

Find

Quantify

Categorize, compare and mitigate

whether they are acceptable for politics and would work in real life situations.

Conclusion

It can be concluded that the framework does provide for a relatively easy to use method of quantifying attacks, which does provide for tremendous amounts of information for policy makers. Furthermore it helps finding these attacks, categorizing, comparing and mitigating them, although some improvement points definitely exist.

Astonishing results

Besides the usage of this model, it may be noted that the current – very limited – case study already presented some interesting results. As it turned out, an attack on the integrity of Estonian e-voting results can be carried out with only 37.000\$, while an attack on the availability on the last day of e-voting (see chapter 5) costs a maximum of 8.000\$ (but can probably be carried out with less costs when using the advertised DDoS-capabilities in the underground). A simple attack to break the trust in the e-voting scheme (and break secret suffrage of a few voters) only costs about 3.500\$. These results show the importance of conducting such an investigation using this framework.

FRAMEWORK VERIFIED

To verify if the developed framework for this thesis actually solves the research question and reaches the research goal, expert interviews were conducted with several people and organizations concerned with the Dutch voting practice and the IT security industry. The expert interviews were not only held to verify if both the research goal and question would be solved, but also to verify whether all the requirements for the framework were matched. Furthermore, they were asked to determine the usefulness and usability of the framework. This chapter will first elaborate on the setup of the interviews and the people and organizations they were held with in section 7.1, after that the results of the interviews will be presented in section 7.2. The initial requirements of the framework will be discussed in section 7.3. Section 7.4 discusses the implications of these interviews and the requirements. These will again be summarized in chapter 8 Conclusion and chapter 9 Discussion and future work.

This Chapter will partly help answering research questions four: *“Apply the formalized safeguards to an e-voting scheme and verify the results and the framework.”*

7.1 INTERVIEW SETUP

Three professionals in the area of IT Security, the voting practice in the Netherlands and e-voting research in general were interviewed as part of the validation of the framework presented in chapter 5. Their expertise is visually depicted in figure 7.1. The IT domain means expertise in IT in general, the security domain means expertise in IT Security topics like governance, risk and compliance, the e-voting domain means expertise in specific e-voting topics like schemes and verifiability and privacy properties, the voting domain means expertise in actual voting practice currently deployed in countries.

The IT Security professional is very experienced in both the IT domain and the security domain; the e-voting researcher has a strong background in e-voting, but researches general security topics (including IT) as well, furthermore he has been involved in multiple e-voting projects that turned into actual voting standards; the Dutch voting professional has a background in IT but is mostly associated with the Dutch voting practice.

The author of this research first introduced the problem statement to the interviewees, than explained the goal of the research. From

Interview

Setup

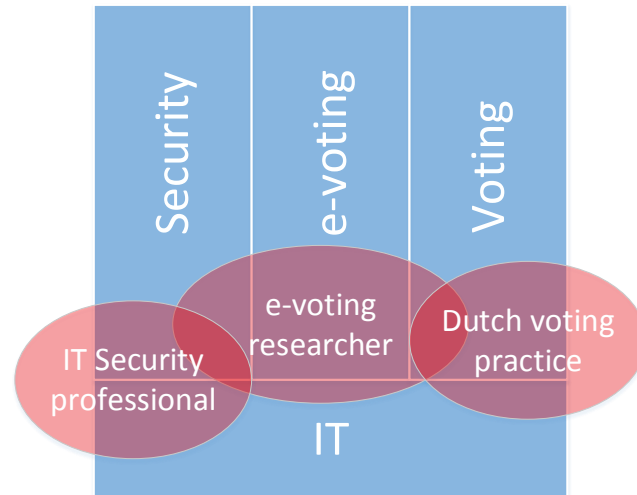


Figure 7.1: Venn-diagram explaining expertises covered by the experts.

there on, the framework was explained, and all the steps and substeps were showed with the help of both the case study and the framework itself. This was all done with the help of a slidedeck which included images and diagrams that can also be found in this research. In several occasions, the interviewer and interviewee elaborately discussed different topics and items that supported the research. Especially the different substeps led to interesting questions of how they actually worked.

At the end of the presentation of the slidedeck, there was time to discuss whether or not the framework would solve the initial problem statement or the research question. Furthermore, the usefulness and the usability of the framework were discussed at length with these different interviewees. Finally the interviewees were asked if the framework could be used in practice in the Netherlands right now and what any leftover improvement and strong points are.

7.2 INTERVIEW RESULTS

This section will present the summaries of the expert interviews, the more elaborate results can be found in appendix D.

The results are split into usefulness; and usability of the framework; whether we can start using it right now; and additional improvement points; and strong points.

USEFULNESS

- All experts agree on the usefulness of having a framework that identifies realistic risks in e-voting schemes. They all agree that such a methodological approach to address these was missing.

The explicit aim for e-voting schemes is named by one expert to be very important.

- One of the experts mentions that such an analysis will only work if the protocol, cryptography and implementation are investigated, as well as a penetration test.
- The usefulness of the model is, according to one expert, bound by the attack vectors found in the find-step. The expert is unsure if all properties can correctly be identified to not miss-out on any important attacker vectors, though, on the other hand, the expert believes that running the model checker may result in having too many attack vectors.
- According to one expert, the framework is useful for determining the template e-voting scheme as well as discussing implementation details thereof. The expert is not convinced politicians will actually use the objective information provided by this framework.

USABILITY

- Since the case study did not involve the use of the model checker, the amount of attack vectors to be sieved is unknown. According to two experts, this could be a problem to the usability. One expert does not agree, mentioning that the underlying processes in the voting practice do allow for an initial easy filtering.
- Two of the interviewees agree on the importance of constructing the graphical model of the voting scheme. Experts from different disciplines should be involved to ensure that all important information concerning both the digital and the physical aspects are translated into the model. One interviewee notes that it could be that some experts are not capable of understanding these high-level schemes, and therefore care should be taken when having such round table sessions. Another interviewee suggests to start involving the external party that will translate the graphical model into NuSMV source code should be involved early on in the process of using this framework.
- External experts should be involved in sieving and quantifying the attacks, to ensure that both the sieving and quantifying is done correctly, according to one expert.
- The usability is, according to two experts, influenced by the fact that politics may not be willing to use the results from such a framework. They rather base their decisions on non-rational arguments. The experts are convinced though, that the framework is useful when discussing implementation details after that decision.

CAN WE START USING IT RIGHT NOW?

- The lack of a complete case study makes that “playing around” with the results is not possible, this hinders adoption according to one expert.
- As said before, politics isn’t ready for such objective results yet. Rather the results are interesting for discussing implementation details at the Ministry of Interior after the decision to adopt e-voting.

IMPROVEMENT POINTS These are all individual improvement points from the different interviews.

- Time investment should not be a separate attacker effort, because a determined attacker does not lack time. Time investments do influence financial investments and should be taken into account, though, but not as a key attacker effort.
- In the current framework, cost functions can not easily be compared. Not only because the input and output variables differ, but also because it is unclear how to indicate the superior function if these functions cross.
- It is not very realistic to assume a game theoretic attacker that always takes the “cheapest” attacker path. E-voting schemes therefore may be valued much more insecure than in a realistic scenario.
- Spread of costs is not taken into account. With so much estimations, the cost function could benefit from ranges.
- The first two steps are more elaborate than the last three. Especially the last step could be more elaborate to support users of the framework.
- The framework does not really answer the research question (it is not the answer of operationalizing the safeguards), but it helps operationalizing them.
- The results of the quantification-step could deviate from the actual situation due to the use of estimations. Especially if thresholds are used (e.g. in the quantification or the comparing-step) this could lead to undesirable situations.
- Maintenance of this framework can be hard. It is unclear if the framework is resilient to working with currently uninvented schemes.
- The more the framework is used, the better it gets. Therefore a more thorough case study would have been beneficial for the framework.

STRONG POINTS These are all individual strong points from the different interviews, except for the first point.

- All experts mention the modular approach to be a very important point.
- Two experts mention the adaptive attacker model crucial to model real life risks.
- The quantify and categorize steps make for an objective comparison of attacks.
- The quantification of attacks in general could be interesting in other areas as well.
- The framework does not take stand in any political argument, but objectively shows what certain decisions cause.
- The framework does show what decisions are important to take into account when deciding on acquiring such an e-voting scheme. The framework points to those decisions and gives objective feedback to make rational choices.
- The translation of safeguards to types of attacks is very useful.
- The framework is very systematic, it can be understood by lots of different disciplines and it provides for a good starting point of taking such risks to e-voting schemes into account.

7.3 REQUIREMENT VERIFICATION

Requirements

To check whether the requirements for the framework from chapter 1 were met, table 7.1 sums the different requirements up. Some of these requirements are easy to verify being met, while others need some more thought.

Requirement 5: addressing the gains for the attacker in number of affected votes is only partially met. There are different attacker goals that can not be expressed in the number of affected votes, for example a DDoS-attack that is expressed in gigabytes is not expressed in number of affected votes. It may be noted that with hindsight, the requirement should have been different: some attacker goals may better be expressed in some other quantity than votes. However, this does make comparing different cost functions hard or impossible, since different input and output variables make for an inability to compare them.

Requirement 6: a clear and concise overview of different risks is met, although it is not straightforward. The expert interviews were used to test whether the framework was useful, thereby meeting this requirement.

Requirement 8: the results and mitigation strategy should be understandable by non-technical decision makers is met, but that also involved the expert interviews. They were asked whether we could start using the framework right now and if it is useful and usable. These three questions combined led to the conclusion that the policy makers could start using it right now. Roughly translated, this became a “met” on this requirement.

Table 7.1: Verification of the framework requirements

	Requirement	Met
1	The framework should allow for a quantitative comparison of the risks of different safeguards of an e-voting scheme	Met
2	The comparison should be based on a risk assessment for the safeguards proposed by The Election Process Advisory Commission	Met
3	The risk should be addressed according to the effort an attacker has to put into breaking this safeguard	Met
4	The effort of the attacker should be split into time and monetary investments for the attacker, based on actual dollars and days	Met
5	The gains for the attacker should be addressed in number of affected votes	Partly met
6	The risk assessment should give a concise and clear overview of the different risks associated with each safeguard	Met
7	The risk assessment should give some pointers for mitigation strategies based on the attacks	Met
8	The risk assessment results and mitigation strategies should be easily understandable by non-technical decision makers	Met

7.4 CONCLUSION

Conclusion

After having conducted the different interviews and having verified whether the framework meets all the requirements, the overall conclusion of the validation is positive. The framework is considered useful and usable by people from disciplines, involved in IT and (e-)voting. The lack of such a scheme was recognized, and the proposed framework may fill the gap. It may also be concluded that there are a few improvement points that do influence both the usefulness and the us-

ability, that lead to the conclusion that the current framework is still rather immature. Almost all requirements are fully met though.

The most important points from the expert interviews was the lack of a case study. This resulted not only in a lack of proper mitigation strategies, but also to a lack in insight into the find-step: it is unclear how many attack vectors will be found using this step and how to sieve them. To work with the framework, several experts noted that setting up the initial visual model of the scheme requires multiple people from multiple disciplines involved with e-voting to participate, as well as one or more external parties which can verify correct use and correct sieving in the find-step, and can monitor correct quantification in the quantify-step. The experts agree on the fact that politics may not be helped with this framework, due to the fact that they don't use rational but rather emotional argument when deciding to adopt such technology as e-voting. The framework can be of help when determining the implementation details. Comparing cost functions turned out to be harder than expected due to the nature of them. The modularity and the adaptive attacker model are very well received.

Reflection

The author of this thesis fully agrees with the experts on their feedback. Due to time constraints, the case study did not involve a complete run of the framework, which resulted in some important gaps in the usage. The author also agrees on how to use this framework in practice: involving multiple disciplines in round table sessions to ensure a proper visual model of the e-voting scheme, before letting an external party translate it into NuSMV source code. The implementation details can then be varied to determine the best setup.

Most other feedback about the framework can either be mitigated, or is of less importance according to the author. Section 9 Discussion and future work discusses these issues more elaborate.

CONCLUSION

This research was motivated *to improve the current voting debate in the Netherlands, by establishing a quantified framework for reviewing and objectively comparing internet voting schemes according to the safeguards presented by *The Election Process Advisory Commission* and in accordance with the state-of-art in computer science literature on voting schemes, such that the actors of the debate can objectively discuss the security implications of different internet voting scheme designs, in order to have a better informed debate on whether to start using internet voting in the Netherlands.*

Research goal

To be able to reach this goal, the main research question *how do we formalize the safeguards for voting schemes, presented by *The Election Process Advisory Commission*, to allow for a quantitative comparison of the risk of different e-voting schemes in the presence of an attacker?* needed to be answered. Initially, two knowledge questions had to be answered, namely how these safeguards compare to other voting safeguards and what the currently deployed e-voting schemes in other countries are. Consequently, a framework was designed to answer the design question, which was later applied and verified.

Research questions

8.1 SAFEGUARDS

The first knowledge question was answered using a literature review, and showed that the safeguards cover most of the laws and regulations that apply in the Netherlands. They also comply with recommendations from different bodies and working groups. These safeguards can also be recognized in safeguards that apply in other countries as well, while they also comply with most safeguards found in literature. However, the safeguards do hardly compare to safeguards from an IT Security perspective. A few very important points from IT-concentrated e-voting literature are not covered in these safeguards. While it could be argued that those are too detailed for such high-level safeguards, they are important not to miss out on whenever is decided to adopt e-voting technology within the democratic process in the Netherlands. In general, it was concluded that the to be developed framework can be based on these safeguards, as they do comply with most of the safeguards from law, regulations, literature and other countries.

8.2 SCHEMES

The second knowledge question was answered using a literature review and by building visual representations of these schemes. These schemes were reviewed to both get a view of important points the framework should be comparing and to identify numbers and figures for the basic attacker model. Because these schemes are the most fertile examples of what a Dutch implementation could look like, one of them (the Estonian) was used for the application of the framework.

8.3 FRAMEWORK

The design question of this research focused on how to actually formalize the safeguards into a framework that can compare e-voting schemes. This question was answered with the introduction of a framework that can objectively compare e-voting schemes based on the effort an attacker has to put into breaking safeguards. The effort of an attacker is addressed with time and financial investments and is based on both a research into the digital underground economy and on case studies showing (proof-of-concept) attacks.

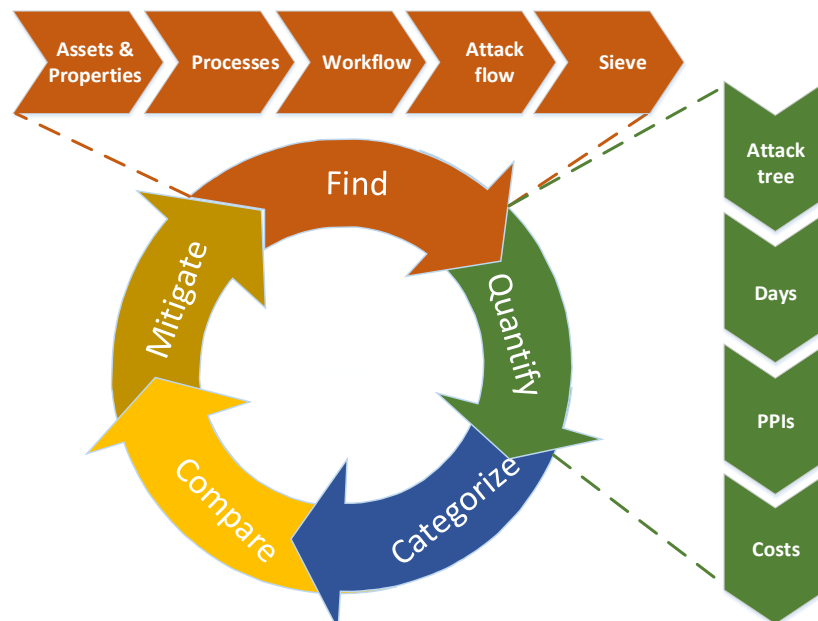


Figure 8.1: Complete framework including substeps

8.3.1 Find-step

The different steps of the framework are shown in Figure 8.1. The first step of the framework is to find all possible attacks of an e-voting scheme. First, the scheme is visually modeled using the assets and their properties, followed by the processes and then the complete workflow of the e-voting scheme, where after it is translated in the

NuSMV syntax, and run in a model checker to find all possible attack vectors. These attack vectors are then sieved on plausible attack scenario's before going to the next step of the framework.

Most of these attacks contain manipulation of some data on the client domain. Increasing the scale is usually done by buying pay-per-install services at the digital underground economy. These services offer the possibility to install malware on already infected computers, which were selected on some criteria (like country they reside in). This enables an attacker to get his custom-built malware to manipulate data on the client domain on large scale.

8.3.2 *Quantify-step*

The quantify step helps building attacks from the identified attack vectors using the concept of attack trees. Subsequently, the attack trees are decorated with time investments for an attacker, where after the number of pay-per-installs is determined. This allows for the decoration of the attack tree with the cost factors as well. The time and financial investments stem from adaptable tables that can be changed to allow for an other type of attacker (e.g. a more skilled attacker that can built malware quicker). This makes the framework work with an adaptive attacker model, namely, an attacker model inserted with a table containing time and financial investments for certain steps in the attack.

8.3.3 *Categorize-step*

All quantified attacks on the e-voting scheme are then categorized in the different safeguard categories using the categorize step of the framework. Two methods were presented to do so, one based on the information security properties that were violated, which are then mapped onto the safeguards, leaving most attacks to be categorized in multiple safeguard categories; and one that does a best effort to place an attack in one of the safeguard categories based on the final attacker goal.

8.3.4 *Compare-step*

The compare step of the framework can be used in two ways. It can either provide a comparison of the results of the previous step to other e-voting schemes which were run through the first three steps of the framework; or the result can be compared to a baseline that was developed by e.g. politicians who e.g. state that it should be impossible for an attacker to break the integrity of the results for a certain amount of voters with a certain amount of effort. This step helps identify how

the e-voting scheme compares to other schemes or to the baseline by simply comparing them per safeguard.

8.3.5 *Mitigation-step*

The final step of the framework offers an insight into mitigation strategies for the attacks that were found. Together with the categorized list of attacks, their quantified attack tree and the plans for the schemes they were based on, the weak points of a scheme can be identified. These weak points stand out even more, when the previous step involved a comparison with another scheme, which either didn't suffer from a particular attack, or did suffer from such an attacker but contained much more attacker effort. The mitigation strategy in the latter case can be found among the implementation details of that scheme. Most of the mitigation strategies that are identified with this step in the framework need a thorough analysis on both the legal framework bothered with elections; and an analysis of the political willingness to use this mitigation strategy. Both analyses are necessary due to the impact decisions on the design of the e-voting scheme have.

After having revised the e-voting scheme with the mitigation strategies, the same framework can be applied to identify new attacks, or to re-quantify existing ones. This circularity in the framework allows for a continues improvement process of e-voting schemes.

8.4 VERIFICATION

The fourth question of this research was aimed at verifying the framework by applying it to a real-life case and by doing expert interviews with people concerned with voting, e-voting and IT security. The application of the framework showed how to apply the different steps in a real case, and identified some improvement points for the framework. Results of the limited case study include some fascinating results, e.g. it was found that "buying" a seat in the Estonian parliament would only cost about 40.000\$.

Case study

The find-step of the framework does need the involvement of multiple stakeholders with different disciplines to get to a complete overview of the e-voting scheme. The quantification-step of the framework works relatively easy, and outputs very specific cost functions. These cost functions are used for the categorization and comparison-steps, where they are found hard to compare because there can be different input and output variables for these functions. The mitigation step of the framework needs active involvement of policy makers to get positive results. Some of these improvement points will also be discussed in chapter 9 Discussion and Future Work.

Expert interviews

The experts were interviewed and asked for their view on the usability and the usefulness of the framework, as well as on whether

this could already be applied in the Netherlands. In their view, a framework that allows objective comparison of real life risks in e-voting schemes is really needed, especially one that has an adaptive attacker model. The current framework, in their view, allows for such a comparison. On the other hand, they consider the current state of the framework as rather premature, especially since this is the first time a framework like this was developed. According to the experts, a framework based on the effort of the attacker is a valuable framework for the current debate on e-voting but they recognize the amount of work it will take researchers to apply this framework in real life situations. In some of the interviewees view, especially the first step of the framework could contribute to a lot of work, while others think it can be done in a reasonable amount of time. However, they all do recognize that quantifying each of the found attacks takes minimal effort and contributes to valuable insights into the scheme. One of their most valued features of the framework is the fact that every step in the framework can individually be changed, e.g. to change the categorize step for it to be used in a different country, or by changing the find step with another method of finding attacks or to allow the framework to be used on different types of voting. The overall usefulness of the framework is mostly seen as testing the implementation details for risks, while they do not feel politics will use the results when deciding on adopting e-voting.

8.5 CONCLUSION

To conclude, this research was conducted to aid the current debate on e-voting in the Netherlands with a framework that can objectively compare different e-voting schemes based on the safeguards proposed by [The Election Process Advisory Commission](#). The framework that was developed for this research reaches this goal by quantifying attacks on e-voting schemes using the effort an attacker has to put into breaking these safeguards. Schemes can then objectively be compared on these attacker efforts.

According to a case study and expert interviews, this first attempt to such a framework reaches this goal. Future research should reinforce the current framework to reach the maturity level with which it can be used by policy makers in the Netherlands or elsewhere.

Besides the usage of this model, it may be noted that the current – very limited – case study already presented some interesting results. As it turned out, an attack on the integrity of Estonian e-voting results can be carried out with only 40.000\$, while an attack on the availability on the last day of e-voting costs a maximum of 8.000\$ (but can probably be carried out with less costs when using the advertised DDoS-capabilities in the underground). A simple attack to break the trust in the e-voting scheme (and break secret suffrage of a few vot-

ers) only costs about 3,500\$. These results show the importance of conducting such an investigation using this framework.

DISCUSSION AND FUTURE WORK

This chapter will address issues up for discussion and will present some future work. Most of the items that are listed in this chapter originated with the author of this thesis, however some were identified during the expert interviews.

9.1 DISCUSSION

Six different discussion points about this research were identified and are presented in the following subsections.

9.1.1 *Efficiency of the find step*

Important details on how the find step should be applied to a complete e-voting scheme were excluded from the case study, leaving that for the reader to find out. Few of these steps are not straightforward, such as the proper modeling of the properties of assets and how to sieve all attacks that are taken onto the second step of the framework. Furthermore, it is not clear whether there actually is an efficient way to determine all these attacks, it is e.g. unclear whether sieving the found attack vectors can be done by hand.

9.1.2 *Comparing cost functions*

As was noted during both the case study as well as during the interviews, there are situations for which it is impossible to compare cost functions during the compare-step of the framework. Take for example an attack on the availability of the system, two different attacks could have a different input to the cost functions: the number of affected victims (through malware prohibiting them from voting); versus the strength of a DDoS attack (through a bought DDoS-attack in the underground).

The compare-step of the framework can therefore not always cope with these cost functions. A quick-fix could be to compare cost functions with similar input, and still group the lowest cost functions per category. This would still allow for comparison of e-voting schemes, and could benefit from different types of baselines setup by politicians.

9.1.3 *Limited case study*

The application of the framework on the case study of Estonia was rather limited. The first step in the framework involved the visual modeling of the case on a limited part of the actual voting process and did not include the translation of this visual model into NuSMV source code. This influenced the later steps of the framework, because only two attacks were identified to show how the following steps worked. This allowed a fair presentation of the quantification step, but all other steps suffered from this limited case study.

9.1.4 *Steps in premature state*

A few steps in the process are in a more premature state than others. The find and quantify steps are elaborated upon, but similar elaboration is missing with the categorize, compare and mitigate steps. Since the categorize and compare steps are rather straightforward, they might not have suffered from the limited case study. That's another story for the mitigate step, which should help policymakers to make better informed decisions. A more elaborate work on this step would definitely aid the reader, especially since the case study hardly pays attention to this step either.

On a positive note, different knowledge questions about implementation details (such as: *how does an extended voting period influences risks associated with the e-voting scheme*) can still easily be answered using this step of the framework. The interviewees value this part of the mitigation step greatly.

9.1.5 *Time investments*

The framework uses both time and financial investments as effort for an attacker. However, when taking a closer look at the different steps in the framework, it appears that there is a dependency between the financial investments and the time investments. The actual output of the quantify step mentions time investments, but does not really rely on those for any comparison methods in the compare step. Furthermore, intuitively it is the case that attackers that want to attack an e-voting scheme are not lacking in time, especially during the advanced days. Their resources are more limited on finances than on time. The focus should therefore be on financial investments aided by the time investments on the actual voting days (since those influence the financial investment).

9.1.6 *Estimates and statistics*

The quantify step contained a few estimations, and presumes the use of some statistics. In the case study, it became apparent that a lot of these figures and numbers are rather out-of-date. Furthermore, these statistics (i.e. the estimated number of computers in a country) highly influence the costs of an attack, while these statistics change a lot over time.

9.1.7 *Conclusion*

While the aforementioned discussion points do influence a possible adoption of this framework, it may be noted that most of them are not fundamental weak points of the framework, but only stress the limited maturity level of the current framework. With little effort and a proper case study, most of these can be fixed to allow for a model that could be used in the Netherlands.

9.2 FUTURE WORK

This research leads to some future work, which is presented in the following subsections.

9.2.1 *Case study*

As was identified already, the case study that was currently performed is rather limited. The framework would benefit from having some hands-on practice with the find-step, but also the categorize, compare and mitigate steps would evolve from having performed a complete case analysis.

9.2.2 *Advances on intermediate and server side attacks*

In the current framework, modeling intermediate or server side attacks is hardly covered. Future research could focus on those attacks as well. Chapter 5 does show an example of a DDoS-attack, which can be modeled by the framework, but to actively involve such attacks, adjustments and additions to the framework are necessary.

9.2.3 *Bribes and coercion*

In the current framework, there is little material on bribes and coercion. Future research could focus on that to strengthen these types of attacks in schemes as well. Furthermore, bribes or coercion of more than one member of the IT staff, election authority or auditors should

be included in such a future framework as well. An interesting view could include the ideas from Buldas et al. [7], which would add the chance of discovery of different steps in the attack in the framework as well.

9.2.4 *Determining baseline*

There currently is no method to determine the baseline. If the framework would have been used more often, ideas on how to construct baselines would help politicians to read and understand the results of the categorization step.

9.2.5 *Advancements categorize and mitigate steps*

The mitigation step does need some additional research to identify the most important categories of mitigation steps. This would identify some initial pointers to calculate what the impact of certain policy decisions would be, before a first blueprint of the e-voting scheme is built.

INTERVIEWS

This Appendix contains the interviews that the author of this thesis conducted to identify the key players and their opinions on and requirements for internet voting. To guarantee the objectivity of the interviews, the interviewer did not take a stand in the conversation either in favor or against internet voting. Furthermore, the results, as presented in the following Sections, of the interviews were verified by all the interviewed afterwards to ensure the correctness. The questions that were used for these semistructured interviews can be found in Appendix A.5.

The following people / organizations were interviewed.

- Appendix A.1 shows the summary of the short telephone interview with Marion Veerbeek in her role as administrative secretary intergovernmental task force with the “Vereniging Nederlandse Gemeenten” (Association of Dutch Municipalities) on the 23rd of October.
- Appendix A.2 shows the summary of the interview with Pamela Young and Jan-Jouke Vos in their roles as Deputy Secretary-Director of the Election Council Secretariat, coordinator cluster Election Issues (Ms. Young) and IT staff member (Mr. Vos) at the Electoral Council Secretariat (Kiesraad secretariaat) on the 10th of October 2013.
- Appendix A.3 shows the summary of the interview with Jan Smit in his role as Chairman of the Election Committee at the “Nederlandse Vereniging Voor Burgerzaken” (Dutch Association of Civil Affairs) on the 11th of October 2013.
- Appendix A.4 shows the summary of the interview with Rop Gonggrijp in his role as chairman of the pressure group “We don’t trust voting computers” (Wij vertrouwen stemcomputers niet) on the 4th of October 2013.

A.1 VERENIGING NEDERLANDSE GEMEENTEN (ASSOCIATION OF DUTCH MUNICIPALITIES)

On Wednesday the 23rd of October, Marion Verbeek was contacted for an interview. During the phone call, both Verbeek and the author of this research came to the conclusion that internet voting is not on the agenda of the VNG. Therefore this interview was not held.

A.2 KIESRAAD (ELECTORAL COUNCIL SECRETARIAT)

On Thursday the 10th of October 2013, Pamela Young and Jan-Jouke Vos were interviewed according to a semi-structured list of questions. Young and Vos verified the results of the interview on the 4th of December 2013. Young and Vos both work at the Electoral Council Secretariat ("*Secretariaat van de Kiesraad*"). The role of the Electoral Council within the Dutch voting process is not to make the law but, among other roles, to advise government on proposed legislation and improvements in electoral process.

OPINION The Electoral Council ("*Kiesraad*") values the results that were presented by the Election Process Advisory Commission (Committee Korthals Altes) [103]. The safeguards presented in that report have since become the framework the Electoral Council works with, making explicit what always was used implicit in their advices.

In the past few years there haven't been any concrete proposals on (introducing) internet voting. That is why it hasn't recently been on the agenda of the Electoral Council. Young and Vos make clear that the Electoral Council is not in favor or against the use of technology in the voting process, they do however feel that it is time to discuss under which circumstances technology can be used and what risks are involved doing so. In their advices the Electoral Council checks if proposed legislation sufficiently complies with the safeguards that were found by the Committee Korthals Altes.

Part of why the discussion has come to a standstill, according to Young and Vos, is because the government hasn't come up with a clear set of requirements hardware and/or software should comply with, if it would be used in a voting process. If that set of requirements would exist, the discussion could be focused on whether a voting system complies with the requirements, instead of focusing the discussion on having technology involved in elections at all.

Young and Vos are of the opinion that the critics of technology with regards to the voting process focus a lot on the possibly new scenario, instead of looking at the current process critically as well. They argue that the current process could benefit from such a critical view too. They think that critics should be clear about the risks in both of the scenario's, so that we are able to come up with processes to secure the safeguards of the elections even better.

Compared to the total number of Dutch citizens working or living abroad, the current number of voters within this group is low, according to Young and Vos. They recognize that the current accessibility for Dutch citizens working or living abroad is under pressure due to the complicated process of becoming an eligible voter and the process of actually casting a vote. They think that politics in the Netherlands are now moving towards recognizing the tension among

the safeguards of the election process for voters abroad. Young and Vos expect that the non-residence register ("*niet-ingezetenen register*") will help increase the outcome of Dutch voters abroad. However, they recognize two possible gaps: the difference between eligible and non-eligible Dutch voters working or living abroad (those who do, or do not apply for eligibility); and the difference between eligible Dutch voters working or living abroad and the number of actual votes casted by them (those who could vote, but don't do it). The first would benefit a lot from the non-residence register. The latter could be boosted, among other solutions, by the introduction of internet voting.

REQUIREMENTS On the one hand, Young and Vos mention that market influence on the actual voting process is not beneficial for the verifiability of the election process. On the other hand, they argue that, generally speaking, it could be a good thing if a voting system in a country would not be under control of the government for the full 100%. They mention that, in some countries, it could also be problematic for the verifiability of the election process, if the complete infrastructure of internet voting would be under the control of the government.

In general, the Electoral Council considers independence of the elections as the top priority. Neither the government nor the market should have full control over the voting process. To have this independence, it is necessary to have a transparent process, which shows how the final results have been calculated.

PLAYING FIELD Young and Vos recognizes the playing field as the lobby parties (which can hardly be called lobby parties according to Young and Vos), the Electoral Council (*Kiesraad*, themselves), the government (including the two chambers), the Council of State (*Raad van State*) and the Ministry of Interior and Kingdom Relations. They furthermore think that the Municipality of The Hague (*Gemeente Den Haag*) is also a player in the playing field, because they handle incoming postal votes.

The lobby parties that were mentioned are the "NVVB", the "VNG", the "Stichting Wij vertrouwen stemcomputers niet" and the "NGB" ("*Nederlands Genootschap van Burgemeesters*"). Young and Vos mention that they have regular meetings with the NVVB and the VNG in which they discuss the practical issues concerning the voting process. They do not see the input of these parties as lobbying. They argue that these parties lobby with the Minister and his Ministry of Interior and Kingdom Relations, and probably at other politicians too. Furthermore, the "Stichting Wij vertrouwen stemcomputers niet" does not lobby at the Election Council either, they only do information requests.

Young and Vos don't think that there is a lobby from the market regarding internet voting.

They mention that Joost Taverne of the political party VVD is one of the strong advocates of internet voting in Parliament. He wants to have an easier way of organizing the elections, therefore he argues in favor of the return of the voting machine and also in favor of internet voting for Dutch voters working or living abroad.

In general, Young and Vos are of the opinion that everybody's arguments are clear, and that there are definitely groups either in favor, or against internet voting.

A.3 NEDERLANDSE VERENIGING VOOR BURGERZAKEN (DUTCH ASSOCIATION OF CIVIL AFFAIRS)

On Friday the 11th of October 2013, Jan Smit was interviewed according to a semi-structured list of questions. Smit verified the results of the interview on the fourth of December 2013. Smit works at the municipality of Groningen, besides that, he is the chairman of the Election Committee of the Dutch Association for Civil Concerns ("*Nederlandse Vereniging Voor Burgerzaken*", NVVB).

In general it must be noted that the NVVB is currently much more focused on the voting computer instead of internet voting, therefore, some remarks that are made by Smit are better suited for the voting computer.

OPINION Smit starts off by sharing that currently a lot is done via the internet, i.e. internet banking and filing taxes. We currently live in a digital society, in that spirit of time, internet voting would fit this 21st century.

Smit recognizes the safeguards found by the Korthals Altes Committee as being the benchmark for the election process. Smit tells that the NVVB wants to do as much as possible electronically regarding the voting process as long as it fits alongside the safeguards of the Korthals Altes Committee. He argues that the current costs and effort for the election process are very high due to the many polling stations and manual tallying of the votes.

When the voting computers in the city of Groningen were bought i.e., they were financially depreciated after being used seven times already. Smit thinks that it is worth researching the costs of acquiring and maintaining an internet voting system as long as that system falls within the legal framework and the safeguards presented by the Korthals Altes Committee, because then we need far less polling stations and government officials.

Internet voting could not only bring financial advantages, it could also make the voting process much more flexible. For example, the duration and timing of the voting period could easily be stretched.

In between the current situation (ballot voting) and internet voting are other milestones according to Smit, like printing of the ballot at home; using the chip in the personal ID card to identify a voter; A4 paper ballots for easier tallying and the most favorable to the NVVB: the voting computer. Smit questions why we needed to go all the way back to the ballot voting, *“couldn't we meet half way?”*

Smit is curious what the committee that is looking into the voting computer will come up with. He is of the opinion that whatever the outcome of the committee is, that should be respected if the following two questions can be answered: do we still want it and what does it cost? If these questions suggest a positive attitude towards the voting computer, they should be bought, otherwise not. Smit thinks that the results of a similar committee for internet voting should be handled in the same way, but he doesn't think that that will happen for regular voting in the Netherlands within ten years time.

The tallying that currently happens is, according to Smit, not 100% accurate. He questions the switch from electronic tallying (with the voting computer) towards manual tallying, *“is it as accurate as electronic tallying?”*. There have been reports from political parties which claim that their votes weren't tallied¹. Smit claims that tallying by the voting computer or servers in an internet voting system could reduce or mitigate this problem.

REQUIREMENTS Next to the costs, Smit agrees that the reliability and the transparency of a voting system are very important. All voting systems should fulfill the Korthals Altes safeguards, to be regarded an option in the voting process by the NVVB.

Currently, we allow voting by proxy (although there is a lot of criticism by the OSCE), Smit asks himself whether this doesn't allow *Family voting*², whereas that is an argument not to do internet voting. *“We apparently think this is acceptable”*, claiming that family voting is as much as a problem with proxy voting as with internet voting.

Smit is not sure whether the accessibility safeguard that is mentioned by the Korthals Altes Committee is currently met. He for example mentions that polling stations accessible for wheelchairs are sometimes located in another part of town. Internet voting would bring a certain level of accessibility to a subgroup of the population that is currently lacking this accessibility, according to Smit. However he argues that this solves the problem of a relatively small group of voters.

¹ For example, political party PVV voted in several cities on candidates very low on the list, and looked up the number of preference votes for these candidates. They found out that their votes on those candidates were sometimes not tallied. However their claims can't be checked, because proving what you voted for is impossible in the Netherlands. Smit mentions that individual voters too, sometimes claim they can't find their vote for a specific candidate within the published results.

² Family voting is a term for vote coercion

Smit is of the opinion that, if we would introduce internet voting, it should be understandable and clear. Not only should it be technically clear to at least a large group of voters how it works, but moreover should it be a clear webapplication in which voters can't get lost but clearly get how to actually vote.

When asked about the feeling the general public will have about technology in voting, Smit answers that he is not sure whether the public has lost trust in the voting computers. He furthermore thinks that internet voting would probably be trusted by many of the voters too. He argues that everybody is using internet a lot (i.e. for online banking or filing taxes). It would, according to Smit, fit the spirit of the time to allow voting via internet. But he believes the voting computers will show up at polling stations much earlier than internet voting would be introduced.

PLAYING FIELD The NVVB takes on the view of the municipalities and the government officials in the conversations with the Ministry of Interior and Kingdom Relations, the VNG and the Electoral Council. The NVVB doesn't really see this as lobbying. They *"use each others expertise"*.

Smit recognizes the points by the *"Stichting Wij vertrouwen stemcomputers niet"*. He argues that the possible security leaks presented by them are very theoretical and needed very expensive equipment. With the introduction of other measures, possible fraud could have been prevented.

A.4 WIJ VERTROUWEN STEMCOMPUTERS NIET (PRESSURE GROUP "WE DON'T TRUST VOTING COMPUTERS")

On Friday the 4th of October 2013, Rop Gonggrijp was interviewed according to a semi-structured list of questions. Gonggrijp verified the results of the interview on the 3rd of February 2014. Gonggrijp is the chairman of the pressure group *"Wij vertrouwen stemcomputers niet"* (We don't trust voting computers) with which he was responsible for the switch from voting with the use of voting computers in a polling station, back to the paper ballot voting in a polling station. Gonggrijp furthermore is a computer scientist with a great passion for and interest in security.

OPINION Gonggrijp identifies several arguments that are used to introduce the topic of internet voting (or sometimes the voting computer) within Dutch politics. For example, he notices that an argument that is used a lot, is that the municipalities have tremendous trouble during elections: they need to find volunteers or government officials to man the polling stations and afterwards tally the votes, which regularly takes until the middle of the night, with possibly

much less accurate results than if tallying would have happened by a computer. He furthermore thinks that the accessibility of elections is one of the main arguments in favor of internet voting for both increasing the voter turnout at Dutch polling stations and dramatically increasing the voter turnout for Dutch citizens living or working abroad.

Gonggrijp argues that the problems for which internet voting (or voting computers) would be the solution (among others, tallying and increasing the voter turnout), could be mitigated without the introduction of either voting computers at a polling station or internet voting. He, for example, argues that polling stations are currently held open until 9 o'clock in the evening to ensure the accessibility for those that have to work long hours during the day. In his opinion, doing elections on Sundays would increase the likelihood of a high voter turnout during the day, meaning that polling stations can close and start tallying much earlier. He cites the progress that was made towards a new paper ballot that could be used for voting. This A4 paper ballot is believed to make tallying significantly quicker and less prone to errors. He furthermore is of the opinion, that increasing the voter turnout for Dutch citizens working or living abroad could already be reached with the upcoming non-residents register (*niet-ingezetenen register*), which will be introduced in 2014. This would make the registration for the list of eligible voters much more convenient for voters abroad.

Next to the arguments which would favor internet voting, Gonggrijp also recognizes some assumptions about security for internet voting or security in general, which he claims to be false. He for example quotes the bill by Joost Taverne³ from the political party VVD, in which Taverne argues that there is much more trust in internet systems and that the security situation for cyber space has changed positively. Gonggrijp disagrees strongly with Taverne on this point, arguing that the recent news about Edward Snowden, Diginotar, Stuxnet and the DDoS' at Dutch banks⁴ prove that cyber space hasn't gotten any more secure over the recent years. *"What kind of illusions have you got if you think about building such critical systems securely?"*

He believes that many people see the security of a voting system similarly as the security needed to secure online banking transactions. He argues that the trust model of banks is a top-down one, having a central "key" and authority, compared to the bottom-up trust model

³ The bill is ment to pave the way for alternative voting methods like the voting computer or e-voting. <https://zoek.officielebekendmakingen.nl/kst-33354-3.pdf>

⁴ Gonggrijp means the following with the examples given: Edward Snowden proved the tremendous efforts by the US Government to get control over cryptographic standards; Diginotar showed that non-technical processes securing keys and servers are not satisfying the risks involved; Stuxnet offered an insight into Advanced Persistent Threats in which even offline systems were prone to malware; the DDoS attacks at Dutch banks show that availability is hard to ensure.

needed for voting systems, in which you still want to have valid elections when you cannot trust the government.

He repeatedly mentions that he has much more trust in 40.000 tallying volunteers / government officials than in any secure computer system one can think of. Even if the results are not 100% accurate due to either actively or mistakenly introducing errors in the tallying process, he argues that this would fade out either by the large population or by the fact that mistakes are mathematically speaking noise, which is stochastically distributed over all the parties.

Gonggrijp notices that the last system that was used for internet voting in the Netherlands (RIES) was prone to a lot of simple and basic mistakes. He for example shares that, if someone would still have copy of the Bulletin Board that was used for publishing the votes, with current computing power could decrypt all the votes. He furthermore asks himself how to guarantee that the intelligence agency (AIVD), with much more knowledge and computing power, would not try to identify who voted i.e. in favor of a fundamentalist Muslim party, or a fundamentalist right wing party? He mentions that such efforts have been made before by BVD (the former AIVD), to identify voters for the CD (former fundamentalist right wing party).

Currently, due to the voting by proxy ("*volmacht*"), some woman lose their right to vote to their husband (especially in religious minorities), and this would be further deteriorated if internet voting would be allowed, according to Gonggrijp.

REQUIREMENTS Gonggrijp explains that for him, ballot privacy is the fundamental requirement for any election. This should always be the number one requirement when designing a voting system, thus, he rules out that accessibility to elections should ever way out in favor of ballot privacy. In his opinion, no system that involves electronics or the internet could ever guarantee ballot privacy better than the currently employed voting system with paper ballots. He furthermore believes that the postal voting, used for Dutch citizens working or living abroad, issues ballot privacy better than an electronic or internet voting system: the security requirements necessary to guarantee a valid result of postal votes are similar to those of the regular voting process.

Next to ballot privacy, verifiability of the results and controllability of the voting system are extremely important to Gonggrijp. Since all internet voting systems would be using (a lot of) cryptography, he argues that only a few hundred people actually understand what is really going on, making the controllability of the voting system (and thereby the verifiability of the results) limited to only a small portion of the eligible voters. He motivates that a voting system should be understandable for way more voters. He furthermore disagrees with experts on voting systems who claim that end-to-end verifiability is

all you need. He for example mentions that for internet voting to really work, you still need a lot of non-technical processes which could introduce flaws in the results, i.e. you need the servers with really secret keys, you need to generate the secret keys of the server in a secret environment, etc.

Gonggrijp wants to stress that he favors the opportunity that one could register for voting online, similar to what the non-residents register (*niet-ingezetenen register*) will offer. He is okay with digitalizing the whole voting process, except for the part where actual votes are cast and tallied.

The verifiability that could be offered to voters in an internet voting system will be, according to Gonggrijp, hardly ever used. He mentions that within the RIES voting system, only several tens of voters actually verified their vote. Similarly, Gonggrijp motivates that the same will apply to the paper re-vote offered to Estonian voters.

The verifiability of the hard- and software that would be needed to allow internet voting can only be done if the companies providing it would allow a full open source code check. He argues that this was tried to be forced upon in Austria⁵, but that turned out to be a farce. If, and Gonggrijp is definitely against this, internet voting would be allowed, this is extremely important.

Gonggrijp remarks that all votes that are casted in an insecure environment (like a computer), are by definition insecure. Hereby he concludes that internet voting has *“too much complexity with some very beautiful illusions”*.

PLAYING FIELD Gonggrijp recognizes the playing field as being a few lobby parties (among which themselves as a pressure group), the Electoral Council (*Kiesraad*), the government (including the two chambers), the Council of State (*Raad van State*) and the Ministry of Interior and Kingdom Relations. He furthermore recognizes a strong lobby by the market, not only by the actual developers of electronic voting systems (among which internet voting systems), but also the consultancy firms which push the government into more eGovernment related projects.

He describes the two lobby parties (VNG and NVVB) as being reasonable parties who try to opt for the best opportunities for municipalities and government officials. He agrees with the fact that tallying does take a long time, but motivates that it could be done much more efficient (i.e. by using the A4 paper ballots mentioned before). He does not resent that these lobby groups try to lobby for their own

⁵ Barbara Ondrisek, who published about e-voting before, argues that the source code review that was performed for the Austria Student Election was not appropriate. She argues that, for example, the experts could only review the source code for one day and the experts had to sign an NDA. She argues that Scytl did not put effort in the Security Analysis. Source: <http://papierwahl.at/2009/05/11/bmwf-behindert-oh-wahlkommissionen/>

good, although they don't see the bigger picture of the democratic process they are involved in.

Another large lobby was found by Gonggrijp to be the market, he is convinced that municipalities are being coerced by market parties like Nedap (Groenendaal)⁶ and consultancy firms.

The Electoral Council (*Kiesraad*), government and both of the chambers set a lot of store by what the municipalities and their mayors say, according to Gonggrijp. The real security issues involved with the voting computers and internet voting are not seen by municipalities.

According to Gonggrijp, it is not surprising that most prominently the political party VVD tries to smoothen the process for Dutch citizens working of living abroad. He argues that the number of additional voters would mean about 2 of the 150 chairs in the first chamber, and according to research brought up by Gonggrijp, the majority of them would vote for the VVD. Which, according to Gonggrijp, still is a legitimate motivation: more involvement in the democratic process.

A.5 QUESTIONS USED FOR THE INTERVIEWS

The questions that were asked during the semistructured interview are presented below. They are ordered in different categories, not particularly represented in the summary of the conversations.

GENERAL QUESTIONS

1. Do the current voting methods, according to you, satisfy for the needs of citizens (voting booth, proxy voting and postal voting)?
2. What are, according to you, the safeguards for fair elections?
3. If there would be tension between the safeguards for fair elections. How would you decide which of the safeguard is more important? Are there any more "fundamental" safeguards than other?
4. What do you think about the safeguards proposed by *The Election Process Advisory Commission* (transparency, verifiability, fairness, eligibility, free suffrage, secret suffrage, equal suffrage, accessibility)?
5. Do the current voting methods satisfy these safeguards? And how much?

⁶ Gonggrijp refers to the letter written by the former Groenendaal CEO in which he threatens the government not to stop using the Nedap voting computers. Source: http://wijvertrouwenstemcomputersniet.nl/images/7/7e/20061110_groenendaal2bzk-koop_mijn_bedrijf_of_ik_kap_er_nu_mee.pdf

6. Who does, according to you, decide upon the safeguards for elections in the Netherlands?
7. Who influences this decision maker the most?

VOTING FROM ABROAD

1. What do you think about the current voting method for Dutch citizens living or working abroad (postal voting)?
2. A possible alternative for postal voting is internet voting. What do you think about this voting method?
3. Does internet voting satisfy your safeguards for fair elections?
4. Does internet voting satisfy the safeguards by [The Election Process Advisory Commission](#)?
5. What are, according to you, the differences between postal voting and internet voting on the topics of:
 - Feasibility to satisfy the safeguards by [The Election Process Advisory Commission](#);
 - Feasibility of your more “fundamental” safeguards for fair elections;
 - Procedures defined for elections.

INTERNET VOTING

1. Do you think internet voting could be a substitute or a complement to postal voting?
2. Do you think internet voting could be a complement to regular voting in a voting booth?
3. What do you think about the recent developments of internet voting in other countries (Estonia, Norway, ...)?
4. Do you see any opportunities for implementing internet voting in the current voting process (either for all voters or for voters living or working abroad)?
5. Do you think internet voting will ever be used for elections in the Netherlands?
6. What technological improvements should be made to implement internet voting?
7. What societal improvements should be made to implement internet voting?
8. What additional improvements should be made to implement internet voting?

9. What are the success factors to allow for the implementation of internet voting?
10. What are the biggest opportunities and threats for the implementation of internet voting (Technical, societal, political, media, ...)?

PLAYING FIELD

1. What are, according to you, the players in the playing field surrounding internet voting (e.g. lobby groups, suppliers, government bodies, advisory bodies, media, ...)?
2. What are the opinions and interests of these parties (Goals and arguments)?
3. What are the actions these parties undertake to get to their goal (lobbying, solicited advice, unsolicited advice, decisions, ...)?
4. What are the means these parties have to give their actions strength (finance, pressure, media, politics, ...)?
5. What is your opinion on these players?
6. What is your own position in this playing field (including opinions, interest, actions, means, ...)?

APPROACHES TO SAFEGUARDS

This appendix will get into depth with different approaches to safeguards. Section [B.1](#) will address the international context, while section [B.2](#) will cover the literature view on safeguards.

B.1 INTERNATIONAL APPROACH TO SAFEGUARDS

This section covers the international laws and regulations that concern the e-voting practice in the Netherlands in appendix [B.1.1](#) and presents the international safeguards that are found for e-voting projects in appendix [B.1.2](#), this is concluded in subsection [B.1.3](#).

B.1.1 *International laws and regulations*

Different international laws and regulations describe the necessity of national elections. Some of the safeguards implemented in countries can be dated back to these laws and regulations, therefore, they will be discussed shortly in the next paragraphs.

Elections are defined in the Universal Declaration of Human Rights from the United Nations: *“The will of the people shall be the basis of the authority of government; this will shall be expressed in periodic and genuine elections which shall be by universal and equal suffrage and shall be held by secret vote or by equivalent free voting procedures”*. (United Nations General Assembly [[105](#), Article 21(3)])

Eligibility has been defined in the International Covenant on Civil and Political Rights from the United Nations: *“Every citizen shall have the right and the opportunity, without distinction of any kind, such as race, colour, sex, language, religion, political or other opinion, national or social origin, property, birth or other status and without unreasonable restrictions to vote and to be elected at genuine periodic elections which shall be by universal and equal suffrage and shall be held by secret ballot, guaranteeing the free expression of the will of the electors”*. (United Nations General Assembly [[106](#), Article 25(b)])

Free elections are also described in The European Convention on Human Rights from the Council of Europe: *“The High Contracting Parties undertake to hold free elections at reasonable intervals by secret ballot, under conditions which will ensure the free expression of the opinion of the people in the choice of the legislature”*. (Council of Europe [[22](#), Protocol 1 Article 3])

*Universal
Declaration of
Human Rights from
the UN*

*International
Covenant on Civil
and Political Rights
from the UN*

*European
Convention on
Human Rights from
the Council of
Europe*

*Copenhagen
Meeting of the
OSCE*

Guidelines for good practice in Electoral Matters from the Venice Commission

In the Copenhagen meeting from the OSCE ¹, two agreements were made on elections: *“The participating states solemnly declare that among those elements of justice which are essential to the full expression of the inherent dignity and of the equal and unalienable rights of all human beings are the following: [...] free elections that will be held at reasonable intervals by secret ballot or by equivalent free voting procedure, under the conditions which ensure in practice the free expression of the opinion of the electors in the choice of their representatives”*, ([20, Agreement 5.1]) and *“To ensure that the will of the people serves as the basis of the authority of government, the participating States will [...] ensure that votes are cast by secret ballot or by equivalent free voting procedure, and that they are counted and reported honestly with the official results made public”*. ([20, Agreement 7.4])

The European Commission for Democracy Through Law from the Council of Europe (Venice Commission) issued a guideline for good practices in electoral matters which included the following article: *“For the voter, secrecy of voting is not only a right but also a duty, non-compliance with which must be punishable by disqualification of any ballot paper whose content is disclosed”* Venice Commission [108, Article 4(a)].

All of the national legislations in different countries should implement these laws and regulations, they furthermore need to ensure that their electoral protocol and procedures comply with them as well. In their report, *The Election Process Advisory Commission* show how they comply with these laws and regulations.

B.1.2 International safeguards

Recommendations for legal, operational and technical standards for e-voting by the Council of Europe

The Council of Europe [23] has recognized the importance of having guidelines for fair elections for e-voting already in 2004. In a Recommendation from the Council, the following is stated: *“Conscious, therefore, that only those e-voting systems which are secure, reliable, efficient, technically robust, open to independent verification and easily accessible to voters will build the public confidence which is a pre-requisite for holding e-voting, Recommends that the governments of member states, where they are already using, or are considering using, e-voting comply, [...] with paragraphs [...] below, and the standards and requirements on the legal, operational and technical aspects of e-voting, as set out in the appendices to the present Recommendation[.]”* [23, p7]. In the Recommendations, different principles (universal suffrage; equal suffrage; free suffrage; and secret suffrage) and procedural safeguards (transparency; verifiability and accountability; reliability and security) are discussed in great length. Furthermore, the technical requirements and operational standards that are described further on are also very thorough and pre-

¹ Organization for Security and Co-operation in Europe.

cise. The total of principles, procedural safeguards, technical requirements and operational standards, sums up to 112 recommendations.²

It is no coincidence that the list of principles and procedural safeguards is similar to the list of safeguards proposed by [The Election Process Advisory Commission](#). The committee took them into account when composing their safeguards. Interestingly, the committee choose, not include accountability and security and renamed the reliability to accessibility. Both accountability and security are implicitly present, security can be found in the fairness safeguard and accountability is addressed separately in the report by the committee.

In Austria, the election of a student body was performed, and later analyzed according to the recommendations of the [Council of Europe](#). The elections suffered of four types of attacks which let to different comments to the recommendations [34]. The attacks and additional comments on the recommendations were the following:

- A DDoS attack was performed on the voting servers three days before the preparations started. Leading to the comment of allowing paper re-voting on election day to ensure availability;
- A fake (phishing) website was setup to phish for voting credentials. Leading to the comment of making the implementation of adequate countermeasures to phishing a more explicit necessity;
- A smear campaigns was setup by making a video in which votes were flipped through malware (but the vote wasn't actually flipped, the video was fake). Leading to the comment of making a special security strategy to increase the acceptance level of e-voting among the other voting channels;
- A vote buying campaign was setup to harvest as many voting credentials as possible (probably a smear campaign too). Leading to the comment of making awareness programs, trained staff and well-designed processes a requirement.

*Analysis on the
Recommendations
from the Council of
Europe*

B.1.3 Conclusion

What can be concluded from the international approach to safeguards, is the fact that [The Election Process Advisory Commission's](#) safeguards are compliant with both the international laws and regulations the Netherlands should comply with, as well as with the best practices stated by the [Council of Europe](#). This means that these safeguards can be applied to the case of e-voting as well. The 112 recommendations posted by the [Council of Europe](#) should however be

² An interesting read is the analysis by an independent institute for the Norwegian government on compliance with these 112 recommendations for their e-voting project: Esteve et al. [39].

checked upon implementing the actual e-voting scheme, as well as any additional recommendations as e.g. the authors of Ehringfeld et al. came up with since 2004.

B.2 LITERATURE APPROACH TO SAFEGUARDS

The following subsections will describe the different safeguards that are found in the literature. Where possible, they will be compared to the Dutch safeguards.

B.2.1 Norwegian e-voting project

In an extensive survey, performed for the Norwegian government by an independent international organization, Esteve et al. [38] compared the different e-voting schemes deployed in other countries. Furthermore, they remark the safeguards for establishing trust in the Norwegian e-voting scheme, based on the work by Spycher et al. [99]. *“It is important to note that as Spycher et al. state, the mechanisms used to enable trust are complex, can be difficult to enact, and can entail significant extra cost and complexity for any internet voting project.”* [38, p39] They furthermore state that it is not necessary to comply with all of them, but to consider them *“as a menu of options for obtaining the required level of trust [...] However, the more of these mechanisms that are implemented, the higher the level of trust is likely to be in any Internet voting system”* [99, p10].

The safeguards proposed by Esteve et al. [38] is a superset of those by Spycher et al. [99], and include the following:

- Transparency, by which they mean both the information about the system itself, but also the ability to disclose the findings of analysis conducted on the basis of this access;
- Separation of Duties, which is argued by Pieters and Becker [81] to ensure tampering with the election results is made much more difficult. *“Such compartmentalization of roles prevents any one party from exercising too much responsibility and power over the system”* [38, p32];
- Enabling vote updating, which is important to ensure the secrecy and freedom of a vote;
- Enabling verifiability, which could count as partial transparency of the system. According to Pieters and Becker [81], transparency supersedes the need for absolute vote secrecy, *“especially as people are voting from unsupervised environments anyway. They accept the failure in vote secrecy as necessary to implement verifiable Internet voting. However, others argue that the secrecy of the vote is of greater*

concern than transparency." [38, p34]. But the recommendation by the Council of Europe [23] explicitly forbids receipts.

- Evaluation, which is concerned with general IT system evaluation, for example those of the Common Criteria;
- Test elections, which are inspired by the recommendations of the Council of Europe [23], are in place to establish trust in the election system by *"providing [them] with an opportunity to practice any new method of e-voting before, and separately from, the moment of casting an electronic vote"* [23, recommendation 22];
- Integrity of the System, newly introduced in [38], which covers the range of features to ensure the reliability of voting systems, includes (but not limited to): voter authentication mechanisms, equality of the vote, system security mechanisms, cryptographic protocols and encryption key control;
- Testing / Certification / Audit, newly introduced in [38] in order to comply with the new Council of Europe [24] regulations concerning e-voting systems. It includes the testing, certification, formal certification and audit of voting systems.

The safeguards that were identified by [Spycher et al.](#), with additions from [Esteve et al.](#), are on a somewhat different level of abstraction than the safeguards posed by [The Election Process Advisory Commission](#). On the one hand, they are very specific by forcing the scheme to allow for re-voting, but on the other hand a safeguard which forces the integrity of the system is rather broad. Due to this mismatch in abstraction level, it is not possible to compare them to the Dutch safeguards. Interesting however are the more implementation-driven advices about the separation of duties, the vote updating and the certification which are well argued design principles.

In 2011, Volkamer et al. [112] already announced a subset of the above list, which also included the *"Allowing independent implementations of voting client software"*. In later work, the same authors got rid of this safeguard. Again, this safeguard is more of a design decision than an actual safeguard, but moreover, it is debatable if this is a requirement that spans all implementations of e-voting schemes around the globe. As is shown in section 4.1, the Estonian system was confronted with a proof-of-concept attack which actively abused the fact that the client could be persuaded not to use the regular voting application. On the other hand, it could be argued that, when offering a completely transparent e-voting scheme, it is not possible to forbid the client from using a self-made voting application. A well-informed discussion about this safeguard is considered out-of-scope and left until the design phase of an e-voting scheme.

B.2.2 Threat analysis

Another analysis of the safeguards for e-voting schemes focuses on the way requirements mitigate the threats posed to an e-voting scheme. The research, performed by Volkamer and Grimm [111, p98] in 2006, found two open problems for which the identified requirements could not mitigate: ensuring the *possibility* to vote and limiting that to a *maximum* of one vote despite any type of failure; the difference in adapting a vote to political events between voting in advance with postal voting and e-voting. The following quote shows the way Table B.2 was constructed [111, p98-99].

Deduced from the universal principle the election system must ensure that no eligible voter is excluded from the election - Req_u. This must also hold for any kind of server or client software breakdown as well as communication breakdown. In addition, no voter has the possibility to cast more than one ballot within such a break down (equal). To ensure the equality principle, no unauthorized person should be able to add, remove or alter votes undetected. This must hold during ballot casting - Req_{e1}, ballot transmission - Req_{e2} and ballot storage - Req_{e3}. The principle of secret elections demands that only the voter is aware of her voting decision. Nobody else is able to link the voter to her vote neither during nor after the election - Req_{s1}. In addition, voters must be unable to prove their voting decisions - Req_{s2}. There are two more requirements, which are less technical but more general. The principle of free elections requires that voters cast their ballot free of duress and without influence - Req_f. In addition, the principle of equal elections requires that all voters can cast their ballots in the same way - Req_{e4}.

An attacker has four attacking points either in order to break the ballot secrecy (violation of the secret and free election principle) or to manipulate the election result (violation of the equal, free and universal election principle):

Observing a voter casting her ballot - The attacker could be next to the voter casting her ballot in order to observe the voters choice or to coerce her to vote in a specific way (e.g. imaginable in an old people's home) - Threat_O. This is not an online voting specific attack but one for any remote voting system because the electoral office cannot ensure that voters cast their ballots in a free and secret environment. This is why postal voting is not allowed in many countries, and in some countries only as an exception.

Manipulation of the voters' voting device - The attacker could also program malicious code and try to install it on the voter's PC. This code could read the voter's ID, and vote on his behalf - Threat_{D1}, or change the voter's choice before sending it to the electoral server - Threat_{D2}. Moreover, attacking the voter's PC is much more critical than the observation attack from above because now it is possible to manipulate or read several votes automatically. Of course, this attacker needed technical expertise.

Manipulation or sniffing on the communication layer
- The Internet is a public network so we cannot prevent an attacker to read or manipulate the connection between the voter and the electoral servers. The attacker can try to manipulate the election result by changing, adding or deleting ballot messages on the network - Threat_M. He can also read and store messages in order to evaluate them - Threat_S. The attacker could wait until someone will find a fast algorithm or faster PCs to decrypt the stored messages.

Manipulation of the election servers - The election servers store beside other data both information, the voters' IDs and their votes. Thus, an attacker could try to get access to the election servers in order to get the corresponding data - Threat_{E1}. He could also try to manipulate the servers - Threat_{E2}.

As can be deduced from the research performed by Volkamer and Grimm [111], the following set of requirements must be met when designing an e-voting scheme:

- No eligible voter is excluded from the election (Req_u);
- No unauthorized person should be able to add, remove or alter votes undetected:
 - During ballot casting (Req_{e1});

Table B.2: Comparison Requirements - Threats [111, p99]

	Req _u	Req _{e1}	Req _{e2}	Req _{e3}	Req _{s1}	Req _{s2}	Req _f
Threat _O					x	x	x
Threat _{D1}					x	x	x
Threat _{D2}	x	x					
Threat _M			x				
Threat _S					x	x	x
Threat _{E1}					x	x	x
Threat _{E2}				x			

- During ballot transmission (Req_{e2});
- And during ballot storage (Req_{e3});
- Nobody else is able to link the voter to her vote neither during nor after the election (Req_{s1});
- Voters must be unable to prove their voting decisions (Req_{s2});
- Voters cast their ballot free of duress and without influence (Req_f);
- All voters can cast their ballots in the same way (Req_{e4}).

Although these requirements are necessary, according to [Volkamer and Grimm](#) this does leave the two earlier mentioned problems. The second (adapting a vote to political events during the longer voting period than regular voting) is out of scope for this research, since it is more of a philosophical discussion than a risk for e-voting. The first problem (availability of voting interface and limiting the voter to vote only once, under all circumstances) could be turned into a requirement for e-voting schemes too, as is done by, for example, [The Election Process Advisory Commission](#).

Table B.3 demonstrates how the requirements from [Volkamer and Grimm](#) relate to safeguards proposed by [The Election Process Advisory Commission](#). As can be seen from the Table, the "transparency" and "verifiability" safeguards are not found in the requirements from [Volkamer and Grimm](#), while [The Election Process Advisory Commission](#) does not have a safeguard in place which ensures that all voters can cast their ballots in the same way. The latter point being more of a philosophical than an e-voting safeguard, and is therefore considered out of scope. In general, the safeguards map relatively good.

Table B.3: Volkamer and Grimm [111] vs The Election Process Advisory Commission [103] on election safeguards

	Transparency	Verifiability	Fairness	Eligibility	Free suffrage	Secret suffrage	Equal suffrage	Accessibility	Not accounted for
Req _u				✓			✓	✓	
Req _{e1}			✓				●		
Req _{e2}			✓				●		
Req _{e3}			✓				●		
Req _{s1}						✓			
Req _{s2}						✓			
Req _f					✓				
Req _{e4}									✓
Not accounted for	✓	✓							

B.2.3 Ethics of e-voting

Pieters and Becker [81] conducted a study requirements and values for internet elections. Based on a literature study, they found the following three requirements and their subrequirements.

- Correctness of the results:
 - Only eligible voters vote;
 - They only vote once;
 - All votes counted are valid votes and all valid votes are counted.
- Verifiability of results by involved parties
- Secrecy of votes
 - No should be able to derive a relation between the vote cast and the involved voter (preventing forced voting);
 - A voter should not be able to prove which vote she cast (preventing sale of votes).

It is easy to see that the Dutch safeguards do cover all of these requirements, and in fact, even provide more safeguards than those (namely transparency and availability). One could discuss whether the Dutch safeguards comply with the validness of votes subrequirement, but for this research, it is considered to be compliant.

Besides this list of requirements, the authors also reason about the necessity of all the requirements for e-voting. They argue that when voting from an uncontrolled environment anyway, a proof of vote could aid the verifiability of the system while on the other hand, does not really help an attacker: if the attacker controls the uncontrolled voting environment anyway – for example by looking over a voter’s shoulder or by demanding the voting credentials – the proof of a vote is worth much less than in the controlled environment of voting in a voting booth [81].

They on the other hand discuss the actual value of verifiability of elections, if it mostly depends on the individual voter verifying his individual vote. How many people will actually do this, and what will this mean for the democratic value of the elections? While this value is indeed questionable, they argue that this level of individual verifiability outperforms the individual verifiability of regular ballot voting, since you can trace your own vote through the (rather complex) e-voting system, while you can’t do that with paper voting.

The Election Process Advisory Commission [103, p21-25] themselves also note some tension between the different voting safeguards:

- Accessibility could affect free suffrage: the more accessible elections are, the less the voting environment can be controlled with trained staff;
- Fairness and equal suffrage could affect secret suffrage: the fact that every vote should affect the final result should be in balance with the secrecy of suffrage, complying with both is hard;
- Free and secret suffrage could affect availability: if everyone (also disabled voters or voters abroad) should be able to vote, how do we control the voting environment to allow for free and secret suffrage;

They furthermore note the need for fast election results and having elections for reasonable costs.

Reasoning in general about election requirements could debouch into an directed graph of requirements with subrequirements somewhat similar to those composed by Pieters and Becker. Starting from the very nature of elections: elections are meant to derive the will of the population. Therefore it is essential to have fair elections, which consists of allowing everyone to vote, with a maximum of one vote per voter and to make those elections accessible to everyone. Furthermore, to allow everyone to vote, it should be clear that these votes and up correctly in the results. In order to have this fairness property, we also need everyone to vote free from any pressure, which can only be assured when there is some sort of vote secrecy. Free suffrage also needs a property which prohibits proving of a vote. Fair elections

furthermore need to be transparent and clear. The following directed graph could be constructed from this reasoning.

- Fair elections (have an election results that resembles the will of the citizens)
 - Eligibility (allowing everyone to vote);
 - * Equal suffrage (only once);
 - * Accessibility (in an accessible manner);
 - * Verifiability (and be sure that this worked correctly);
 - Free suffrage (vote free of pressure);
 - * Secret suffrage (have the ability to vote secretly);
 - * Receipt-freeness (have no proof of what you voted);
 - Transparent and easy to understand (everyone should be able to see how votes and up in the results);

B.2.4 *Functional requirements*

Recently, the ballot independence has gotten new attention, as discussed in section 2.2.1. According to one of the advocates for this property, it is not so much a privacy property, but it relates more to fairness [21]. Besides the ballot independence, the same authors mention two more properties that they think are vital for fair elections, namely “no early results” and “no pulling out”. They respectively mean that no partial results will be published before the election is closed and once the elections have closed and (partial) results are becoming public, it is not possible to pull out anymore.

None of these functional requirements are explicitly found in the safeguards from [The Election Process Advisory Commission](#), it could be argued however that these are partly covered in the fairness safeguard. Furthermore, the abstraction level of these additional functional requirements is lower than the safeguards. Therefore, they are considered to be covered enough by the safeguards from [The Election Process Advisory Commission](#).

B.2.5 *Conclusions on compliance with e-voting literature*

The safeguards from [The Election Process Advisory Commission](#) [103] do comply with international laws and regulations and meet the principles and procedural safeguards by the Council of Europe [23]. They furthermore comply with the safeguards found after a threat analysis on e-voting by Volkamer and Grimm [111] and comply generally to the additional functional requirements identified in literature.

This section also showed that the different requirements for voting schemes have a certain order in which they help achieve another

(higher level) requirement. One could reason from this perspective, that certain leaves in this ordered graph are of less importance than other nodes in the tree, although additional research is needed to prove this. Note that the requirements that were found for the ethical perspective of e-voting are exactly the same as those proposed by [The Election Process Advisory Commission](#).

CRYPTOGRAPHY

To allow for any of the described privacy or verification related properties, cryptographic building blocks are used. In the following subsections, the cryptographic building blocks to fully understand these e-voting properties will be presented, starting with encryption, going to zero-knowledge proofs followed by complete cryptographic systems used in practice.

Keep in mind that the following subsections in no way represent the full concepts of these cryptographic building blocks, they merely cover the few basic steps involved and the value these concepts add to the voting schemes. For further reading, the reader is referred to interesting material were possible.

C.1 ENCRYPTION

This subsection describes the most important encryption, hash and signature types for voting schemes. First, symmetric encryption is described, thereafter asymmetric encryption will follow, followed by probabilistic encryption, homomorphic encryption, malleability and threshold encryption. This subsection will end with commitments.

The format that will be used is as follows.

$data$ = the data to be protected

key = the key used in that part of the process

$enc(data, key)$ = encryption of the data with the key to the ciphertext

$dec(data, key)$ = decryption of the ciphertext with the key to the data

pk = the public key used in that part of the process

sk = the secret key used in that part of the process

$h(data)$ = hashing of the data

C.1.1 *Symmetric encryption*

Encryption is used when data needs to be secure against eavesdroppers. The simplest form of encryption is called symmetric encryption, in which the algorithms to encrypt and to decrypt the data uses the same key.

$$dec(enc(data, key), key) = data$$

To use symmetric encryption in communication between two parties, both parties need to know this shared key beforehand. This regularly

needs some form of key agreement protocols or key distribution protocols to be able to work. Symmetric encryption is considered to be a very efficient type of encryption, also with a great support within hardware solutions like smart cards or other small types of processors. Examples of symmetric encryption include AES [26], which has the following cryptanalysis properties: AES-128's key can be recovered with a computational complexity $2^{126.1}$, AES-192 with a computational complexity of $2^{189.7}$ and AES-256 with a computational complexity of $2^{254.4}$, but with related-key attacks, AES-192 and AES-256 can be broken with a computational complexity of 2^{176} and $2^{99.5}$ respectively [6]. Until now, when configured correctly, these are considered unbreakable.

For further reading, the reader is referred to Backes and Pfitzmann [3].

C.1.2 Asymmetric encryption

In contrast with the symmetric cryptography, asymmetric cryptography works by having separate keys for both operations, encryption and decryption.

$$\text{dec}(\text{enc}(\text{data}, \text{key}), \text{key}) \neq \text{data}$$

$$\text{dec}(\text{enc}(\text{data}, \text{pk}), \text{sk}) = \text{data}$$

To get asymmetric encryption to work, the secret key and the public key should be related. This can be achieved using mathematical principles called one-way trap-door functions. This works as follows: with some secret information y , it is easy to compute x from $f(x)$, but it is hard to do without y . An example trap-door function is modular multiplication: with modular n , message m , public key h , secret key x and ciphertext c .

$$h = g^x \text{ mod } n$$

$$c = \text{enc}(m, h) = m * h = m * g^x \text{ mod } n$$

$$m = \text{dec}(c, x) = \frac{c}{g^x} = c * g^{-x} = m * g^x * g^{-x} (= m) \text{ mod } n$$

To break asymmetric cryptography, one needs to break the trapdoor function. For modular exponentiation, this can be done in

$$e^{((\frac{64}{9})^{1/3} + o(1)) * (\ln n)^{1/3} * (\ln \ln n)^{2/3}}$$

steps (using Index Calculus [92]). Another well known example is RSA [87], which uses integer factorization as trapdoor function which can be broken in $e^{(0.5 + o(1)) * (\ln n)^{0.5} * (\ln \ln n)^{0.5}}$ steps (using Number Field Sieve [68]). Until now, when configured correctly and used with a large security parameter, these are considered unbreakable.

For further reading, the reader is referred to Rivest et al. [87], El-Gamal [35] and Paillier [78] for RSA, ElGamal and Paillier encryption respectively.

c.1.3 Probabilistic encryption

When using asymmetric encryption to encrypt e.g. a vote or another guessable piece of information, an eavesdropper can easily verify what you voted for, by encrypting all candidates with the public key, and compare it to the encrypted vote. If it matches, you have found the vote.

With symmetric encryption, this is not as easy, because the eavesdropper doesn't know the key and therefore cannot construct the list of encrypted votes. However when several voters e.g. use the same key, an eavesdropper can find side channel information from comparing different votes.

To solve these two issues, probabilistic encryption was introduced. Probabilistic encryption provides for different ciphertexts related to the same data. ElGamal cryptography [35] is an example of probabilistic encryption, using modular exponentiation as trapdoor function: with modular n , message m , public key h , secret key x and ciphertext (c_1, c_2) .

$$\begin{aligned}h &= g^x \bmod n \\c_1 &= g^y \bmod n \\s &= h^y = h^{x \cdot y} \bmod n \\c_2 &= m \cdot s \bmod n \\m &= \frac{c_2}{c_1^x} = c_2 * c_1^{-x} = c_2 * g^{y \cdot (-x)} = m * g^{x \cdot y} * g^{-x \cdot y} (= m) \bmod n\end{aligned}$$

Since the sender of the ciphertext chooses the y randomly, the ciphertext (c_1, c_2) changes for similar messages m because of the changing random y .

c.1.4 Homomorphic encryption

To perform arithmetic on encrypted data, homomorphic encryption was introduced [41]. It provides for specific functions that can be ap-

plied to encrypted data, while keeping it encrypted. An example function is addition of plaintexts by multiplying the ciphertexts.

$$\begin{aligned}
 h &= g^x \bmod n \\
 c_1 &= g^y \bmod n \\
 c_1' &= g^{y'} \bmod n \\
 s &= h^y = h^{x*y} \bmod n \\
 s' &= h^{y'} = h^{x*y'} \bmod n \\
 c_2 &= m*s \bmod n \\
 c_2' &= m'*s' \bmod n \\
 m*m' &= \frac{c_2 * c_2'}{c_1^x * c_1'^x} = c_2 * c_2' * c_1^{-x} * c_1'^{-x} = c_2' * c_2 * g^{y'-x} * g^{y-x} \\
 &= m'*m * g^{x*y} * g^{-x*y} * g^{x*y'} * g^{-x*y'} (= m*m') \bmod n
 \end{aligned}$$

Other arithmetic functions can be found (and used) with other encryption types to fulfill the homomorphic encryption property, for some arithmetic function \circ and ϕ :

$$\text{enc}(x) \circ \text{enc}(y) = \text{enc}(x \phi y)$$

C.1.5 Non-malleable encryption

The above public key encryption system is vulnerable to a related-encryption attack [31]. If an attacker is able to, for example, triple the value of c_2 before the receiver receives the ciphertext, the message looks legit, but the content of m was tripled. Consider this being the rent you pay your landlord.

$$\begin{aligned}
 c'_2 &= 3 * c_2 \bmod n \\
 m' &= \frac{3 * c_2}{c_1^x} = 3 * c_2 * c_1^{-x} = 3 * c_2 * g^{y-x} = 3 * m * g^{x*y} * g^{-x*y} (= 3*m) \bmod n
 \end{aligned}$$

The concept of non-malleability covers the encryption types that defend the sender against these related-encryption attacks.

C.1.6 Threshold encryption

Threshold encryption can be used to distribute a key among a set of trustees [95]. This can for example be used to secure the recipe a brand of coke. Some enhanced version of threshold cryptography can be used to distribute the decryption capabilities instead of the key to decrypt messages. To explain how this works, assume a linear function $y = Ax + B$ of which both A and B are unknown and kept secret and B is the actual recipe of cola. It is possible to publish *one* point (x,y) on this linear line without anybody being able to find

A or B, but if two points on this linear line would be public, everybody can find A and B and thereby the recipe of cola. Similarly, if the function would be quadratic, $y = Ax^2 + Bx + C$, and two points (x,y) would be public, nobody is able to find A, B or C, but an additional third point (x,y) would reveal A, B and C and the cola recipe. In general, for a n -order function, $n + 1$ points are needed to reveal the secret parameters and with n points or less, finding any parameter is impossible, even for unlimited advisories (unconditionally or information-theoretic hiding). Every trustee would be handed over a point on the line drawn by the function, and with $n + 1$ trustees, they can recovery the secret.

Besides hiding a private key, some cryptographic systems such as ElGamal [35] can be used to decryption without reconstruction of the key [25], assuming all trustees work along. Every trustee gets a part of the key, k_i . Example found in Jonker et al. [60]:

$$\begin{aligned} k &= \forall_i k_i \\ y &= g^k \\ \text{enc}(m) &= (c_1, c_2) = (g^r, y^r m) = (g^r, (g^k)^r m) \\ \text{dec}(c_1, c_2) &= m = \frac{c_2}{c_1^k} = \frac{c_2}{c_1^{\prod k_i}} = \frac{(g^k)^r m}{(g^r)^{\prod k_i}} (= m) \end{aligned}$$

c.1.7 Commitments

Commitments can be used to first show commitment to some unknown value which you, in a later phase reveal. Commitments are meant to both hide the value committed to, and bind to the value committed to. Commitments can be either information theoretic hiding and computationally binding or vice versa. An example of a information-theoretic hiding and computationally binding commitment scheme for a bit b using random u is called the Pedersen commitment [80]:

$$\text{commit}(u,b) = g^u h^b$$

A commitment scheme that offers information-theoretic binding and computationally binding properties could be constructed as follows:

$$\text{commit}(u,b) = (g^u, h^{u+b})$$

To open up the commitment (*reveal* the commitment), one publishes both u and b and everybody can check that the commitment outputs the same result.

C.2 ZERO-KNOWLEDGE PROOFS

Zero-knowledge proofs can be used to proof statements about a secret, of which the proof does not provide any information *on* the

secret. An example could be a “Where is Waldo picture”¹ by which the location of Waldo is the secret and the zero-knowledge proof of knowledge of where Waldo is, shouldn’t reveal the location of Waldo. Imagine a big sheet of paper covering the “Where is Waldo picture” multiple times, cut a hole in the sheet and secretly lay that hole precisely over Waldo, you can then show everybody you have knowledge of where Waldo is, without providing any help to anyone looking for Waldo in the original picture. Because we all know what Waldo looks like, the proof did not give any additional information. Even more interesting, the proof is not exchangeable, in the sense that if you took a picture of the sheet of paper with Waldo sticking out, anyone could have made that picture because we can find pictures of Waldo on the internet already. Thus, the proof itself is not exchangeable: with hindsight, the verifier could have made the proof himself. In mathematics, this can for example be expressed as a run of Schnorr’s identification protocol [94]:

A chooses x privately and distributes g and $y = g^x$ publicly

A chooses nonce N_a random and computes $a = g^{N_a}$

1. A \rightarrow B : a

B chooses nonce N_b random

2. B \rightarrow A : N_b

A computes $r = N_a + N_b * x$

3. A \rightarrow B : r

B checks that $g^r = a * y^{N_b}$

If succeeded, B knows that A knows x

But didn’t learn anything about x

In general, every NP-statement can be proven with a zero-knowledge proof [45]. In voting systems this can, for example, be used to prove that a shuffle and re-encryption of messages happened correctly (no new messages inserted, altered or deleted).

Fiat and Shamir designed an algorithm to turn every zero-knowledge protocol into a non-interactive zero-knowledge proof by removing agent B in the protocol and replacing his job of providing the random nonce by a hashing algorithm with input the messages from agent A to B and output the random nonce from agent B to agent A (including the generator g and public key y). The proof is the set of messages normally sent between the agents A and B put together. [40]

Since every type of NP-statement can be proven with a zero-knowledge proof, even statements like “I am Alice OR I voted for Party A” can be proven with zero-knowledge. This could for example be used by election authorities to provide Alice with a receipt of her vote, which

¹ Example from Berry Schoenmakers

she can't use to prove to others how she voted, because she could have made this receipt herself (being the first part of the OR-clause). These types of zero-knowledge proofs are called designated-verifier non-interactive zero-knowledge proofs [54].

C.3 CRYPTOGRAPHIC SYSTEMS WITHIN E-VOTING

The above cryptographic primitives come together in cryptographic systems which by themselves provide privacy and verifiability to a voter. The next paragraphs explain the concepts behind "two agents", "double envelope", "homomorphic tallying", "mix-nets" and "pollsterless schemes" respectively.

C.3.1 *Two agents*

To split the eligibility test from the vote counting among the voting authorities, Chaum invented the "Blind signature". This allows for an encrypted vote to be sent to an eligibility tester for a signature, after which the voter can remove the encryption and his eligibility information and send it to a vote counting authority (anonymously), still having the signature provided by the eligibility tester. The following shows how the concept works. First a message is blinded by agent B, then agent A provides a digital signature on that blinded message, after which agent B removes the blinding, leaving a plaintext message, signed by agent A without agent A knowing the content.

$$\text{deblind}_b(\text{sign}_{\text{sk}(a)}(\text{blind}_b(m))) = \text{sign}_{\text{sk}(a)}(m)$$

Blind signatures thus provide a means to have a message signed without the signing authority to know the content, this could be seen as an envelop with carbon paper on it, to sign the inside of the envelop [60]. Figure C.1 explains the two agent system visually.

C.3.2 *Double envelope*

An e-voting scheme is rather related to the physical matter of postal voting is called "two envelopes" or "double envelope". In the physical case, two envelopes are used, of which the outer envelope holds both an identification method (i.e. a copy of a passport) and the inner envelope. The inner envelope contains the vote of that particular voter. Before tallying, the outer envelope is opened, the credentials are checked and the outer envelope and the credentials are then thrown away while the inner envelope is (still sealed) thrown into a ballot box. After all postal votes have been stripped of their credentials, the ballot box is shuffled, the envelopes checked for the containment of only one vote, and then the tallying phase starts. This method is employed in countries allowing for postal voting, furthermore, a digital

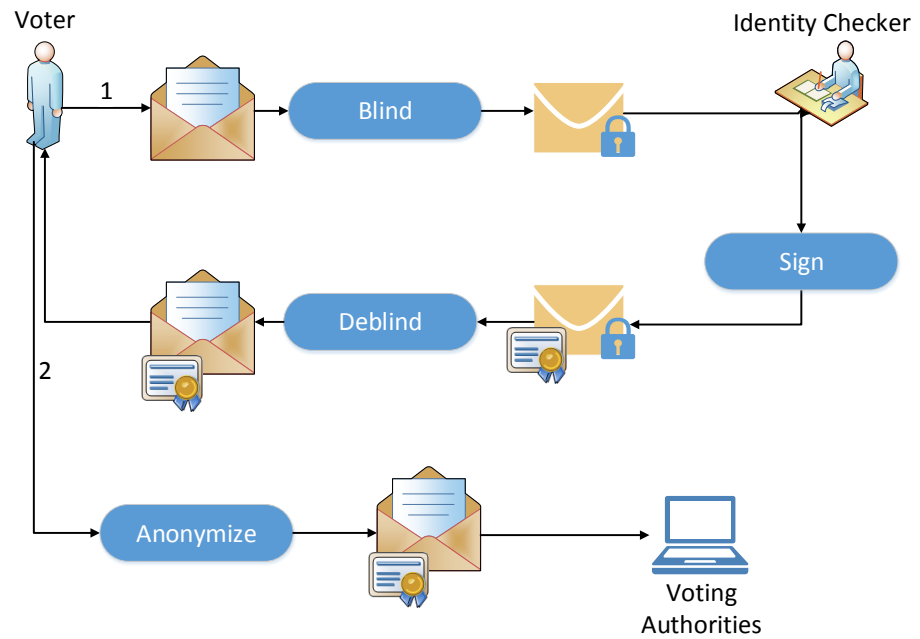


Figure C.1: Two agents

implementation is also used in Estonia [109]. Digitally, this can be represented by having the voter encrypt the vote using the public key of the voting authorities, and then sign that encrypted vote using their private signing key. Figure C.2 shows this: the double envelopes are split into a set of identities (E-voters) and votes (E-votes). The private key only comes into play when the identities are split from the votes.

$$\text{vote} = \text{sign}_{\text{sk}(a)}(\text{encrypt}_{\text{pk}(b)}(m))$$

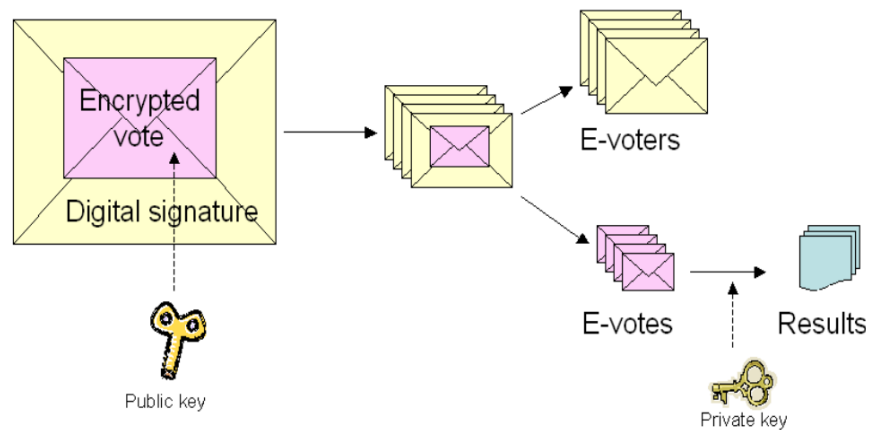


Figure C.2: Double envelope voting system [109, p8]

C.3.3 Homomorphic tallying

The properties of homomorphism and joint decryption can be used in e-voting schemes with “Homomorphic tallying” [25]. These schemes rely on zero-knowledge proofs that prove the existence of exactly one vote in an encrypted vote. The votes are received by a voting authority and checked for a valid zero-knowledge proof. All votes together are added up (still being encrypted) after which the result is decrypted, Figure C.3 shows this process. The decryption could be done by a threshold number of election authorities, thereby ensuring that individual votes cannot be opened with at least one honest election authority.

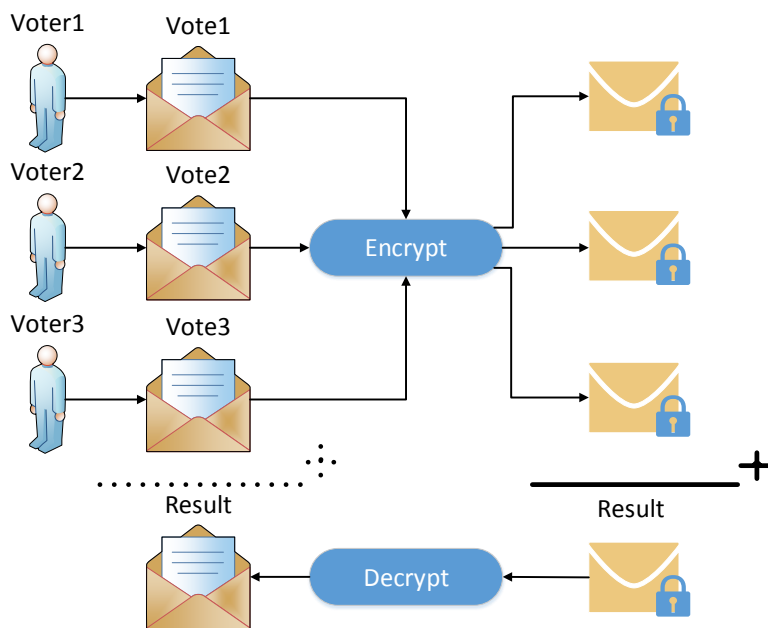


Figure C.3: Homomorphic tallying

C.3.4 Mix-nets

Another method of providing anonymity and verifiability to e-voting schemes is provided by the use of mix-nets [55]. Mix-nets consist of a network of computers who partly decrypt / re-encrypt the votes of voters, shuffle them, and send them to the next computer in the network of the mix-net. To prove that they shuffled the set of votes correctly, they provide zero-knowledge proofs of correct shuffling, furthermore, they also provide zero-knowledge proofs of correct partial decryption / re-encryption. These proofs together provide a verifiable link for the voter to “follow” his vote to the counting authorities.

Figure C.4² visually depicts mix-nets, but without addressing the zero-knowledge proofs.

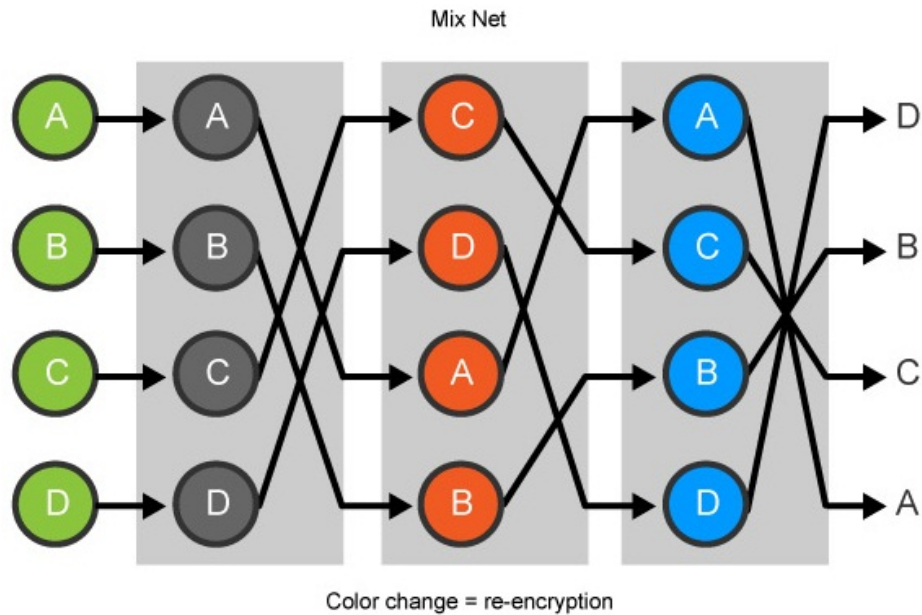


Figure C.4: Mix-nets

C.3.5 Pollsterless schemes

In 2003, Malkhi et al. [70] reviewed the problems with existing cryptographic systems used for e-voting, and found a continues problem throughout all of these schemes: the end user's computer. They found that the voter always needs to trust the application run on their computer for both encrypting and verifying their vote. The software used for these actions they call a "pollster". Besides the software (and potentially hardware) the user needs to trust, he also needs to trust the correctness of some "non-trivial" math. The foregoing paragraphs described, for example, the zero-knowledge proofs to prove correctness of the reshuffling of votes by a mix-net computer. In theory, this sounds like a good idea, but in practice the voter still relies on trusting software that checks these proofs because the voter is himself unable to perform such hard math.

Malkhi et al. propose pollsterless schemes that eliminate the use of such software without relying on any additional knowledge by the voter. An example of such a scheme is the one proposed by Spycher et al. [99] for the Norwegian e-voting scheme. In this scheme, each voter receives a personal list of return codes for every candidate and every party before the elections by postal mail. After the voter casts a vote, he receives an SMS with the return code, which he can check in the list he received earlier.

² Source: <http://www.wombat-voting.com/wp-content/uploads/2011/05/mixnet.jpg>

A similar system was proposed by Storer et al. [102] and is shown in Figure C.5.

In a research performed by Oostveen and Van den Besselaar [76] in 2004 (before the controversy on voting methods in the Netherlands started), it is shown that the public has little understanding of the security properties of voting schemes. In several proposals [102, 100, 101], there are improvements on excluding a pollster from a voting schemes. Storer et al. [102] performed a user acceptance test with regards to the mCESG pollsterless voting scheme [100, 101]. The results of the user acceptance test showed that, among other results, voters consider this scheme to provide high mobility, but they also expressed some concerns with the security of the internal working of the system. The researchers suggest *“The concerns raised by the participants suggest that the implementation of voting schemes will need to be accompanied by explanation as to the reasons voters should accept voting as secure”*. This could however be considered as walking away from the principal of having a pollsterless scheme in which the voter doesn't need to trust anything at all. Furthermore, questions could be raised on how to trust that the return codes are correctly formulated and could not be forged.

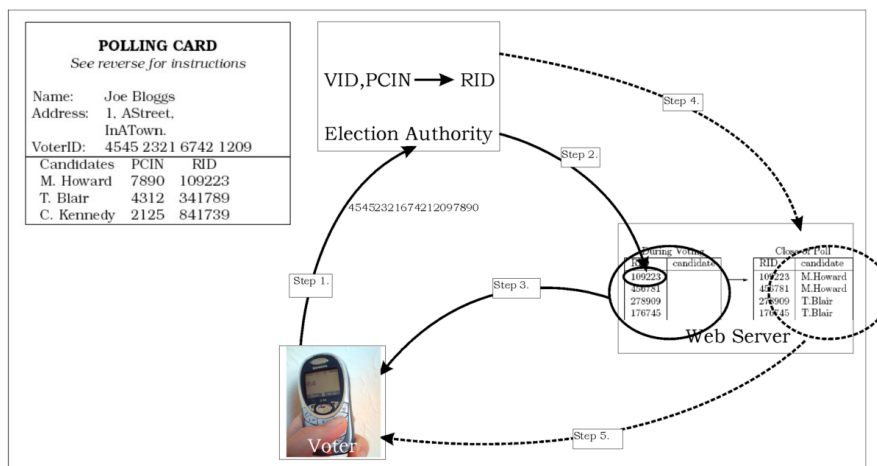


Figure C.5: Pollsterless scheme mCESG [102, Figure 1]

C.4 CONCLUSION

In this Section, the cryptographic systems for e-voting schemes were presented by first introducing the necessary cryptography very briefly, followed by the notion of zero-knowledge and the actual cryptographic systems. The introduced cryptographic building block (two agents, double envelope, homomorphic tallying, mix-nets and pollsterless schemes) have all been used in theoretical and sometimes even practical e-voting schemes.

Through time, the cryptographic systems for e-voting have changed. While at first, the main focus in literature was on both the double envelope and two agents, later the focus became mixing and homomorphic tallying (or a combination of the two) with lately the introduction of pollsterless schemes as an addition to the other systems. In practice, a combination of mixing(, threshold decryption) and pollsterless verification methods is seen the most (see [Chapter 4](#)).

EXPERT INTERVIEWS FOR VALIDATION

The results of the expert interviews validating this framework can be found in this appendix. Section D.1 presents the results of the interview with a security professional from the services industry, section D.2 presents the results of the interview with a professional from the Dutch voting practice and section D.3 presents the results of the interview with a professional in the e-voting research discipline.

D.1 IT SECURITY PROFESSIONAL

This verification interview was held with an IT Security professional who works as a consultant in the IT Security industry and has a wide experience in the field of risk management.

In general, the framework is considered useful and usable, given some improvement points. The most important feedback was due to the fact that the framework has not been fully applied to a real life case. Especially since the first step of the framework is found to be very labor intensive and could benefit from some additional pointers on how to do this, it is unclear if the framework is effective. Care must go into determining the feasibility of efficiently discovering all relevant properties of the assets, as well as the feasibility of an efficient filtering method to sieve the model checkers results into meaningful attacks. The framework does provide for a solid method of objectifying the risks that arise from using such an e-voting scheme, as well as to identify how to effectively adjust e-voting schemes to better fit the security requirements a country has.

The detailed comments from the interview are summarized as bullet points in different paragraphs, these bullets are in random order.

USEFULNESS

- It is unclear how to determine if all properties were identified in the find-step. Each of these properties influences the (number of) found attack vectors. This influences usefulness of the framework.
- It is unclear if the model checker can find all (realistic) attack vectors. A partial mitigation strategy is to involve experts in the round table session to identify attacks in the visual workflow already.

USABILITY

- The use of the model checker is expected to be the most labor intensive job, but is not performed in the case study of this thesis. It is unclear how many results will come from the model checker and how easy it is to analyze and filter them before going to quantify-step.
- Without the demonstration of the model checker, it is unclear if any filtering on the properties should be done to get a limited list of attack vectors.

CAN WE START USING IT RIGHT NOW?

- Since no complete case study has been performed, it is hard to convince people to start using this framework. They can not “play” around with the results.
- Politics is probably not ready for such a framework, they are more focused on emotional than rational arguments to discuss such matters.

IMPROVEMENT POINTS

- Time investment should not be a separate attacker effort, because a determined attacker does not lack time. Time investments do influence financial investments and should be taken into account, though, but not as a key attacker effort.
- In the current framework, cost functions can not easily be compared. Not only because the input and output variables differ, but also because it is unclear how to indicate the superior function if these functions cross.
- It is not very realistic to assume a game theoretic attacker that always takes the “cheapest” attacker path. E-voting schemes therefore may be valued much more insecure than in a realistic scenario.
- Spread of costs is not taken into account. With so much estimations, the cost function could benefit from ranges.
- The first two steps are more elaborate than the last three. Especially the last step could be more elaborate to support users of the framework.
- The framework does not really answer the research question (it is not the answer of operationalizing the safeguards), but it helps operationalizing them.

STRONG POINTS

- The framework has a modular approach, the different steps can be interchanged by new methods (such as finding attack vectors). This allows other countries to use the framework as well, by only changing the categorize-step.
- The quantify and categorize steps make for an objective comparison of attacks.
- The quantification of attacks in general could be interesting in other areas as well.
- The framework does not take stand in any political argument, but objectively shows what certain decisions cause.
- The framework does show what decisions are important to take into account when deciding on acquiring such an e-voting scheme. The framework points to those decisions and gives objective feedback to make rational choices.

D.2 DUTCH VOTING PRACTICE PROFESSIONAL

This verification interview was held with a professional from the Dutch voting practice.

In general, the framework is perceived as interesting for the Dutch voting practice. There is appreciation for the modularity of the different steps of the protocol: a similar framework can e.g. be used to identify risks and attacker effort in the voting computer case, which has a much higher priority on short notice. The interviewee recognizes the fact that the framework *could* be used for decision making of whether to adopt e-voting, but doesn't believe it will. According to the interviewee, politicians will base their decision on emotional rather than rational choices with regards to these risks, where after the Dutch Ministry of Interior is asked to implement it. In the latter phase, the interviewee sees added value from the use of this framework.

The framework is considered useful for both judgments on what e-voting scheme to adopt (in the case the Netherlands will do so); and to validate the effects certain policies will have on the risks concerned with e-voting (e.g. the number of voting days influencing the attacker effort needed to gain a seat in parliament). The usability of the framework is considered high, all steps in the framework are considered to assume a minimal amount of time. The find-step will take a few workshops to get to the graphical version of the workflow, which can then be implemented in the model checker by an external party. Sieving attack vectors before going to the quantify step is not considered to take much time, especially since clear processes define what can

and can't be done on the level of asset manipulation. The biggest challenge for this framework, according to the interviewee, is to get this on the table with policy makers and officials from the Ministry.

The detailed comments from the interview are summarized as bullet points in different paragraphs, these bullets are in random order.

USEFULNESS

- On short notice, the framework could be used for determining risks within voting computer cases with slight modifications to the framework.
- Only after politics has decided to adopt e-voting technology, this framework could be of use. According to the interviewee, the decision of whether or not to adopt e-voting will not be taken with such rationals as provided by this framework.
- The most useful elements according to the interviewee are to decide what e-voting scheme can be used as a blueprint for the Dutch case; and after that the comparison with the baseline set by politics can be made to check if the final version of the e-voting scheme complies with the wishes of politics. Besides these two elements, the possibility to objectively calculate the increased or decreased risk from certain possible changes by politics is considered very useful.

USABILITY

- To perform the find-step, you need to have a round table session (workshop) with people from different disciplines and backgrounds to successfully identify the assets and properties. The interviewee doesn't think this will be a problem and should be done in only a limited number of days. Translating the graphical model into the source code needed for the model checker will then be outsourced to a third party. The interviewee suspects that the third party should already be involved in the earlier phase of determining those assets and properties to ensure a proper model check.
- Sieving attack vectors that go on to the quantify-step can be done quite quickly, due to the underlying processes that aid the current voting process. Depending on the number of attack vectors found by the model checker, the interviewee doesn't expect this to be a lot of work.
- The biggest challenge for this framework, according to the interviewee, is to get this framework on the table of policy makers. They first need to know about this framework, and after that they also need to adopt it.

CAN WE START USING IT RIGHT NOW?

- Politics isn't ready for any risk calculations for e-voting yet, they first have to decide on emotional arguments whether or not to adopt e-voting after which this framework becomes useful.
- Specialists are needed to first get the visual representation of the workflow; thereafter additional specialists are needed to translate this representation into the code for the model checker. Depending on the expertise of the third party, the framework could be applied already.

IMPROVEMENT POINTS

- Politicians do not want precise figures and statistics to determine whether or not to adopt such systems, they base their decision on emotional arguments only.

STRONG POINTS

- The modularity of the system is perceived to be a strong point.
- The translation of safeguards to types of attacks is very useful.
- The realistic risks really benefit from the adaptiveness of the attacker model.

D.3 E-VOTING RESEARCH PROFESSIONAL

This verification interview was held with a professional from the e-voting research discipline.

In general, the research professional is enthusiastic about the fact that a framework is proposed that can identify and quantify real life risks. The interviewee himself is of the opinion that such additional research is mostly not taken into account when deciding about e-voting. Besides such an analysis, the researcher strongly believes that such a framework can be used as a final step after the protocol, the cryptography and the actual implementation have been researched and a penetration test was performed. This framework is a first step to accomplish a more thorough analysis of e-voting schemes, though the framework should still evolve.

The first step of the framework is thought to be relatively hard. It could be that the visual representation of the workflow is hard to construct if not all experts involved are capable of understanding such schemes. After having identified all the attack vectors, the sieving of those could be very problematic since the model checker could output an extremely large list of attack vectors. That's currently unknown. Furthermore, experts should be involved in all the steps to ensure that there is nothing missed out on: in the find-step, experts

could find additional attack vectors that the model checker missed; the quantify-step also benefits from experts checking whether all estimations are used correctly and if the attacks are correctly built.

The detailed comments from the interview are summarized as bullet points in different paragraphs, these bullets are in random order.

USEFULNESS

- A systematic approach for such risk analysis in general is a good idea, details can be up for discussion. Though it is beneficial for the e-voting debate that there is an additional model that specifically aims at e-voting schemes and addresses realistic attack scenarios. It should be an additional model, next to thorough protocol, cryptography and implementation analyses and a penetration test.

USABILITY

- It is very important to still incorporate experts to analyze the results of applying the framework to an actual case. Not only to check whether all attacks have been found, but also to check whether they have all been sieved correctly. The model checker will only find attacks that fit the model, therefore the framework benefits from including a professional. Not only does the outcome benefit from this professional, the framework could also be enhanced by the view of the professional.
- It is unclear if the framework allows for more complex e-voting schemes that involve handling out-sourced system components; threshold cryptography versus single point central servers; etc.

CAN WE START USING IT RIGHT NOW?

- To get to the initial talk book of the scheme, different experts should be involved. It could be that some of them are incapable of understanding such high-level talk books. Furthermore it could be that they can't ensure that all their knowledge ends up in the talk book due to the way it is structured. Special care should be taken to enhance the round table session so that all expertise goes into the talk book.

IMPROVEMENT POINTS

- It is unclear how to actually sieve the results of the first step correctly. This is a major point in working with this model.
- The results of the quantification-step could deviate from the actual situation due to the use of estimations. Especially if thresholds are used (e.g. in the quantification or the comparing-step) this could lead to undesirable situations.

- Maintenance of this framework can be hard. It is unclear if the framework is resilient to working with currently uninvented schemes.
- The more the framework is used, the better it gets. Therefore a more thorough case study would have been beneficial for the framework.

STRONG POINTS

- The framework is very systematic, it can be understood by lots of different disciplines and it provides for a good starting point of taking such risks to e-voting schemes into account.
- The modularity of the framework enhances the fact that it can easily be used in the future if new methods for e.g. finding attack vectors are found.
- The realistic risks really benefit from the adaptiveness of the attacker model.

BIBLIOGRAPHY

- [1] Lillian Ablon, Martin C Libicki, and Andrea A Golay. *Markets for Cybercrime Tools and Stolen Data: Hackers' Bazaar*. Rand Corporation, 2014.
- [2] Gail-Joon Ahn, William Enck, and Dongwan Shin. Guest editors' introduction: Special issue on security and privacy in mobile platforms. *IEEE Transactions on Dependable and Secure Computing*, 11(3):209–210, 2014.
- [3] Michael Backes and Birgit Pfitzmann. Symmetric encryption in a simulatable dolev-yao style cryptographic library. In *Computer Security Foundations Workshop, 2004. Proceedings. 17th IEEE*, pages 204–218. IEEE, 2004.
- [4] Colin Barry and Ian Brightwell. Technology assisted voting. Technical report, 2011.
- [5] Josh Benaloh. Simple verifiable elections. In *Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop 2006 on Electronic Voting Technology Workshop*, pages 5–5, 2006.
- [6] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique cryptanalysis of the full aes. In *Advances in Cryptology—ASIACRYPT 2011*, pages 344–371. Springer, 2011.
- [7] Ahto Buldas, Peeter Laud, Jaan Priisalu, Märt Saarepera, and Jan Willemson. Rational choice of security measures via multi-parameter attack trees. In *Critical Information Infrastructures Security*, pages 235–248. Springer, 2006.
- [8] Mike Burmester and Emmanouil Magkos. E-ELECTIONS IN THE NEW ERA. (January 2000):1–14, 2002.
- [9] Juan Caballero, Chris Grier, Christian Kreibich, and Vern Paxson. Measuring pay-per-install: The commoditization of malware distribution. In *USENIX Security Symposium*, 2011.
- [10] David Chaum. Blind signatures for untraceable payments. In *Crypto*, volume 82, pages 199–203, 1982.
- [11] David Chaum. Elections with unconditionally-secret ballots and disruption equivalent to breaking rsa. In *Advances in Cryptology—EUROCRYPT'88*, pages 177–182. Springer, 1988.
- [12] David Chaum. Surevote: technical overview. In *Proceedings of the workshop on trustworthy elections (WOTE'01)*, 2001.

- [13] David Chaum. Secret-ballot receipts: True voter-verifiable elections. *Security & Privacy, IEEE*, 2(1):38–47, 2004.
- [14] David L Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981.
- [15] Thomas M Chen and Saeed Abu-Nimeh. Lessons from stuxnet. *Computer*, 44(4):91–93, 2011.
- [16] Michel Chevallier, Michel Warynski, and Alain Sandoz. Success factors of geneva’s e-voting system. In *Proceedings of the 6th European Conference on e-Government*, pages 77–85. Academic Conferences Limited, 2006.
- [17] Benoit Chevallier-Mames, Pierre-Alain Fouque, David Pointcheval, and Jacques Traoré. On some incompatible properties of voting schemes. In *In IAVoSS Workshop On Trustworthy Elections, WOTE’06*. Citeseer, 2006.
- [18] Alessandro Cimatti, Edmund Clarke, Enrico Giunchiglia, Fausto Giunchiglia, Marco Pistore, Marco Roveri, Roberto Sebastiani, and Armando Tacchella. Nusmv 2: An opensource tool for symbolic model checking. In *Computer Aided Verification*, pages 359–364. Springer, 2002.
- [19] Sijbren Cnossen. *Hoe beschaafd is Nederland?* Sdu Uitgevers, 2009.
- [20] Commission on Security and Cooperation in Europe. Document of the copenhagen meeting of the conference on the human dimension of the csce, 1990.
- [21] Véronique Cortier and Ben Smyth. Attacking and fixing helios: An analysis of ballot secrecy. *Journal of Computer Security*, 21(1): 89–148, 2013.
- [22] Council of Europe. European convention on human rights, 1950.
- [23] Council of Europe. *Legal, operational and technical standards for e-voting: recommendation Rec (2004) 11 adopted by the Committee of Ministers of the Council of Europe on 30 September 2004 and explanatory memorandum*. Council of Europe, 2004.
- [24] Council of Europe. Certification of e-voting systems: Guidelines for developing processes that confirm compliance with prescribed requirements and standards, 2011. [Online; accessed 25-March-2014].

- [25] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. *European transactions on Telecommunications*, 8(5):481–490, 1997.
- [26] Joan Daemen and Vincent Rijmen. Aes proposal: Rijndael. In *First Advanced Encryption Standard (AES) Conference*, 1998.
- [27] Stephanie Delaune, Steve Kremer, and Mark Ryan. Coercion-resistance and receipt-freeness in electronic voting. In *Computer Security Foundations Workshop, 2006. 19th IEEE*, pages 12–pp. IEEE, 2006.
- [28] Stéphanie Delaune, Steve Kremer, and Mark Ryan. Verifying privacy-type properties of electronic voting protocols. *Journal of Computer Security*, 17(4):435—487, 2009.
- [29] Denise Demirel, Hugo Jonker, and Melanie Volkamer. Random block verification: Improving the norwegian electoral mix-net. In *Electronic Voting*, pages 65–78, 2012.
- [30] Danny Dolev and Andrew Yao. On the security of public key protocols. *Information Theory, IEEE Transactions on*, 29(2):198–208, 1983.
- [31] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM review*, 45(4):727–784, 2003.
- [32] Jannik Dreier, Pascal Lafourcade, and Yassine Lakhnech. Vote-independence: A powerful privacy notion for voting protocols. In *Foundations and Practice of Security*, pages 164–180. Springer, 2011.
- [33] Jannik Dreier, Pascal Lafourcade, and Yassine Lakhnech. A formal taxonomy of privacy in voting protocols. In *Communications (ICC), 2012 IEEE International Conference on*, pages 6710–6715. IEEE, 2012.
- [34] Andreas Ehringfeld, Larissa Naber, Thomas Grechenig, Robert Krimmer, Markus Traxl, and Gerald Fischer. Analysis of recommendation rec (2004) 11 based on the experiences of specific attacks against the first legally binding implementation of e-voting in austria. In *Electronic Voting*, pages 225–237, 2010.
- [35] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *Information Theory, IEEE Transactions on*, 31(4):469–472, 1985.
- [36] Aleksander Essex, Jeremy Clark, Urs Hengartner, and Carlisle Adams. Eperio: Mitigating technical complexity in cryptographic election verification. *IACR Cryptology ePrint Archive*, 2012:178, 2012.

- [37] Saghar Estehghari and Yvo Desmedt. Exploiting the client vulnerabilities in internet e-voting systems: Hacking helios 2.0 as an example. In *Proc. 2010 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE)(Aug. 2010)*, 2010.
- [38] JB i Esteve, Ben Goldsmith, and John Turner. International Experience with E-Voting - Norwegian E-Vote Project. (June), 2012.
- [39] JB i Esteve, Ben Goldsmith, and John Turner. Compliance with International Standards - Norwegian E-Vote Project. 2012.
- [40] Amos Fiat and Adi Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Advances in Cryptology—CRYPTO’86*, pages 186–194. Springer, 1987.
- [41] Caroline Fontaine and Fabien Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP Journal on Information Security*, 2007, 2007.
- [42] Bartek Gedrojc, Matthieu Hueck, Hans Hoogstraten, Mark Koek, and Sjoerd Resink. Advisering toelaatbaarheid internetstemvoorziening waterschappen. Technical report, Technical report, Fox-IT BV, 2008.
- [43] Rosario Gennaro. Achieving independence efficiently and securely. In *Proceedings of the fourteenth annual ACM symposium on Principles of distributed computing*, pages 130–136. ACM, 1995.
- [44] Kristian Gjsteen. Analysis of an internet voting protocol. *IACR Cryptology ePrint Archive*, 2010:380, 2010.
- [45] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to prove all np statements in zero-knowledge and a methodology of cryptographic protocol design. In *Advances in Cryptology—CRYPTO’86*, pages 171–185. Springer, 1987.
- [46] Max Goncharov. Russian underground 101. *Trend Micro Incorporated Research Paper*, 2012.
- [47] Rop Gonggrijp, Willem-Jan Hengeveld, Eelco Hotting, Sebastian Schmidt, and Frederik Weidemann. RIES-Rijnland Internet Election System: A cursory Study of Published Source Code. In *E-Voting and Identity*, pages 157–171. Springer, 2009.
- [48] Gurchetan S Grewal, Mark D Ryan, Sergiu Bursuc, and Peter YA Ryan. Caveat coercitor: coercion-evidence in electronic voting. In *IEEE Security and Privacy Symposium*, 2013.
- [49] Sven Heiberg, Peeter Laud, and Jan Willemsen. The application of i-voting for estonian parliamentary elections of 2011. In *E-Voting and Identity*, pages 208–223. Springer, 2012.

- [50] LMLHA Hermans and MJW van Twist. Stemmachines: Een verweesd dossier. 2007.
- [51] Engelbert Hubbers, Bart Jacobs, Berry Schoenmakers, Henk van Tilborg, and Benne de Weger. Description and analysis of the ries internet voting system. *Report of the Eindhoven Institute for the protection of systems and information*, 2008.
- [52] Eric M Hutchins, Michael J Cloppert, and Rohan M Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1:80, 2011.
- [53] Bart Jacobs and Wolter Pieters. Electronic voting in the netherlands: from early adoption to early abolishment. In *Foundations of Security Analysis and Design V*, pages 121–144. Springer, 2009.
- [54] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated verifier proofs and their applications. In *Advances in Cryptology—EUROCRYPT’96*, pages 143–154. Springer, 1996.
- [55] Markus Jakobsson, Ari Juels, and Ronald L Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *USENIX security symposium*, pages 339–353. San Francisco, USA, 2002.
- [56] Craig James. Discussion paper: internet voting. Technical report, 2011.
- [57] Hugo Jonker and Jun Pang. Bulletin boards in voting systems: Modelling and measuring privacy. In *Availability, Reliability and Security (ARES), 2011 Sixth International Conference on*, pages 294–300. IEEE, 2011.
- [58] Hugo Jonker, Sjouke Mauw, and Jun Pang. Measuring voter-controlled privacy. In *Availability, Reliability and Security, 2009. ARES’09. International Conference on*, pages 289–298. IEEE, 2009.
- [59] Hugo Jonker, Sjouke Mauw, and Jun Pang. A formal framework for quantifying voter-controlled privacy. *Journal of Algorithms*, 64(2):89–105, 2009.
- [60] Hugo Jonker, Sjouke Mauw, and Jun Pang. Privacy and verifiability in voting systems: Methods, developments and trends. *Computer Science Review*, 10:1–30, 2013.
- [61] Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-resistant electronic elections. In *Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, pages 61–70. ACM, 2005. ISBN 1595932283.

- [62] Chris Karlof, Naveen Sastry, and David Wagner. Cryptographic voting protocols: A systems perspective. In *USENIX Security Symposium*, volume 12, page 39, 2005.
- [63] Reto E Koenig, Philipp Locher, and Rolf Haenni. Attacking the verification code mechanism in the norwegian internet voting system. In *E-Voting and Identify*, pages 76–92. Springer, 2013.
- [64] Steve Kremer, Mark Ryan, and Ben Smyth. *Election verifiability in electronic voting protocols*. Springer, 2010.
- [65] Kummeling. Reactie Kiesraad op het rapport Stemmen met Vertrouwen van de Adviescommissie Korthals Altes.pdf, 2007.
- [66] R Kusters and Tomasz Truderung. An epistemic approach to coercion-resistance for electronic voting protocols. In *Security and Privacy, 2009 30th IEEE Symposium on*, pages 251–266. IEEE, 2009.
- [67] Lucie Langer, Hugo Jonker, and Wolter Pieters. Anonymity and verifiability in voting: understanding (un) linkability. In *Information and Communications Security*, pages 296–310. Springer, 2010.
- [68] Arjen K Lenstra, Hendrik W Lenstra Jr, Mark S Manasse, and John M Pollard. The number field sieve. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 564–572. ACM, 1990.
- [69] Ülle Madise and Tarvi Martens. E-voting in estonia 2005. the first practice of country-wide binding internet voting in the world. *Electronic voting*, 86, 2006.
- [70] Dahlia Malkhi, Ofer Margo, and Elan Pavlov. E-voting without “cryptography”. In *Financial Cryptography*, pages 1–15. Springer, 2003.
- [71] Derek Manky. Cybercrime as a service: a very modern business. *Computer Fraud & Security*, 2013(6):9–13, 2013.
- [72] Tarvi Martens. Internet voting in estonia. Technical report, Technical report., 2010.
- [73] Sjouke Mauw and Martijn Oostdijk. Foundations of attack trees. In *Information Security and Cryptology-ICISC 2005*, pages 186–198. Springer, 2006.
- [74] Charles Miller. The legitimate vulnerability market: the secretive world of o-day exploit sales. In *WEIS*, 2007.
- [75] Tal Moran and Moni Naor. Receipt-free universally-verifiable voting with everlasting privacy. In *Advances in Cryptology-CRYPTO 2006*, pages 373–392. Springer, 2006.

- [76] Anne-Marie Oostveen and Peter Van den Besselaar. Ask no questions and be told no lies: Security of computer-based voting systems, users' trust and perceptions. In *EICAR 2004 Conference CD-Rom, UE Gattiker, ed., Copenhagen: EICAR EV*, pages 1–18, 2004.
- [77] OSCE. THE NETHERLANDS Early Parliamentary Elections. (March), 2010.
- [78] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in cryptology—EUROCRYPT—99*, pages 223–238. Springer, 1999.
- [79] Joseph Pamula, Sushil Jajodia, Paul Ammann, and Vipin Swarup. A weakest-adversary security metric for network configuration security analysis. In *Proceedings of the 2nd ACM workshop on Quality of protection*, pages 31–38. ACM, 2006.
- [80] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology—CRYPTO—91*, pages 129–140. Springer, 1992.
- [81] W Pieters and MJ Becker. Ethics of e-voting: An essay on requirements and values in internet elections. 2005.
- [82] Wolter Pieters. What proof do we prefer? Variants of verifiability in voting. 2006.
- [83] Wolter Pieters. *La volonté machinale: understanding the electronic voting controversy*. PhD thesis, Radboud University of Nijmegen, 2008.
- [84] Tiphaine Pinault and Pascal Courtade. E-voting at expatriates' mps elections in france. In *Electronic Voting*, pages 189–195, 2012.
- [85] J Puiggali. Universally Verifiable Efficient Re-encryption Mixnet. pages 1–7, 2010.
- [86] Vladimir J Radunovic. Ddos-available weapon of mass disruption. In *Telecommunications Forum (TELFOR), 2013 21st*, pages 5–8. IEEE, 2013.
- [87] Ronald L Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
- [88] Jean-Jacques Rousseau. *Du contrat social, ou, Principes du droit politique*. 1762.

- [89] Peter YA Ryan, David Bismark, JA Heather, Steve A Schneider, and Zhe Xia. The prêt à voter verifiable election system. *IEEE transactions on information forensics and security*, 4(4):662–673, 2009.
- [90] Vineet Saini, Qiang Duan, and Vamsi Paruchuri. Threat modeling using attack trees. *Journal of Computing Sciences in Colleges*, 23(4):124–131, 2008.
- [91] Kazue Sako and Joe Kilian. Receipt-free mix-type voting scheme. In *Advances in Cryptology—EUROCRYPT’95*, pages 393–403. Springer, 1995.
- [92] Oliver Schirokauer, Damian Weber, and Thomas Denny. Discrete logarithms: the effectiveness of the index calculus method. In *Algorithmic number theory*, pages 337–361. Springer, 1996.
- [93] Bruce Schneier. Attack trees. *Dr. Dobbs’s journal*, 24(12):21–29, 1999.
- [94] CP Schnorr. Efficient identification and signatures for smart cards. In *Advances in Cryptology—EUROCRYPT’89*, pages 688–689. Springer, 1990.
- [95] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.
- [96] Ben Smyth. *Formal verification of cryptographic protocols with automated reasoning*. PhD thesis, University of Birmingham, 2011.
- [97] Ben Smyth and David Bernhard. Ballot secrecy and ballot independence coincide. *IACR Cryptology ePrint Archive*, 2013:235, 2013.
- [98] Aditya K Sood and Richard J Enbody. Crimeware-as-a-service—a survey of commoditized crimeware in the underground market. *International Journal of Critical Infrastructure Protection*, 6(1):28–38, 2013.
- [99] Oliver Spycher, Melanie Volkamer, and Reto Koenig. Transparency and technical measures to establish trust in norwegian internet voting. In *E-Voting and Identity*, pages 19–35. Springer, 2012.
- [100] Tim Storer and Ishbel Duncan. Polsterless remote electronic voting. *Journal of E-Government*, 1(1):75–103, 2004.
- [101] Tim Storer and Ishbel Duncan. Practical remote electronic elections for the uk. In *PST*, pages 41–45. Citeseer, 2004.

- [102] Tim Storer, Linda Little, and Ishbel Duncan. An exploratory study of voter attitudes towards a pollsterless remote voting system. In *laVoSS Workshop on Trustworthy Elections (WOTE 06) Pre-Proceedings*, pages 77–86. Citeseer, 2006.
- [103] The Election Process Advisory Commission. Stemmen met vertrouwen. pages 1—109, 2007.
- [104] Alexander H Trechsel and Kristjan Vassil. Internet voting in Estonia-A comparative analysis of four elections since 2005. *Council of Europe*, 2010.
- [105] United Nations General Assembly. Universal declaration of human rights. Retrieved February, 22:2010, 1948.
- [106] United Nations General Assembly. International covenant on civil and political rights, 1966.
- [107] André van Cleeff, Trajce Dimkov, Wolter Pieters, and Roel Wieringa. Realizing security requirements with physical properties: A case study on paper voting. In *Proceedings of the International Conference on IT Convergence and Security 2011*, pages 51–67. Springer, 2012.
- [108] Venice Commission. The code of good practice in electoral matters, 2002.
- [109] Priit Vinkel. Internet Voting in Estonia. In *Information Security Technology for Applications*, pages 4—12. Springer, 2012. ISBN 9783642296147.
- [110] VKA. Onderzoek internetstemmen voor kiezers in het buitenland. Technical report, 2014.
- [111] Melanie Volkamer and Rüdiger Grimm. Multiple casts in online voting: Analyzing chances. 2006.
- [112] Melanie Volkamer, Oliver Spycher, and Eric Dubuis. Measures to establish trust in internet voting. In *Proceedings of the 5th International Conference on Theory and Practice of Electronic Governance*, pages 1–10. ACM, 2011.
- [113] Komminist Weldemariam and Adolfo Villafiorita. Procedural security analysis: A methodological approach. *Journal of Systems and Software*, 84(7):1114–1129, 2011.