BLOCKCHAIN BALLOT

Electoral Enhancement or Danger to Democracy?

December 2017
Gijs van de Water

" Those who cast the votes decide nothing; those who count the votes decide everything"

- Joseph Stalin (alleged, Bazhanov, 1980)

Student Name	Gijs van de Water
Supervisor	dr. H. Weigand
ANR	524173
Email address	g.f.a.vdwater@tilburguniversity.edu
Company	None

Preface

It seems like a long time ago I had to start thinking about a thesis topic. I already knew I wanted to write something about blockchain, because I find new developments in IT rather interesting. Last year was also marked by controversial incidents related to vote counting during several elections across the world. Since trust is an important factor of an election and seeing that the blockchain can help to mediate trust, this seemed like an ideal combination. I therefore decided to investigate if blockchain can enhance the electoral process.

After five months of hard work, it feels satisfying I found an answer to the question whether we are better off with a blockchain based ballots or not. I do realize that I could not have done this without the people who supported me throughout the process.

I would like to thank dr. Hans Weigand who supervised my work throughout the entire thesis cycle. Both his feedback and directions were helpful.

I also want to thank Jan Willem Barnhoorn for providing me with valuable information about the blockchain technology.

I am also grateful to Guus, Joep and Roos for proofreading my thesis.

Gijs van de Water,

Hoeven, 18 december 2017

Management Summary

Several incidents related to the integrity of elections occurred globally last year. These incidents are not limited to countries with authoritarian regimes, but also took place in democratic countries such as the Netherlands. The Dutch government banned the use of IT for electoral processes due to security issues found in the software, shortly before the elections were held. The IT systems allowed malicious actors to alter the election results. Other incidents also occurred: thousands of votes were not correctly processed.

Dutch municipalities want to reintroduce IT based techniques to support the election process, which requires secure IT techniques. This thesis study determines whether blockchain is a suitable technique to enhance the election process. Blockchains are known for their security, because data is immutable once stored on the chain. This suggests that the technique might be useful for elections, as it would make it impossible for malicious actors to change the polling results in favour of a certain party.

States must comply with several criteria when organizing free and fair elections. These criteria are governed by a treaty. Examples of these criteria are ballot secrecy, accessibility, having uncertain election results and transparency. This implies that blockchain based elections should comply with these criteria. Several election scenarios created in this study are not compliant with these criteria however. Basing an election on a public blockchain allows citizens to view the blockchain contents during the election day, which could influence the decision of the electorate. Solutions to mitigate this problem exist, but it would result in two voting rounds. Requiring citizens to attend a polling station twice is not complaint with the accessibility criteria. The government could also base the election on the private blockchain. However, this is not an ideal solution since such a black box for the electorate is not compliant with criteria related to transparency. Lack of transparency gives people running for a seat the option to challenge the legitimacy of the election result, even if their arguments are not objective.

Another problem is that computers cannot guarantee a trusted election result. Malicious actors such as vendors with a political incentive to cheat or state level attackers can manipulate the voting machine to change the election result. For instance, a voter can vote for party A but the rogue computer can write party B to the blockchain in secret. There is no way to test whether a voting machine is trustworthy. Therefore, a blockchain based ballot is not suitable for organizing free and fair elections.

Table of Contents

1 Introduction

This chapter describes the problem indication of this thesis. Moreover, the problem statement and problem indication are outlined. The structure of the document is provided in the final paragraph.

1.1 Polling problems

Every four years, general elections are held in The Netherlands. People who are eligible to vote receive a polling card based on citizen registration to visit a nearby polling station and cast a vote (Jacobs and Pieters, 2009). In this station, the voter receives a ballot paper. The paper is deposited in a ballot box after marking the preferred candidate for a parliamentary seat. Checking the box for your favorite candidate is done in a voting booth to improve the ballot secrecy.

Counting the votes consists of several steps. At first, election staff opens the ballot boxes and begin to count. The results are send to the municipality. Next, the municipality adds up the votes from each polling station and delivers the result electronically to the constituency they belong to. Finally, the Electoral Council count the votes and publishes the result (Ritzen, 2016). Both the counting process and the determination process of the results are supported by the *Ondersteunende Software Verkiezingen* (OSV) software (Kiesraad.nl, 2017). However, due to vulnerabilities disclosed by *RTL Nieuws*, the government decided to ban all data carriers (e.g. USB sticks) from the election process, because the security could not be guaranteed.

The main issue with the software tool is that the software temporarily stores unsecured data on the computer, allowing third parties to alter the results. (RTL Nieuws, 2017). Moreover, inserting the vote results in the system is prone to errors as well. The Dutch Ministry of the Interior recently announced that 7.600 votes were not included in the final result, because a municipality failed to add the vote count from several parties to the system (Rijksoverheid, 2017). Although the division of parliamentary seats was not affected, it could have caused complications since the final election result is irrevocable once it is announced by the Electoral Council.

Whilst the thesis study mostly focusses on the Dutch elections, concerns about the voting process are raised in other countries as well. For instance, in the United States, a candidate must obtain at least 270 votes in the Electoral College to win the office of president. The popular vote result is not used to determine the winner of the elections (Gringer, 2008). Yet, one could argue that winning the presidency without winning the popular vote is bad for the new president's authority. Such situation occurred after the elections in 2016. In the aftermath, the then president-elect Donald Trump claimed that besides winning the Electoral College vote he would have won the popular vote if illegal votes are deducted from the result (Rampton and Volz, 2016).

The Turkish constitutional referendum in 2017 caused an outcry as well. The role of the Turkish president is mostly ceremonial. However, the leader of the ruling party desired to usurp more power. The government therefore organized a referendum in which the electorate voted in favour of more power for the president by a very small margin. The outcome of the referendum is questionable, as the Turkish Electoral Board decided to consider unstamped and unverified ballot papers as valid (Tattersall, 2017).

Finally, the party of the Venezuelan president Nicolás Maduro lost its majority in the country's National Assembly. He then organised an election for a so called 'Constituyente', a body with the power to rewrite the constitution in favour of the sitting president and to set aside the opposition (López and Brodzinsky, 2017). According to the company which worked on the voting system, the actual turnout and the turnout announced by the government differed by one million (Faulconbridge, 2017), showing the importance of a transparent election.

1.2 Problem indication

Vote counting is a vital part of free and fair elections. According to Bishop and Hoeffler (2016) the counting of votes should be trackable during the entire process. Evidence for fraud such as result tampering should not be found either. Another crucial part of elections is the voting process itself. Secrecy of the ballot and one vote per person contribute to free and fair elections. Despite the security concerns regarding the use of IT, municipalities still want to reintroduce digital voting (Van der Parre, 2017). As the method of voting should be compliant with election criteria, a secure method of voting using IT must be developed.

Blockchain is a distributed database of which the main feature is storing immutable transactions. Any fraudulent attempt to make unauthorized changes is rejected if there is no consensus by others in the network (Swan, 2015). These characteristics give blockchain technology the potential to support the elections as casting a vote is an irreversible transaction. Computers can take over the counting of votes from the election staff resulting in less counting errors and faster results. The electorate can check if their vote is processed due to transparency of the blockchain (Lee et al., 2017). Such changes will improve the trustworthiness of elections and make it easier to counter claims of election fraud.

Databases distributed throughout a computer network suggest that some sort of internet technology should be used. One may have some concerns to expose a voting system to the internet, because such systems are likely to become a valuable target for hackers. Yet, there are countries that used internet based voting despite the potential dangers of hackers. Schryen and Rich (2009) describe a case of e-voting in Estonia. After digital authentication, an Estonian voter can cast a vote. Hashing techniques are used to eliminate the link between the identity of the voter and their decision. Whereas both Estonian voters and elections officials trust the system, experts have concerns regarding transparency and security.

Electronic voting machines do not necessary improve the quality of the election process. Computer scientists question the safety and reliability of electronic voting. Traditional voting machines prevent the electorate to check whether their vote is processed. Furthermore, voters have no way of checking whether the votes are counted correctly, whereas it would be verifiable for the electorate if non-digital voting methods are used (Huijgen, 2006). The lack of a possibility to verify vote processing caused problems in the Netherlands, where the government used electronic voting machines. Even though the voting software of the vendors was certified, there was no way to check whether the same software was used in the delivered voting machines. Moreover, the source code was kept secret. Although there were all kinds of rules the voting machine had to comply with, most of these rules regulated user experience and build quality. No rules and controls were in place to prevent vote manipulation (Wij vertrouwen stemcomputers niet, 2009). The group *Wij vertrouwen stemcomputers niet* (We do not trust voting machines) successfully campaigned against the use of voting machines. They were able to convince the government that voting machines are not transparent, after which the government decided to abolish these computers (Van Heese, 2007).

This thesis study determines whether blockchain can counter these concerns and if a blockchain based ballot enhances the election process.

1.3 Problem statement & research questions

The Dutch government cannot guarantee the security of the IT used during the elections. Yet, municipalities prefer the use of IT in the election process such as voting computers. Blockchain could be a potential technology to create a secure IT backbone if it does not violate election requirements such as ballot secrecy and transparent counting of votes. Furthermore, it should not be possible to digitally interfere (e.g. Distributed Denial of Service or a hack) in the voting process to the extent that it would have an impact on the elections. Therefore, any changes made to the voting process should be secure.

The abovementioned boils down to the following problem statement:

“To what extent could blockchain improve the integrity of the election process?”

The problem statement is divided into several research questions. At first, one must know how the current voting process works in order to change it. Moreover, the new process should comply with certain criteria to ensure that the elections are free and fair. Any changes made to the way the elections work should adhere to these criteria, leading to the first research questions:

1. What are the criteria of free and fair elections?
2. How is the current voting process designed?

Next, blockchain based voting will most likely introduce some implications as well. The inner workings are discussed to identify these implications. The research questions are:

3. How does the blockchain technology work?
4. What are the implications of using blockchain technology for voting?

Finally, a new voting process is designed. One could wonder if the new process is an improvement. Hence the research question:

5. How could blockchain technology improve the election process?

The problem owner is the Dutch electorate, represented in the houses of parliament. Therefore, if the government decides to change the voting process it requires approval from the parliament. The proposed changes should be compliant with the standards of free and fair elections. This research investigates whether blockchain is a feasible solution to improve the election process. The scope of the research is limited to the actual voting process and the counting of votes. It focusses mostly on how blockchain could be used to improve the elections and whether it is an actual enhancement or a danger for democracy. Hardware needed (e.g. the most ideal voting computer) or the estimated costs to realize such a change is beyond the scope of this research.

1.4 Research design

The aim of this research is to solve an identified problem. Design science is a suitable method to conduct such a research. The research consists of two parts – a theoretical foundation and an interview to verify the results. The theoretical foundation provides with the criteria for free and fair elections. Furthermore, knowledge about blockchain is gathered. The theoretical analysis results in several scenarios for voting with blockchain, also called an artifact. (Hevner & March, 2004). These artifacts must comply with the criteria for free and fair elections. The researcher evaluates this artifact by conducting interviews with people who have knowledge about the subject to verify whether the solution is indeed safe and secure. Verification is important as the potential solution relies on the safety and security of blockchain.

1.5 Structure

This document is divided into several chapters. The literature review focusses on the criteria of free and fair elections. The electoral procedure in the Netherlands is also documented, since any replacement is based on the current approach. Moreover, information about blockchain is gathered. All parts of the literature review are used as building blocks for new election scenarios. The chapter hereafter describes how the election scenarios are created and how the verifications are conducted. Next, three different election scenarios are described and evaluated. The final chapter summarizes the results and presents recommendations.

2 Literature Review

This chapter focusses on the relevant two variables: elections and blockchain. Reviewed literature is used as a building block for new election models based on the blockchain technology.

2.1 Free and Fair elections

Not all countries in the world are democratic. Whereas 79 countries are considered a democracy, 51 other countries have been marked as having an authoritarian regime. Countries are categorized based on several indicators of which electoral process is one of them (The Economist, 2015). Several treaties and international law obligate states to organize legitimate elections. This is mainly enforced by the International Covenant on Civil and Political Rights treaty, ratified by the U.N. General Assembly in December 1966. Article 25 provides the right for citizens to take part in the affairs of their state, requiring democratic legitimacy (Tomuscat, 2009).

2.1.1 Criteria for free elections

Calling elections does not necessary make a state democratic, as an election should also be free and fair. Freedom means that one should be able to choose something such as a candidate or party over another. Fairness of elections implies that some citizens are not given advantages over some others (Elklit and Svensson, 1997). These general terms are not sufficient for clear and consistent election assessment by observer groups (Davis-Roberts and Carroll, 2010).

For instance, ballot secrecy is an important characteristic of free and fair elections. It helps people to vote without having to worry about intimidation, threats or retaliation (Geber, Huber, Doherty, Dowling and Hill, 2013). Criminals could force citizens to vote in favor of their preferred party. It would be rather easy to check whether they have indeed casted their promised vote if ballot secrecy would not exist. Furthermore, not being able to cast a vote in private entails corruption as well (Heckelman, 1995). Political parties could pay or reward the electorate voting for them. Therefore, not having systems in place to guarantee the privacy of the voters’ choice could influence the outcome of the elections.

This thesis study requires a more coherent definition of free and fair as this is usable to validate the feasibility of blockchain based elections. Davis-Roberts et al. (2010) compiled a table based on treaties, customary international law and declarations from international forums with a large amount of participating states providing criteria and requirements for democratic elections. The table (table 1) is appended by findings from other papers.

Requirement	**Source**
Genuine and periodic elections	ICCPR, Article 25; Cheibub, Gandhi and Vreeland (2009)

Equal suffrage
Each vote has the same weight, e.g. a vote is not more important if cast by a man instead of a woman. This also means that people can only vote once. Fraud and casting multiple votes should be prevented.
	ICCPR, Article 25;
	Bishop and Hoeffler (2016);
	Goodwin-Gill (2006);
	Jacobs and Pieters (2009)

Secret ballot
It should not be possible to link a vote to a person.
	ICCPR, Article 25;
	Bishop and Hoeffler (2016);
	Goodwin-Gill (2006);
	Jacobs and Pieters (2009)

Universal suffrage The highest number of citizens as possible should be able to cast a vote.	ICCPR, Article 25; Jacobs and Pieters (2009)
Prevention of corruption	UNCAC
Citizen right to vote Each individual citizen has the right to vote. This may be restricted by certain criteria. (e.g. age)	ICCPR, Article 25; Bishop and Hoeffler (2016)
Citizen right to be elected	ICCPR, Article 25; Bishop and Hoeffler (2016)
Participate in public affairs	ICCPR, Article 25
Freedom of assembly Candidates may compete and hold campaign rallies.	ICCPR, Article 21
Freeform of association One may establish a political party.	ICCPR, Article 22

Requirement	Source
Freedom of movement Election officials, observers, citizens and others should have freedom of movement. Abroad citizens should have to possibility to return to participate.	ICCPR, Article 12
Equality under law & absence of discrimination	ICCPR, Article 2, Article 26
Equal access, public service & property	ICCPR, Article 25
Freedom of expression & opinion	ICCPR, Article 19

Access to information / Transparency
One has the right to seek and receive public information. This is considered critical for transparency and accountability throughout the entire election process. Everyone can observe the tallying process during election day.
	ICCPR, Article 19;
	Bishop and Hoeffler (2016);
	Goodwin-Gill (2006)

Right to security of person One should not be interfered or be intimidated during the elections (e.g. voters, observers)	ICCPR, Article 9
Right to a fair and public hearing	ICCPR, Article 14
Right to remedy	ICCPR, Article 2
Uncertain election result The result of the election should not be known before the elections have taken place.	Cheibub, Gandhi and Vreeland (2009)
Access to polling station & ballot box Voting booth access should not be obstructed (e.g. by police). This requirement is not restricted to physical obstruction, but also that access should be as easy as possible. (e.g. make sure the electorate does not have to travel a long way to access the ballot)	Bishop and Hoeffler (2016); Goodwin-Gill (2006) Jacobs and Pieters (2009)
No tampering, manipulation or destruction of ballots	Bishop and Hoeffler (2016)

Table 1: Requirements for free and fair elections

2.1.2 Voting in the Netherlands

In the Netherlands, the executive body of a municipality is responsible for organizing the elections for the lower house of parliament. Such responsibilities involve sending polling cards to citizens eligible to cast a vote, creating a polling station and assigning election officials to the stations. These polling stations are part of a constituency, headed by a main polling station. The final election results are announced by the Dutch Electoral Council ("Regeling - Kieswet", 2017).

The Dutch government published a document containing the rules and procedures in the polling station (Stembureau-instructie voor de dag van de stemming, 2017). According to this document, voters have a restricted timeslot to cast a vote on polling day. Voting is only possible between 07.30 and 21.00. Once a voter enters the polling station, several procedures apply. These procedures are verified by an election official.

(1) The voter should hand over a document verifying his identity such as passport or an ID cart. A polling card to prove that the citizen is a registered voter in the municipality is provided as well. Casting a vote is not allowed if the voter fails to provide either of these documents.

(2) The chair of the polling station checks whether the ID card is valid. Furthermore, the person who wants to cast his or her vote should be the legitimate owner of the polling card. The names and the birthdate on the identity document should match with the data on the polling card. Moreover, the chair checks whether the polling card is a legit document. If there is reason to believe the polling card is forged, the card is seized and the voter is reported to the municipality. The voter is prevented from voting if there is no match or if the polling card is forged.

(3) The chair reads out loud the number on the polling card. Another election official possesses a list of invalidated numbers. If this list contains the number of the polling card, the card is seized and the voter is not allowed to cast a vote.

(4) If the voter passed all checks and everything is fine, the voter receives a ballot paper. This ballot paper contains all candidates running for a seat in parliament. The voter may cast a vote in a polling booth. The voter is supposed to be alone in the booth when casting a vote. Persons with physical disabilities are exempt from this rule. All polling cards are stored by an election official for later use.

The ballot box closes at 21.00. The polling station remains open for the electorate to keep an eye on the vote counting process to improve the transparency. Observers are allowed to file a complaint if they think something is wrong. These complaints are documented by election officials. The polling cards are counted before the ballot boxes are opened as this is usable as an additional check.

Once the counting is finished, the final vote count per candidate and party are transferred to the main polling station of the constituent. Before elections officials were banned from using the OSV-software by the government, the following method of transferring results was used: Members of the polling station enter the result in the software. Next, the software generates a XML file containing the results, which is transferred to the main polling stations. The result for the entire constituency is then calculated by the OSV tool and a new XML file is delivered. Finally, all the XML files of the constituencies are provided to the Electoral Council. The latter is responsible for announcing the official election results (Engberts, 2011).

2.1.3 Security precautions

According to the Council, a hash code is added to the election report to prevent people from tampering with the result. An election official must compare the hash when loading the XML file from the USB stick (Kiesraad, n.d.). Engberts (2011) questioned the use of hashing to secure the result. A hashing function is a function for which the file contents result in the fingerprint of the file. This is a one-way function. It is not possible to extract the file contents from a fingerprint (Naor and Yung, 1989). For instance, hashing the sentence *“Bob has been elected to represent this constituency”* results in the fingerprint *033cd476cb548cf5a9ffea2cbb76582a*. If one would change the result into “*Mallory has been elected to represent this constituency*” the fingerprint changes to *27c769cf6bdd253bba564d89b49d103e*. In other words, calculating a hash of a digital file or string reminds of calculating the sum of a set of numbers and it is not possible to tell from the calculated sum which numbers are in the provided set. Yet, a validator can tell if a number in the original set is changed, because such a change results in a different fingerprint.

Hashing functions are therefore indeed usable to check whether a file is compromised or tampered with (Wang and Yu, 2005). However, hashing algorithms are publicly available on the internet. Anyone could calculate the fingerprint for a certain file by simply running a hash calculator over the file. The Electoral Council wrongly assumed that they could prevent malicious actions by adding the fingerprint from the XML to the result file. If the computer that is running the OSV software is compromised, a new fingerprint matching a tampered result is easily calculated. Moreover, election officials with bad intents can manually edit the XML file placed on the USB stick before handing it over. If they calculate a new fingerprint, their actions are not noticed by the OSV software, because the fingerprint still matches.

2.1.4 Voting process requirements

After examining the voting process, one can conclude that not all requirements described in table 1 are applicable on the voting process itself. As an example, the right for people to be elected or the right to establish a political party are obvious requirements for a functioning democracy. Yet, these rights do not contribute to the technical aspects of the voting process. It is important to distinguish the process related requirements from the non-process related requirements to verify whether blockchain supported elections are free and fair. For instance, elections require identification of the voter. This is used to record that a vote has been cast to guarantee equal suffrage. On the contrary, one should not be able to back trace a cast vote to a citizen as they have the right for a secret ballot (Rössler, 2009).

Another process requirement is the need for access to information for the sake of transparency. Even if an (electronic) voting system is very secure, it is only usable if the public believes it is secure. A voter-verified audit trail is required to gain the trust of the electorate (Oostveen and Van den Besselaar, 2004). Without such trails, vicious actors could manipulate the vote count. In the U.S.A., electronic voting machines print the voting results besides storing it digitally. The redesigned voting process should consider what should happen if the electronic result differs from the paper trail (Jacobs and Pieters, 2009). One must be aware that reading the blockchain contents should not be possible during the elections, because uncertain election results are required. A voting system is not supposed to 'leak' an early estimation. These early estimated results could influence the vote of citizens. Finally, the election process should contain checks and constraints to make sure only eligible people can vote.

2.2 Blockchain

Imagine a village which is about to be sacked by two armies. One army resides in the east and one in the west. To make the attack successful, the armies must attack at the same time. Therefore, the general of the army in the west sends a messenger to the general in the east to let them know they are going to strike next Wednesday. A few hours later, the messenger returns and acknowledges that there is an agreement. When the western army marches to the village on Wednesday, they suffer a terrific defeat. The eastern army did not show up. While the eastern general was in good faith, the messenger was not. The villagers paid him to alter the generals message. As far as the eastern general knew, the sack was planned for next Thursday. This abstract problem is described as first by Lamport, Shostak and Pease (1982) as the Byzantine Generals problem. It can be related to the election process in the Netherlands. Malicious people or software that transfer the election result of a constituency could modify the result during transport.

The election process is more reliable if those who receive the vote count can make sure it has not been tampered with, even if transporters had opportunities to do so. Blockchain is such a technique which has measures against tampering with the results (Cachin, Schubert and Vukolić, 2016). The government could mitigate the security issues in the OSV tool and the vote counting errors by using this blockchain technology. Moreover, blockchain could safeguard the principles for free and fair elections described in table 1. A blockchain is a ledger distributed among peers in a computer network. The blockchain ledger consists of blocks. These blocks are comparable with pages in a physical ledger. A block contains transactions which are not stored yet in a previous block. Blocks in the ledger are ordered as new blocks can only be added at the end of the chain. Furthermore, it is not possible to make changes to blocks which are already stored in the blockchain, meaning that blockchain ledger is immutable. This is enforced as long the majority of computer power in the network is trustworthy (Pass, Seeman and Shelat, 2017).

2.2.1 Trusting the chain

Cryptographical functions are used to guarantee the trustworthiness of the blockchain. For instance, the hash result (digest) of a block in the chain and a string (nonce) are used as input of another hash function. The result is used to link the next block in the blockchain (figure 1).

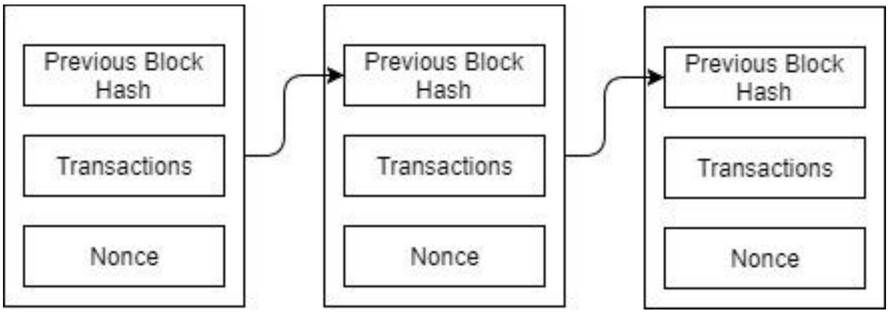

Figure 1 A graphical simplified representation of a blockchain

The challenge computers face in the network is to find the correct nonce. To satisfy the blockchain, hash results to link the blockchain must meet certain requirements. Therefore, a computer must try several inputs until it finds a correct hash. This consensus method is called a Proof of Work (Badev and Chen, 2014).

The result is broadcasted to the network and added to the blockchain if considered valid (Badev and Chen, 2014). Because the blocks are linked using hashes, it is very hard to alter the chain as this would invalidate the consecutive blocks. This also implies that blocks closer to the root block (the genesis) are even more trustworthy, because a new Proof of Work would have to be calculated for all the consecutive blocks. It is nearly impossible to recalculate these blocks while competing with the other network participants as they keep adding new blocks, making the method quite secure.

As multiple computers work on adding new blocks to the chain, it is possible that two valid blocks are produced at the same time, resulting in a new branch on the chain (Decker and Wattenhofer, 2013). A new branch is called a fork. Forks are not desirable, because the network should agree on one globally accepted ledger. Forks could represent multiple versions of the truth. The longest chain is therefore considered valid to prevent different versions of the same ledger (figure 2).

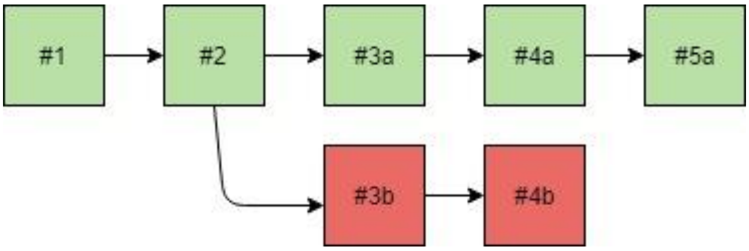

Figure 2 Forks

The forked chains eventually cease to exist and clients can resubmit lost transactions stored on the removed fork, contributing to the theory that blocks closer to the genesis block are more reliable and trustworthy (Eyal and Sirer, 2014). Fresh transaction might end up on a forked chain, which is risky since there is a chance the fork eventually disappears. Transactions residing on the fork are therefore unverified and awaiting confirmation again.

A downside of Proof of Work is that it consumes a lot of energy (O’Dwyer and Malone, 2014). This seems rather logic, given that computers need to carry on calculating a nonce repeatedly. Luckily, other consensus methods exist besides the Proof of Work. The alternatives are much more energy efficient. An alternative is Proof of Stake, where stakeholders preserve and protect the blockchain (Kiayias, Russel, David and Oliynykov. 2017). The philosophy behind Proof of

Stake is that stakeholders are trustworthy, because a breach of security reduces the value of the stakes (Bentov, Gabizon and Mizrahi, 2016). When using Proof of Stake, participants of the blockchain have a better chance of generating a block if they own a bigger part of the network's stake. The chance of creating a new block is determined by the amount of the coin age a node has. The coin age is the amount of coins the user owns multiplied by the time those coins were in possession of the user (Vasin, 2014). Network peers can verify the coin age, because information is publicly available in the chain (Wenting, Andreina, Bohli and Karame, 2016). A user must commit a transaction to itself to become eligible to generate the next block. The coin age is reset for the user once their block is successfully added to the chain (Vranken, H). Some Proof of Stake implementation punish users if a block is not successfully added, because it contains invalid or fraudulent transactions (Tikhomirov, 2017). King and Nadal (2012) rightfully argue that the Proof of Stake system is basically a 'Proof of Work in the past' system, because money is a form of work in the past.

2.2.2 Public and private keys

Blockchain uses encryption methods to sign off transactions. There are different methods to perform encryption. One of these methods is symmetric-key encryption. As implied by the name, the same key is used to encrypt and decrypt the data (Delfs and Knebl, 2007). An example of symmetric-key encryption is using the next letter in the alphabet instead of the actual letter when writing a secret message. Those who know this secret can decrypt your message by simply applying this key in reverse. It is obvious that such an encryption suffers from a flaw. Both parties already need to have knowledge about the secret key. The parties must somehow establish a secure channel to transfer the key, before they are in possession of a key used to create a secure channel.

Asymmetric encryption consisting of two keys is used to solve this problem: a private key and a public key. The public key is used to encrypt data and the private key is used to decrypt data. The keys performing these operations are paired. Therefore, it is only possible to perform decryption operation if the related private key of the public key is used and vice versa (Sako, 2005). Asymmetric encryption technique can also be used to verify the senders' identity. The sender must own the private key if decrypting a message using a public key succeeds. If the private key used to encrypt a message is either invalid or missing, the decryption would fail.

An address is needed to send a transaction to an actor on the blockchain. Just like an email, the transaction needs to know about his destination. Additionally, a private key is needed to sign off this transaction. In Bitcoin and other cryptocurrencies, it is only possible to transfer money from your wallet to another address if a private key is possessed. The address is derived from the public key (Swan, 2015). It has been shown that private- and public keys are paired. This explains why actors can commit a transaction to an account, but not commit a transaction from an account. For the latter, access to the private key is needed. If such a private key is forged, the public key does not match, making the signature invalid.

2.2.3 Transparency and privacy

A tension exists between the transparency of the blockchain and privacy. Transparency does require public verifiability (Wüst and Gervais, 2017). Transparency may not cause privacy problems with monetary blockchain systems such as Bitcoin, because transactions are linked to a public key and not a name (Reid and Harrigan, 2013). Therefore, one can figure out how much money a wallet contains by calculating the sum of the transaction contents, but not to whom the wallet belongs. On the contrary, transparency will cause problems if it occurs in blockchain based election systems. Being able to count the number of votes cast in favor of a specific candidate during the elections violates one of the criteria of free and fair elections described in table 1. The election result must remain uncertain until election day is over. Wüst and Gervais (2017) suggest the use of permissioned blockchain or computationally expensive cryptography to counter these issues.

Otte, de Vos and Pouwelse (2017) describe several types of applying permissions to blockchains. One of these types is permission-less. Therefore, no single authority is needed to approve participation in the blockchain. A permission-less application of the blockchain is transparent as the contents of the chain are publicly available. Another type is a permissioned or private blockchain, hiding transactions by default. Users need approval before sensitive data is accessed, suggesting that permissioned blockchains are suitable to guarantee the uncertainty of the election result.

2.2.4 Smart contracts

Certain conditions apply when casting a vote. For instance, equal suffrage implies that a citizen may only vote once. Blockchain has mechanisms to enforce such rules by using so called smart contracts. These contracts can make transactions conditional based on environmental variables (Peters and Panayi, 2015). Conditional transactions suggest that the election authority can write a smart contract specifying the rule that each citizen can only vote once during election time. Transactions breaching these rules are rejected by the network. Once a contract is created, it can no longer be modified. The contract can only be triggered by a transaction send to the address of the contract (Zang et al., 2016). Furthermore, smart contracts are consistent. The network would fail to reach consensus if a smart contract returns random results whereas the conditions are the same (Christidis and Devetsikiotis, 2016). Smart contracts therefore increase transparency. Anyone could check the rules and conditions used for an election, because malicious actors cannot change the rules afterwards. This implies that developers of contracts should thoroughly test them, as applying patches is not possible either.

2.2.5 Transaction processing capacity

The maximum size of a block and the time required to create new blocks does affect the speed of processing transactions. For instance, the popular blockchain-based currency Bitcoin is only able to process 7 transactions per second, translating to roughly 600.000 transactions per day. (Croman et al., 2016). Although such a transaction speed is enough to support elections for a municipality, supporting national elections requires more capacity. One could argue that increasing the size of the block is a solution. However, according to Decker and Wattenhofer (2013) an increased block size leads to slower block broadcasting throughout the network, evidentially leading to network nodes which are not fully updated. More forks of the main chain are more likely to occur. Sompolinsky and Zohar (2015) state that more forks harm the security of the chain. This makes sense, because it is easier for malevolent actors to draft a long chain which is eventually chosen as the main chain due to the longest-rule. Yet, Gervais et al. (2016) argue that it is possible to significantly improve the amount of transactions per second. They found that increasing the block size up to 8 MB increases the distribution time of the block linearly. The time increases exponentially after the 8 MB. According to their research, an amount as high as 66,7 transactions per second is reachable if the correct block size and interval are used. Even though this remains a limitation, it brings blockchain based elections on national scale much closer. An expert on this subject stated that blockchain developers are likely to solve the scalability problem.

2.2.6 Attacking the chain

There are several methods to attack a blockchain. Given that there are reasons to undermine an election it is useful to charter these attacks. Even though blockchain has only few vulnerabilities, mischievous actors might try to exploit these, especially if there are elections at stake.

The first type of attack is the majority attack, also called the 51% attack. In Proof of Work based blockchains, computers must run calculations continuously to add a new block to the chain. The philosophy behind this method is that evil-minded actors must compete against the entire network when trying to make fraudulent changes to the chain. Because new blocks are still added to the chain when such an attack is performed, the bad-natured actor must compete with these as well. However, they can combine forces with others to gain the majority (51%) of the computer power in the network. Once such power is obtained, the attacker can outperform the nodes in good faith and modify the chain as pleased (Lin and Liao, 2017). It is likely that Proof of Stake based blockchains are also exposed to such attacks. The attacker needs to obtain a majority of the stake instead of gaining the majority of the computer power.

Another way of cheating the chain is to collude with block miners to keep mined blocks private from other nodes in the network. The fraudulent clique continues to mine on their privately kept blocks. The other nodes keep mining on older blocks of the chain, because they are unaware of the unreleased blocks. This is called ‘selfish mining’ (Kaushal, 2016). It makes sense that this type of attack is effective, as the longest chain is considered as the true version if a fork is created (Eyal and Sirer, 2014). A fork is created when the colluding miners release their blocks to other nodes. The part of the chain where honest chains are working on is deemed invalid if the created fork is longer. Colluding miners can succeed, because honest miners are wasting their time as they are not aware of the latest blocks when running calculations.

Additionally, attackers can perform rogue actions by abusing bugs which are found in the blockchain software. It is not uncommon for hackers to exploit mistakes the developers of the software made. For instance, the software OpenSSL used to guarantee a secure tunnel over the internet to transmit confidential data such as passwords or bank details contained an infamous bug disclosed in 2014. This bug, dubbed Heartbleed, allowed hackers to gain access to the confidential data. Furthermore, since hackers could access the private keys of the server it was possible to impersonate the server as well (Gujrathi, 2014). Even blockchain can suffer from such bugs. For instance, the cryptocurrency bitcoin suffered from a bug allowing adversaries to disguise a 0.5 bitcoin transaction as a transaction containing 184 trillion bitcoins (Park and Park, 2017).

Distributed networks relying on identities can face security issues from faulty or malicious participants. A treat arises from the fact that without additional security measures, single nodes can pretend to have multiple identities (Douceur, 2002). For instance, imagine a forum administrator starting a poll to decide on a date for a barbeque. Only forum members can vote. Forum member Mallory can only attend this barbeque on a specific date. Therefore, she decides to register several new accounts on the forum. She uses these accounts to vote on her preferred date. Because of her unfair advantage, Mallory succeeded and the barbeque is organised on the day she prefered. Such an attack is called a *Sybil*-attack. Public blockchains suffer from the same issue as anyone can create as much accounts as pleased, because there is no central authority acting as gatekeeper. Therefore, the blockchain needs additional protection in the consensus phase, where nodes vote on the valid block. The Proof of Work or Proof of Stake are such measures to prevent Sybil attacks. Attackers can create an infinite number of identities, but obtaining a large amount of computer power or money is much harder.

To conclude, blockchain is not as secure and invulnerable as some people believe. Blockchain does suffer from weaknesses allowing attackers to alter the blockchain. Even though it looks unlikely that hackers exploit some of the vulnerabilities that are described here, it remains a possibility. For instance, if state-level actors find an undisclosed bug and if something valuable such as elections are at stake, the attackers can at least try to do so. It would not be the first time that states launch digital attacks. Governments already tried to sabotage nuclear power plants in Iran by crafting the malware Stuxnet. This mischievous piece of software was specially designed to target bugs in the controlling software for these power plant (Chen, 2010).

2.3 Electronic elections

This section describes findings from literature regarding the risks of electronic elections.

2.3.1 Estonian elections

Introducing internet based elections requires digital abilities. A country known for its digital abilities is Estonia. The Baltic country was the first nation which opened their digital borders for the e-residency program. Anyone on earth can apply for e-residency. The purpose of this digital identity is to access and interact with both public and private services. One can open a bank account or register a new company without psychically attending Estonia (Sullivan and Burger, 2017). Besides e-residency, the Estonian government offers a lot of services through the internet, such as vehicle registration, a population register and medical records (Anthes, 2015). Estonia also allows internet based voting for parliamentary elections. According to Alvarez et al. (2009), there are four reasons why voting using the internet in the Baltic country is successful. At first, internet access is widespread. Second, a legal structure to deal with issues regarding internet voting is present. Third, an authentication system which can identify the voter exist. Finally, political support for e-voting is present as well.

Even though 30% of the Estonian electorate used the internet to cast their vote, the system remains controversial. Researchers who investigated the system found serious security flaws (Springal et al., 2014). Although the voting system does not use the underlying technology, their findings (table 2) remain usable to verify if blockchain would counter these problems.

Type	**Description**	**Target**
Ghost click	Malicious software is installed on the client computer to change the cast vote to a vote preferred by the attacker. The person who votes does not immediately notice this, because the computer displays a notification that the vote has been successfully registered.	Client
Bad Verify	Estonia allows users to verify their vote by scanning a QR code on their smartphone up to 30 minutes after a vote is cast. The assumption is made that the chance that both the client machine and smartphone are not compromised at the same time. This is not necessarily true, because smartphones synchronize with PC's quite often. Android allows installation of applications over the air. The Bad Verify attack replaces the verification application with a fraud.	Client
Inject Malware	The Vote Counting server is installed from a DVD disk by election officials. A rogue official could inject malware during the installation. This malware could force the server to make mistakes when counting the votes, in favour of a specific party or person.	Server
Man-in-the-Middle	A hash of the server software installation file is checked against a hash file on the vendor website. However, because a secure connection was not used, malicious actors could provide a fake hash. When a man-in-the-middle attack is performed, a computer makes the communicating computers think they are communicating with each other. In fact, they are communicating with the attacker who can change the contents of the message before sending it to the intended receiver of the message. The use of private and public key techniques could have prevented this.	Server

Type	Description	Target
Vote-Stealing Payload	Malware containing a vote-stealing payload wrapped around the counting process. This wrapper manipulates the vote count before the result is displayed on the computer screen. It basically reminds of the "Ghost click" attack described earlier in this table.	Server

Table 2: Overview of the Estonian election system vulnerabilities.

2.3.2 Weakness on the surface

Blockchain based elections are not likely to suffer from the server based attacks. An attacker would need to compromise at least 51% of the network participants to make an attack successful. However, the client based attacks are a genuine threat. Applications residing at the surface of the blockchain are still vulnerable to these attacks. To elaborate, a malicious actor can compromise the client software which sends a transaction to the blockchain. The client software could either alter the transaction before storing it on the blockchain or voiding the transaction, whilst displaying a "Transaction Successful" message to the user. To illustrate such problem with voting machines in general, Gonggrijp and Hengeveld (2007) successfully installed a chess game on a voting machine, arguing that a voting machine is just a computer and is usable to play chess, or alter the election results. The same authors mention that the best countermeasure used to prevent such threats is to isolate voting computers on election day and test them. They openly question whether this is effective enough.

The Volkswagen emission scandal in 2015 proves them right. In the United Stated and other countries, the maximum amount of emission is constrained. The Volkswagen cars satisfied these constraints. What the authorities didn't know was that the cars were equipped with software to determine whether the car is in a testing environment or not. Therefore, cars behaved differently in the testing room compared to the behaviour on the road. One of the Volkswagen models exceeded the emission limit up to 35 times (Blackwelder et al., 2016). This suggests that malicious actors can also add such software to voting computers, allowing the computers to behave correctly in the tests but cheat during the real election.

Several other authors also challenge the safety and legitimacy of electronic voting. According to Lauer (2004), malware such as trojan horses could compromise the election result. Detection of this malware is rather hard if it is baked into the election software. It is also possible to add this malware without knowledge of the vendor. For instance, the compiler used to translate the computer source code to machine code could have malicious intents. Bratus, Lembree and

Shubina (2010) describe that malevolent compilers can add rogue instructions to the software without the knowledge of the programmer who compiles the program. It is likely that several other factors could influence the software on electronic voting machines as well. For example, the operating system on which the voting machine is developed could have malicious intents and modify the software in a malicious way. Another possibility is that the computers of the voting machine manufacturers are already compromised with malware specifically targeting the development of the voting software. Therefore, Mercuri (2002) hits the nail on the head when arguing that a troubling "trust us" mentality arises when relying on manufacturers to build a digital ballot box.

This evidence suggests that the surface of the blockchain is indeed a weakness. Even if the blockchain is considered 100% secure, the client software writing to the blockchain can have malicious intentions. This claim is confirmed by a blockchain expert.

3 Methodology

To verify whether blockchain is usable for national elections, certain steps are conducted. The first step is to model a few scenarios of how blockchain is applicable. These scenarios are based on the current election process in the Netherlands. Furthermore, the scenarios should not violate the basic electoral rights related to the election process. The main elections process rights mentioned in table 1 which the blockchain application should not violate are:

(1) Equal suffrage
(2) Secret ballot
(3) Citizens right to vote; restricted by certain criteria (age)
(4) Access to information. Both the result and the process should be transparent
(5) Uncertain election results
(6) No tampering, manipulation or destructions of ballots
(7) Access to polling station & ballot box

Scenarios are modelled and validated against these rules once they have been finished. To improve the quality and reliability of the results, the models are validated with experts with knowledge of blockchain. Moreover, information is obtained to pick a preferred election scenario if applicable. Structured interviews are useful if it is known what information is needed (Sekaran and Bougie, 2016). Table 3 illustrates the questions asked during the interview.

Question	**Justification**
What you think of blockchain-powered elections?	Obtain general information about blockchain used for elections.
Would you prefer public or permissioned blockchains if used for electoral purposes?	Confirmation of advantages and disadvantages of both types of blockchain if used for elections.
Blockchain is known for its hardened security. Are the blockchain clients as equally secure?	According to the literature, computers can store different data than the data shown on the computer screen. These systems are not based on blockchain though but it seems like a problem of client software in general

Question	**Justification**
How you think that the scalability constraints of blockchain can be resolved to make the technology applicable for national scale elections.	Currently, organising national scale elections are not possible due to limited transactions per second.
Most public blockchain clients have a limited transaction per second throughput. Are there any other limitations which may hinder national scale elections?	Both experts and literature claim that the performance issue regarding transactions per second is going to be solved in the foreseeable future. It is important to know if the performance is the only constraint for national scale elections.
The literature describes different requirements elections must meet. Accessibility, transparency and an uncertain result of the elections seems to conflict if blockchain is applied to the election process. What do you think about this trade-off?	Confirmation of the trade-off deducted from the literature.

Table 3: questions asked during the interview

The modelled scenarios are used to give the interviewed expert insight in the problem. If any of the scenarios remain valid even after the analysis of experts, one could argue that blockchain-powered elections are a possibility indeed. However, blockchain based elections are not the right solution if they suffer indeed from the trade-off between accessibility, transparency and uncertainty of the election result.

The expert Jan Willem Barnhoorn (IT Integrator at ABN AMRO Bank, currently investigating several use cases based on blockchain technology) is interviewed to answer these questions.

4 Election scenarios

Several methods exist to create a blockchain based election system. For instance, the Electoral Council could adopt a new voting system based on either a public- or permissioned blockchain. Furthermore, the scope of the digitalization is adaptable. The government could opt for a more conservative overhaul, which only digitalizes the vote counting. This implies that the electorate should still visit a polling station to cast a vote. The election officials take care of the authentication process. A voting machine is used to register the vote on the blockchain. The more progressive approach also allows the electorate to vote without visiting a polling station, because a vote can be cast over the internet. This chapter focusses on both type of overhauls.

4.1 Public blockchain

Ethereum is a cryptocurrency based on a public blockchain. It also supports smart contracts, which makes it possible to run a program on the chain. As described before, smart contracts are immutable once they are deployed to the blockchain. In Ethereum, these contracts consist of three parts. The executable component which is executed by the network, a storage file and an account balance. The contract is executed when it receives a notification. Both users and other smart contracts can send such messages. To prevent abuse of the network, programs on Ethereum run on digital gas, which users can buy using the digital currency ether (Delmolino, Arnett, Kosba, Miller and Shi, 2015). This is just like a car. It does not use fuel as long it is not used. The car stops driving when it is out of fuel. The same applies to smart contracts. Running out of gas result in termination of the contract until the gas is replenished.

Whereas Bitcoin uses transactions, Ethereum uses messages. Messages in Ethereum have three characteristics which differ from the Bitcoin system (Buterin, 2014). First, both a contract and an account can generate a message. Allowing contracts to generate messages is a precondition for automated systems. Second, a message can contain actual data. This implies that votes can cast a vote containing a name of someone running for a seat. Finally, contracts support return statements. These statements are basic building blocks for programming languages and allow the use of functions. Therefore, the Ethereum blockchain does have the basic toolset for building an election application. The scalability which has been described before, remains a limitation.

The least obtrusive method to introduce blockchain is to replace only parts of the current election process. Therefore, the first model will only affect casting a vote and tallying the votes.

Election officials remain in charge of the authentication activity and polling station play a key role during the election process. This implies that:

(1) Electorate eligible to cast a vote receives a polling card calling upon them to visit a nearby polling station.
(2) The voter is authenticated in the polling station with both an identification card and the polling card. Permission to vote is granted if both documents are genuine.

In this model, changes are made in the few steps that follow next. Casting a vote and the counting process is replaced with blockchain technology. This is illustrated in figure 3 which represents a part of the election process. Activities which are changed using this model are marked.

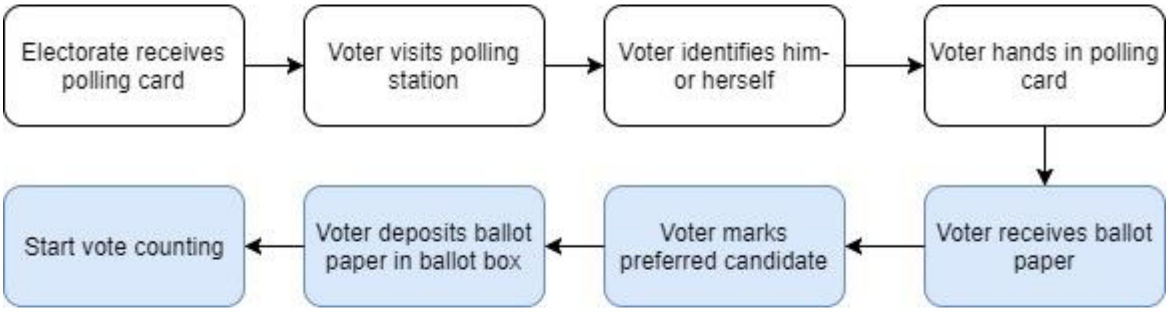
Electorate receives polling card
Voter visits polling station
Voter identifies him- or herself
Voter hands in polling card
Voter receives ballot paper
Voter marks preferred candidate
Voter deposits ballot paper in ballot box
Start vote counting

Figure 3 Election process activities

4.1.1 Equal suffrage and ballot secrecy

Rössler (2009) describes a tradeoff between the constraint that a voter may only vote once and guaranteeing the right of a secret ballot. This is enforced in the current electoral process by handing in the polling card. Nevertheless, the blockchain still needs proof that the citizen committing a transaction has the right to vote. Registering the unique ids of the polling cards on the blockchain and only accept transactions containing a valid ID seems a viable solution at first, but this still violates the secret ballot as a polling card is linked to a citizen. This means that anyone obtaining a polling card can figure out what the person voted. Ballot secrecy is still achievable by separating these domains. The blockchain still needs a list of codes which can are usable to cast a vote. According to the Ethereum smart contract documentation (“Introduction to Smart Contracts”, n.d.) it is possible to compare the address of the account casting the vote with a list of approved addresses. A vote is nulled if the caster is not on the list. The smart contract can use the same technique to make sure a vote is only cast once.

Using the blockchain technique this way suggests that the electorate need their own Ethereum accounts. Considering it is not likely that everyone has the knowledge to create such an account, it would violate the accessibility clause in the Free and Fair election table. For that reason, the government itself should create the accounts for the electorate, merely for voting purposes. This gives the government a chance to whitelist these accounts for voting as well. Next, they can securely seal and distribute the private keys used for accessing these accounts to the polling stations. Election officials can hand over this key to the voter once the voter is authenticated. The Electoral Council should instate controls to prevent fraud with the private keys. For example, polling stations should return both the leftover private keys and the polling cards which were handed in to the municipality. The municipality can detect missing keys if the numbers do not add up.

4.1.2 Uncertain election results

The tensions between transparency and privacy as described by Wüst and Gervais (2017) do apply here. Because Ethereum is a public blockchain, anyone can view the contents of the transactions made. This needs attention as this violates the rule dictating uncertain election results. A possible solution is to encrypt the contents on the blockchain, allowing the election authorities to decrypt the results once election day is over. However, few problems arise using this approach:

(1) Election officials cannot tally the votes if the keys of the encrypted data are lost. There is no viable way to recover the results if something like this occurs. According to Al Hasib and Haque (2008) it is impossible to brute force AES encrypted data, because of the high number of possible keys. Even though their statement is from 2008, it likely applies to more recent encryption techniques as well.

Rogue election officials can also get rid of the keys on purpose if they are not happy with the election results. Moreover, keys could leak or be obtained by malicious actors. If such keys are leaked to the public, the election principle of unsure election results no longer applies.

(2) The nature of public blockchain is that transactions are public. This is an issue if the blockchain should take care of the encryption. To illustrate, if a smart contract contained the code for encrypting data, the contract needs to know the contents of the data in need of encryption. A way to provide this data is via transactions. Encrypting on the

blockchain therefore does not make sense, because the original data is retrievable by looking for the transactions committed to the smart contract.

For these reasons, the model needs another viable method to ensure the uncertainty of the election result. It is possible to store data in a transaction (Wood, 2014). Voting computers can store the encrypted vote here, instead of sending a vote in plain-text to the blockchain. Whereas this solves the problem of public transaction, this method still relies on a central authority which has to decrypt the data.

Another solution is to send a hash of the preferred vote to the blockchain. The main advantage of this is that using this technique eliminates the need of a centralized actor who decrypts the data. Naor and Yung (1989) also state that reversing a hash to its original contents is not even possible as it is merely a fingerprint of the data it represents. The downside of this solution is that it requires two voting rounds. The first voting round is for the electorate to commit a hash of their vote to the blockchain (figure 4).

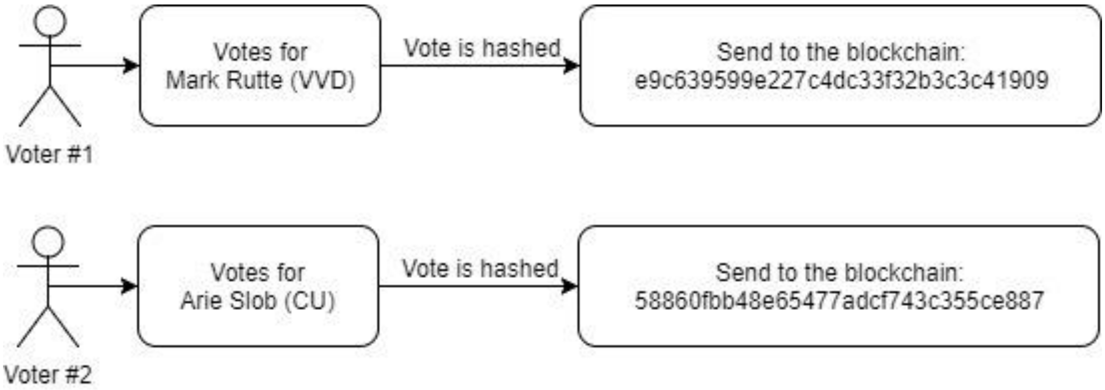

Figure 4 Voting in the first round

The second voting round starts when the first round is closed. The purpose of the second round is to reveal the vote. The electorate casts their vote again in plain text. This implies that the contents of the votes cast are visible during this round. However, the computer can compare the fingerprint stored in the previous round with the vote cast in the second round. It should reject the vote if the vote in the second round does not match with the fingerprint (figure 5). Using such a system limits the effect of strategic voting based on live election results, because the electorate is bound to their vote cast earlier.

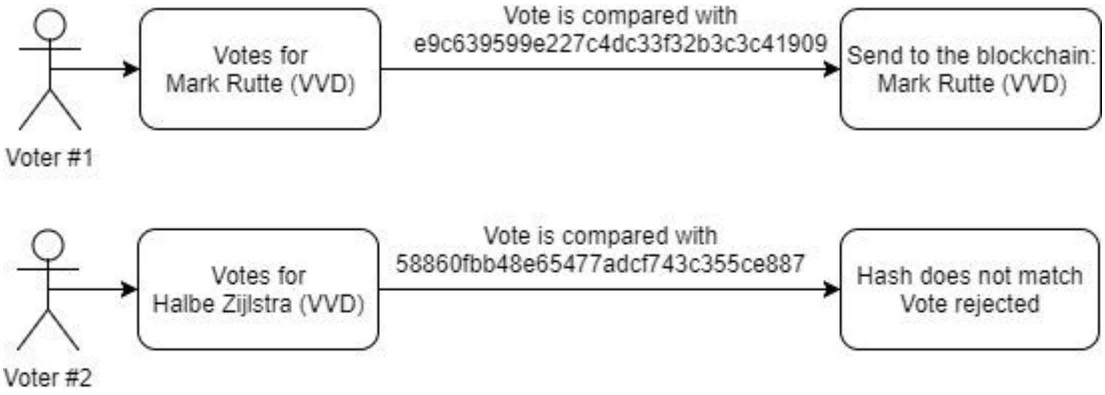

Figure 5 Voting in the second round

One problem remains. Those who observe the blockchain can still predict the result of the election based on the first round, because hash algorithms are publicly available. Malicious actors could create a table with data and the corresponding hash value. Such tables are called rainbow tables. The contents of the hashed blockchain is readable if a rainbow table is used (Kumar H., Kumar, S., Joseph, Kumar, D. et al., 2013). Websites storing login details suffer from the same issue. Hackers may gain access to the hashed user passwords if the security of a website is breached and the database is leaked. Simple passwords such as *123456* or *password* are likely to appear in these rainbow tables. It takes a long time to brute force these passwords hashes to find the original password, but searching for the hashes in a precomputed table is likely to take only a few seconds. Websites therefore add a 'salt' to the password to mitigate the threat of hackers using a rainbow table (Boonkrong and Somboonpattanakit, 2015). A salt is a random set of characters added to the password to make them unique, resulting in a unique hash as well.

For development of the blockchain ballot the same technique is applicable. The computer should add random data to the vote before sending it to the blockchain. On the contrary, the random data should be the same for the same voter in both round. Verification to check whether the same vote is committed is otherwise impossible. Therefore, using the private key the voter used to cast the vote seems an obvious choice to use as salt. Thus, if the same key is used for voting, the hashes should match. The voter does not have to remember an additional password.

Oostveen and Van den Besselaar (2004) recommended the use of a voter-verified audit trail to improve the trustworthiness of the elections. A verifiable audit trail suggests that the voting machine should print the cast vote. If these votes are stored in a ballot box, a voter verifiable trail can be used to audit the election result.

Figure 6 illustrates the flow of Ethereum based elections.

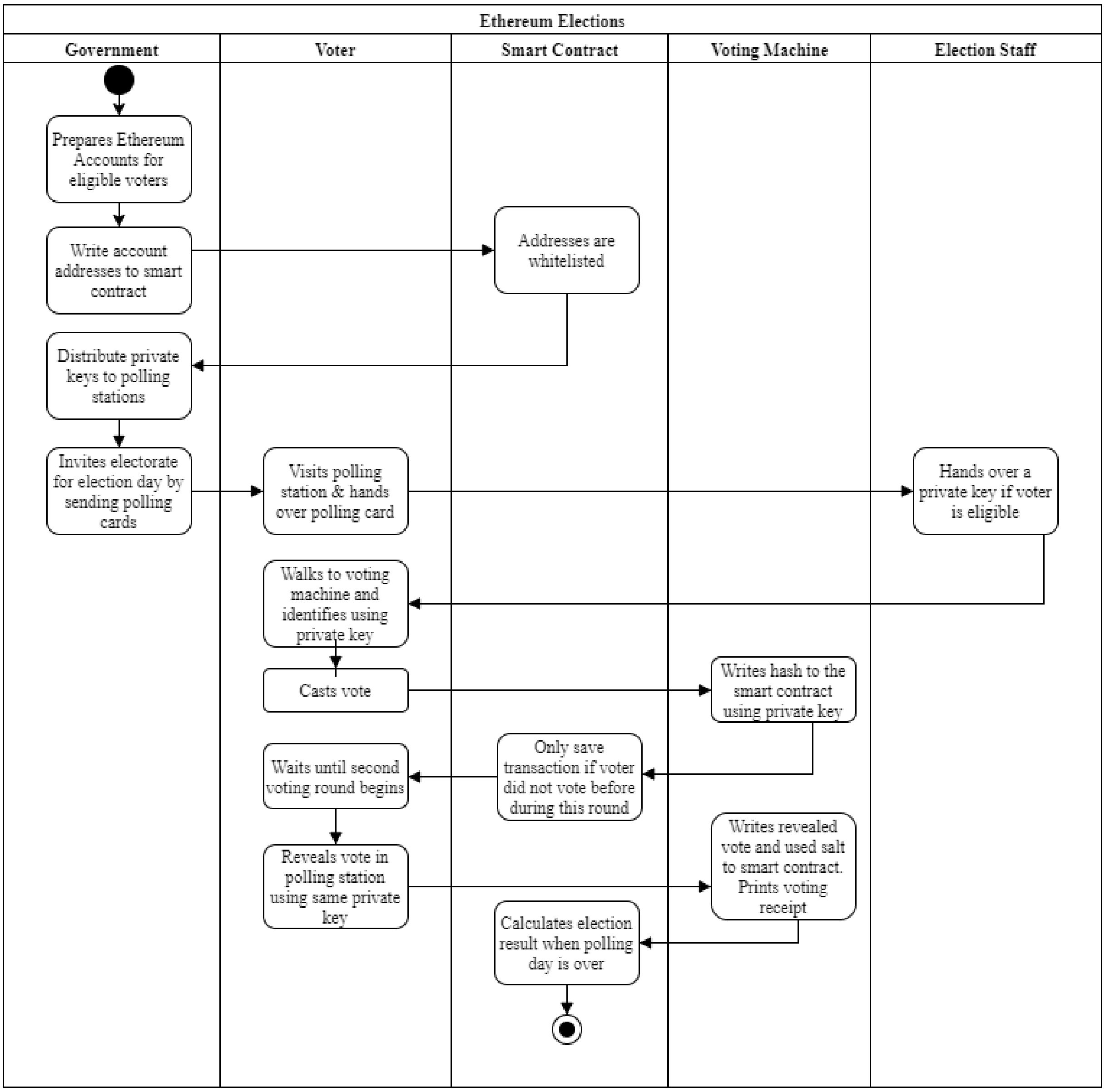
Ethereum Elections
Government
Voter
Smart Contract
Voting Machine
Election Staff
Prepares Ethereum Accounts for eligible voters
Write account addresses to smart contract
Addresses are whitelisted
Distribute private keys to polling stations
Invites electorate for election day by sending polling cards
Visits polling station & hands over polling card
Hands over a private key if voter is eligible
Walks to voting machine and identifies using private key
Casts vote
Writes hash to the smart contract using private key
Only save transaction if voter did not vote before during this round
Waits until second voting round begins
Reveals vote in polling station using same private key
Writes revealed vote and used salt to smart contract. Prints voting receipt
Calculates election result when polling day is over

Figure 6 Ethereum Election Activity Diagram

4.1.3 Wrap up

To summarize, a whitelist of eligible voting keys is used to make sure citizens can only vote once. Furthermore, ballot secrecy is guaranteed by using a voting key provided to the voter when authenticated. A link between de voter and their vote does not exist. Uncertain election results are assured, due to the hashing technique. This is further secured by applying a salt.

The table below (table 4) provides an overview of the compatibility of Ethereum elections with the election requirements.

Election requirement	**Requirement compliant?**	**Elaboration**
Equal suffrage	✓	• Ballot card needed to authenticate • Voting key given after authentication • Fraud with voting keys prevented by adding up unused voting keys and ballot cards
Ballot secrecy	✓	• Separation between voter and vote
Citizen right to vote	✓	• Only citizens authenticated by polling officials can vote
Access to information	–	• Blockchain is viewable by anyone • Voter-verified audit trail • But: people may not understand blockchain
Uncertain election results	✓	• Hashing techniques used to prevent real-time election status
Access to polling station & ballot box	✗	• Electorate must visit polling station twice.
No tampering, manipulation or destructions of ballots	✗	• Blockchain is write-only ledger • But: software on the surface can still manipulate the vote before adding it to the blockchain.
Legend: ✓ Compliant with requirement – disputable ✗ Not compliant with requirement		

Table 4: Overview of compatibility with electoral requirements

Ethereum based elections require the users to cast one vote twice, to guarantee uncertain election results. Similar solutions found by researchers (McCorry, Shahandashti and Hao, 2017) and programmers (Dimitrova, 2016) suffered from the same flaw. Asking voters to vote twice seriously limits the accessibility of the ballot. Voters might not have enough time or are immobilized. Moreover, the model does not consider what happens if citizens voted in the first round but neglect to vote in round two or lose their private key.

Another flaw is the possibility of vote manipulation. Even though the blockchain seems rather secure, client applications are not. The same security issues found in the election client used for voting in Estonia (Springal et al., 2014) are applicable to all kind of issues. Gonggrijp and Hengeveld (2007) also provide evidence supporting these claims. Public blockchains provide support to check the vote afterwards, potentially leading to coercion as voters can proof to an extortionist that their demand is fulfilled.

4.2 Private blockchain

The major flaw in public blockchain based election is that the nature of a public blockchain allows people to retrieve a live result of the polls during election day. A private or a consortium blockchain might offer a solution for this, because this allows developers to impose permissions and access restrictions on the chain (Otte, de Vos and Pouwelse, 2017). It seems rather obvious that a government itself should not control such a blockchain as this gives an incumbent government running for a re-election the opportunity to cheat. A more fundamental solution is a blockchain consortium hosted by a group of countries who share the same values and goals. For elections held in the Netherlands, the European Union seems a logical first choice. Other intergovernmental organisations such as the United Nations might also suffice.

Open source software is available to create a more private blockchain. Hyperledger Fabric, part of the Hyperledger project, support the key building blocks for creating an election system. According to Cachin (2016), Hyperledger Fabric controls the nodes participating in the validation. These nodes are the validating peers. The validating peers deal with transactions (Li, Sforzin, Fedorov and Karame, 2017). Non-validating peer nodes take part in a Hyperledger blockchain as well. They act as a proxy allowing application and services to connect to the network (Cachin, 2016). As a result of managed nodes, permissions are applicable to the chain. Furthermore, Hyperledger Fabric supports smart contracts, also called *chaincode*, implying that code can run on the chain. Just like the Ethereum based elections, the code should make sure that voters can only cast a vote once and that their vote is correctly registered.

Several documentations about the Hyperledger Fabric blockchain suggest that applications based on this type of blockchain consists of two architectural sides (Gulhane and Hoyt, 2017). The first side is the inner side. This is where the nodes running the blockchain belong. The outer side is where the applications communicating with the blockchain reside. In the context of elections, the voting machines reside in the outer side. This architecture is illustrated in figure 7.

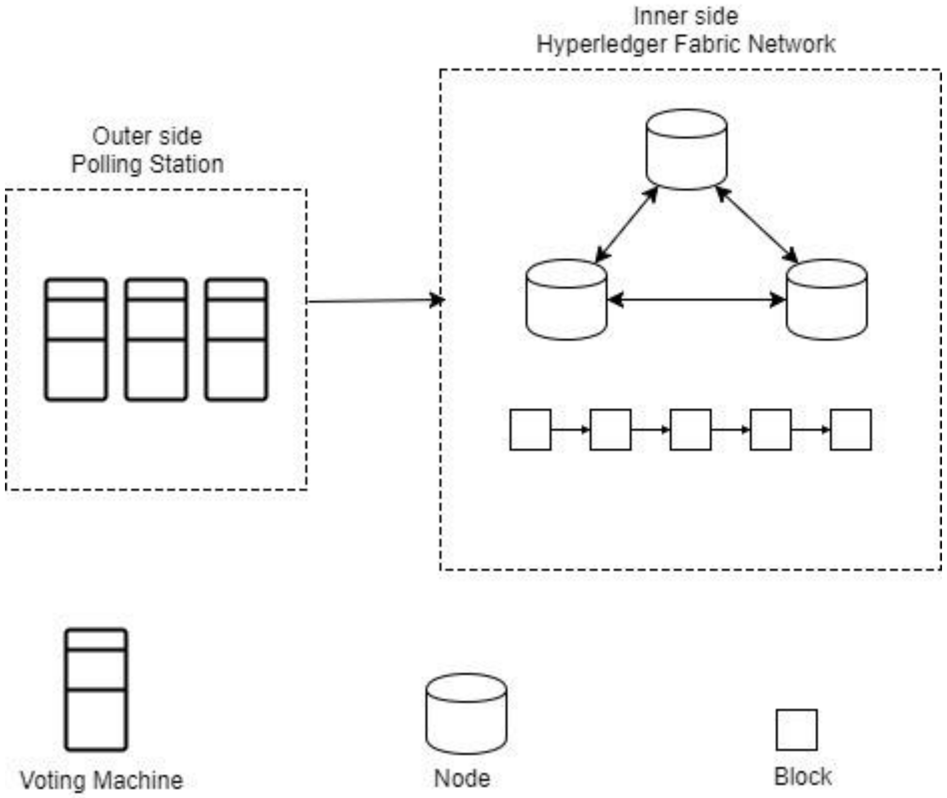

Figure 7 Architecture Hyperledger Fabric based application

Having an inner side suggests that the voting computers cannot write nor read to the blockchain directly, because the computers are not part of this chain. The Hyperledger Fabric documentation (“Writing Your First Application”, n.d.) suggests using an endpoint in the inner side of the architecture. The voting machines can then query to network through this endpoint to send or receive information. The information send by the voting machines should at least contain:

(1) A unique key. This key is handed over by elections officials to the voter after the voter is authenticated.
(2) The vote cast by the voter.

Just like the Ethereum based elections, a voting key is used to remove the link between the voter and their vote. Moreover, the same keys are used to prevent voting multiple times, because the blockchain can store the key on a list of used keys. Similarly, the blockchain can hold a list of keys usable for voting. Therefore, people who try to vote twice are blocked from voting the second time, because their key has been marked as used. Likewise, those trying to outsmart the system and try to make up their own key are blocked, as this key is not on the whitelist.

4.2.1 Lack of transparency

The blockchain application as described is not publicly viewable. As a consequence, complicated workarounds to guarantee an uncertain election results such as hashes are not needed. This means that voters do not have to cast their vote in two rounds unlike the Ethereum approach. Although it solves the accessibility problem the Ethereum approach suffered from, other problems arise:

(1) Having a private or consortium blockchain means that only eligible nodes can see or alter the ledger. This implies that voters cannot see what happens between the moment their vote is cast and during the result. A black box like this makes the election process less transparent as it should be, especially true if compared with paper based voting in the Netherlands as anyone is allowed to keep an eye on the process.

(2) The consortium hosting the private blockchain can still access the data, which implies that some people can obtain knowledge about the parties which are likely going to win the elections, whereas some people don’t. Moreover, the people having access to the ballot data can decide to leak these results.

(3) The Electoral Council or government should decide who hosts the blockchain. The European Union seems like a first safe choice, because the Union is a group of democratic countries who share the same values. However, the EU is a political union as well. National elections of EU member states affect how the European Council is composed, because only the heads of an EU-government are allowed to take part in the council meetings. Since the European Council gives political direction to the EU, it gives an incentive to the EU to alter the election results in favour of political parties supporting the EU project.

(4) Some parties are campaigning for leaving the European Union. In case of the United Kingdom, they succeeded. These parties are not going to accept elections hosted by the EU, nor will their electorate.

(5) If elections are hosted by other organisations than the EU, the same problem still arises. For instance, if elections are hosted by the United Nations or the NATO, or even the incumbent parties in parliament, an incentive to cheat is still apparent.

(6) Companies such as EY, Deloitte, KPMG or PwC can certify the election process. The main question here is if the electorate chooses to believe these companies.

For these reasons, a private or consortium blockchain are not likely to be an ideal tool if used for counting the votes. The lack of transparency of election systems based on private blockchain are an easy target for critics. Even if the organisations hosting the elections are in good faith, the trust in the outcome can easily be undermined by someone running for a seat in parliament. Stating that the election was not won, because "the elite" cheated by abusing the lack of transparency would be enough, even if it is not true. Electoral systems are only as good as the electorate believes it to be.

Figure 8 illustrates the flow of Hyperledger Fabric based elections.

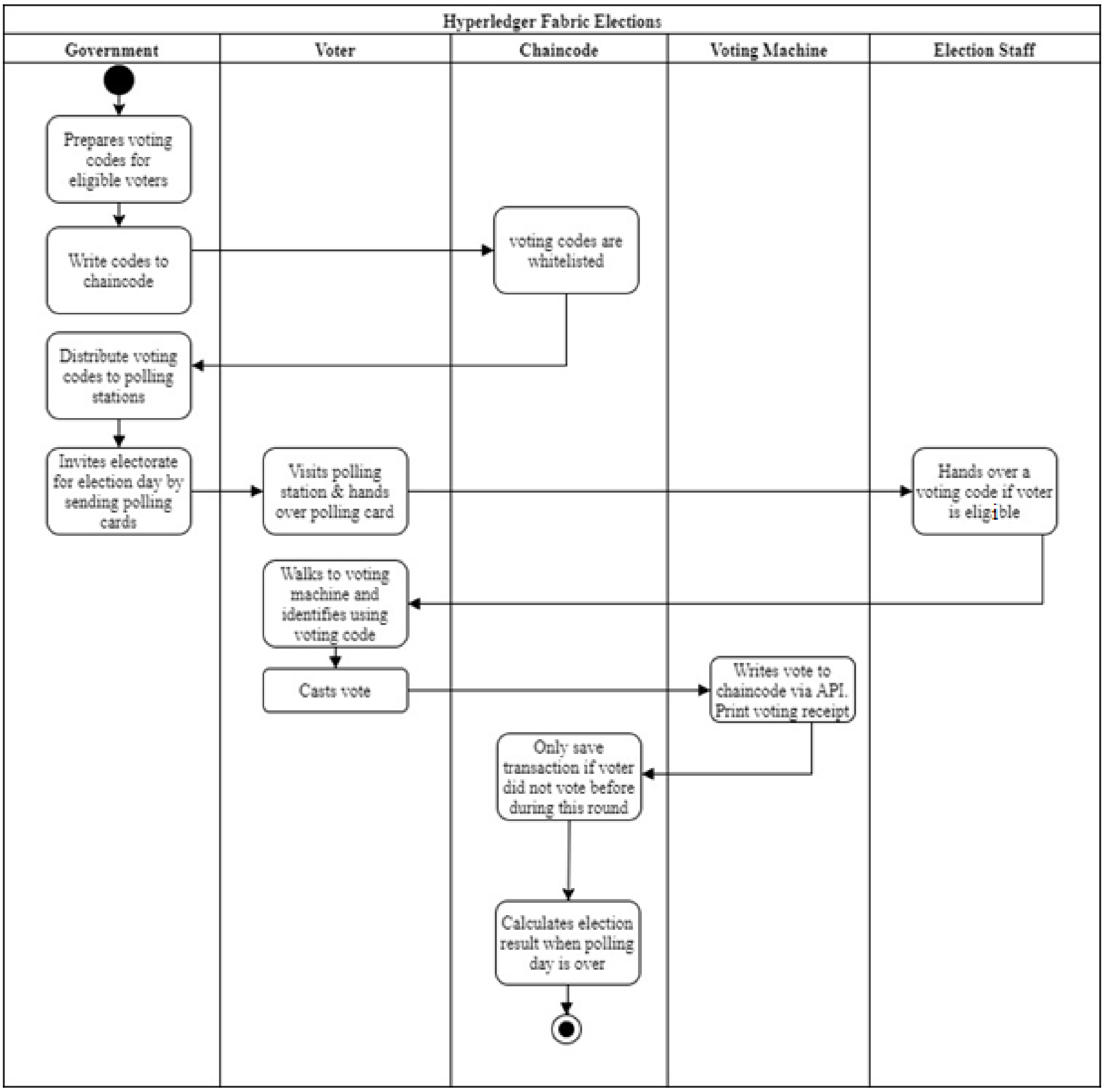
Hyperledger Fabric Elections
Government
Voter
Chaincode
Voting Machine
Election Staff
Prepares voting codes for eligible voters
Write codes to chaincode
voting codes are whitelisted
Distribute voting codes to polling stations
Invites electorate for election day by sending polling cards
Visits polling station & hands over polling card
Hands over a voting code if voter is eligible
Walks to voting machine and identifies using voting code
Casts vote
Writes vote to chaincode via API. Print voting receipt
Only save transaction if voter did not vote before during this round
Calculates election result when polling day is over

Figure 8 Hyperledger Fabric Election Activity Diagram

4.2.2 Wrap up

The table below (table 5) provides an overview of the compatibility of Hyperledger Fabric based elections with the election requirements.

Election requirement	Requirement compliant?	Elaboration
Equal suffrage	✓	- Ballot card needed to authenticate - Voting key given after authentication
Ballot secrecy	✓	- Separation between voter and vote
Citizen right to vote	✓	- Only citizens authenticated by polling officials can vote
Access to information	✗	- Private or consortium blockchains are a black box. The electorate does not know what happens between casting a vote and the announcement of the result.
Uncertain election results	-	- The ballot box data is not available. - But: Those who have access to the data can leak the results.
Access to polling station & ballot box	✓	- Only one voting round is needed.
No tampering, manipulation or destructions of ballots	✗	- Blockchain is write-only ledger - But: software on the surface can still manipulate the vote before adding it to the blockchain.
Legend: ✓ Compliant with requirement – disputable ✗ Not compliant with requirement		

Table 5: Overview of compatibility with electoral requirements

4.3 Fully internet-based voting (public blockchain)

The first election model based on the public blockchain Ethereum is not accessible, because the government cannot ask from the electorate to visit a polling station twice. However, it is possible to reduce the amount of effort required from the electorate. Instead of physical polling station, the government could opt for an approach where people can cast their vote on the internet, requiring a new authentication model. In the more traditional voting models, the voter is authenticated by election officials by providing a polling card and a card proving their identity. An internet based election model should eliminate the need for physically attending a polling station. The government could use DigiD for authentication. DigiD is already used to identify for governmental websites (DigiD, n.d.). Figure 9 illustrates the activities affected by internet based voting.

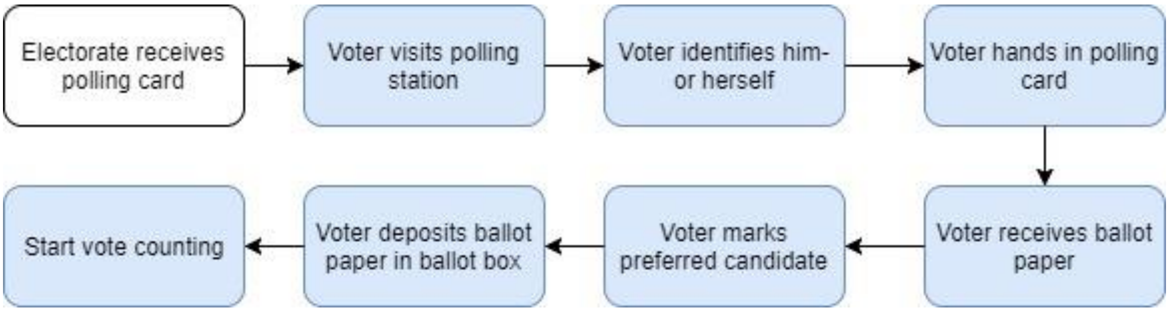
Electorate receives polling card
Voter visits polling station
Voter identifies him- or herself
Voter hands in polling card
Voter receives ballot paper
Voter marks preferred candidate
Voter deposits ballot paper in ballot box
Start vote counting

Figure 9 Election Process Activities

In this model, the polling station is replaced by a polling website. The electorate can visit this website on election day and authenticate using DigiD. Once authenticated, the website displays the list of parties and people running for a seat. The voter can cast a vote by clicking on the preferred candidate. Moreover, a private key is needed to cast a vote. The key is used to make sure a voter can only cast one vote, enforceable by a smart contract. The webserver can retrieve a key by randomly selecting one from a backend database and store the value in a local browser cookie. Therefore, an election official handing over a key is not needed in this model. A citizen can then cast a vote by simply selecting a candidate after which the decision is saved. The website can store the vote on the Ethereum blockchain, just like a model using traditional voting machines. There are two methods to achieve this:

(1) The hash of the vote and the key is send to the website server. The key is sent as plain text to the server as well. An Ethereum client is installed on this server. This allows the server to write the vote to an Ethereum blockchain.

(2) The website writes the hash of the vote and the key directly to the Ethereum blockchain using the JavaScript API. This requires the voter to install a program on their computer though (JavaScript API, n.d.).

The second method is more decentralized and therefore more preferred. Using a single node to act as client for millions of users would defeat the purpose of a decentralized public blockchain.

Both methods write a salted hash to the blockchain. Since no viable way of retrieving the original contents of the vote exists, the electorate must cast the vote in two rounds. The first round is used to cast their vote, the second round to reveal their vote. Because a hash is already stored on the blockchain, the electorate can no longer change their vote based on early result estimates.

To guarantee the electoral requirement of equal suffrage, the system must make sure a voter can only cast one vote. Yet, people with access to the database should not have the possibility to check what a person voted nor if a vote has been cast at all. Rössler (2009) suggests storing a hash of the citizen ID if a vote is cast. Since a hash is a one-way function, those with database access cannot tell which hash belongs to who. However, if a person logs in into the system, it is possible to check if the storage contains the hash of their ID. Storing the citizen ID hash is not directly applicable in the Netherlands. Rössler designed the system based on the idea that his government does not own a register of these ID's. The ID's are stored on a card the citizen owns. The Dutch government does own such a register. This means that the government could check who voted by simply calculating the hashes of the unique ID's and compare it with the hashed storage.

Solutions for this problem exist. As described before, salts (a unique set of characters) make it impossible to create a calculated hash table. A possible candidate for a salt is the password[1]. If the user authenticates for voting on the voting website, the website should store a hash calculated from the password used to log in and the citizen ID. The website can then simply check if it should provide a key for voting or not. Using a password as a salt has a downside though. DigiD should prevent users from changing their password on election day. Changing the password results in a new hash, making it possible for the electorate to cast an infinite number of votes by changing the password after voting.

If this election model is chosen and put in place, critics are likely to target a few domains of the system:

(1) Using DigiD to authenticate means that a part of the process is centralized. Therefore, attackers are presumably going to target DigiD if they want to sabotage the election. For

[1] The assumption is made that DigiD does only store a hash of the password in the database, instead of plain-text. Otherwise, people with access to the database can still generate a table.

instance, centralized digital processes are vulnerable to a Distributed Denial of Service (DDoS) attacks (Singh, Dhindsa, Bhushan, 2017). A malicious large computer network can perform a huge amount of login actions on the DigiD service.

Authentic users can therefore no longer access DigiD, because the servers are too busy processing the requests of the attacker.

Other problems can arise for a centralized process besides a DDoS attack. Examples are a local power outage or loss of internet. A single point of failure makes the election system less reliable.

(2) Votes are cast on the private computers of the electorate, implying that the electorate needs a computer with internet access. This might be a problem for computer illiterate people. Furthermore, if users vote at home they cannot deposit a receipt of the vote in a box for audit trail purposes.

(3) The private keys are stored on the computer. The voter loses the privilege to vote if a program running on this computer wipes the key before the vote is cast, because these keys are only handed out once. The same problem would arise if their computer becomes inaccessible during election day, due to ransomware or a computer crash.

(4) People can challenge the legitimacy of the election, because they can claim the government violates ballot secrecy. There is no way for the electorate to check if the government does not store a link between an identify and a vote. Certifications by auditors would not help, because people can challenge the legitimacy of these companies as well. Note that these claims do not even need to be objective.

(5) There are two rounds of voting. The first round to commit the vote and the second round to reveal the vote. Even though this limits the accessibility less than the Ethereum model with physical polling stations, it may still be too complex for some citizens.

(6) The keys used to vote are stored in central location: a database. Someone who obtains access to this database could abuse these keys.

(7) It is not possible to check whether a voter is intimidated into a certain vote.

For these reasons, public blockchain elections over the internet are not the ideal method to hold elections.

Figure 10 illustrates the flow of Ethereum based elections using a website instead of a physical polling station.

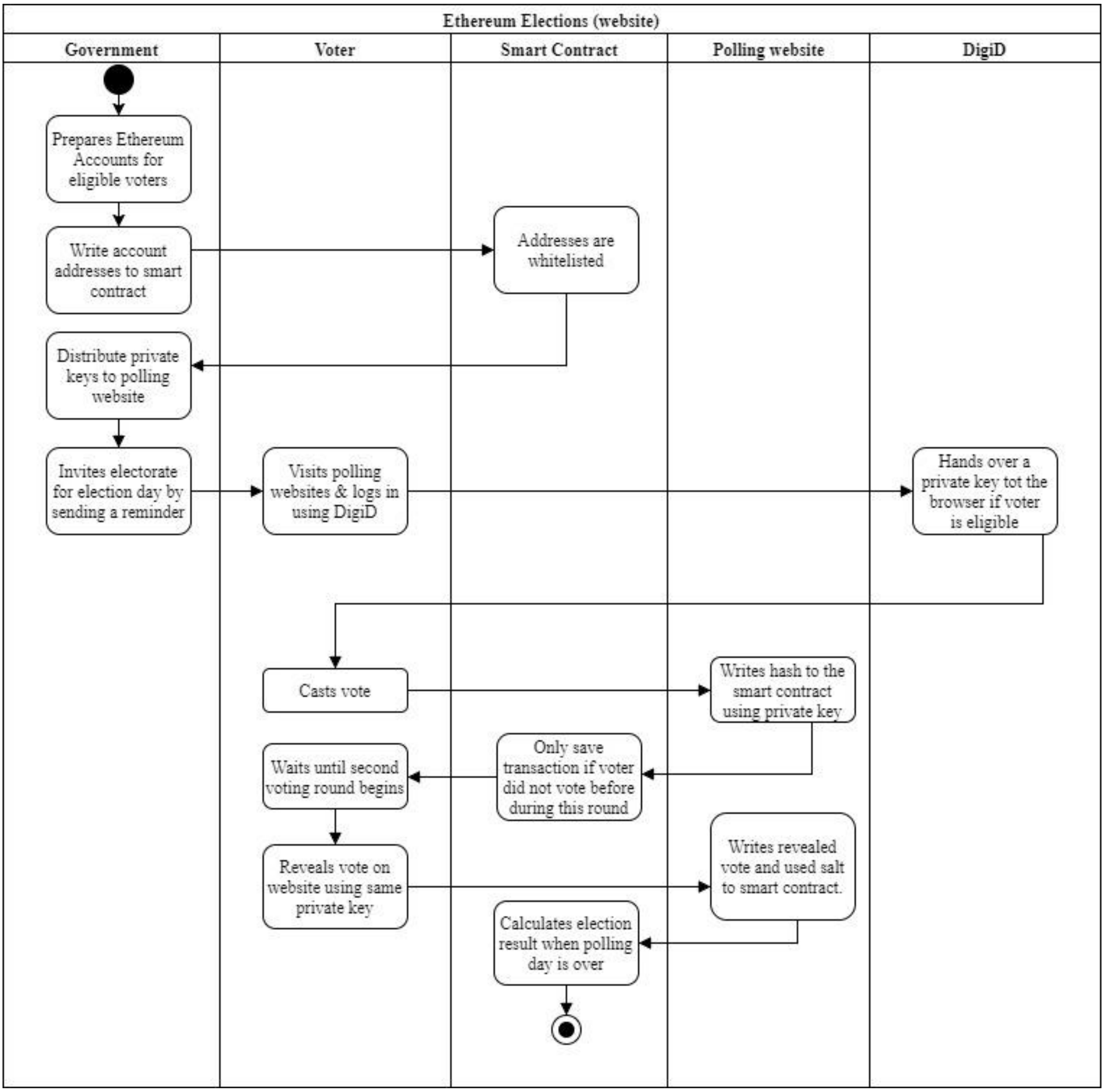
Ethereum Elections (website)
Government
Voter
Smart Contract
Polling website
DigiD
Prepares Ethereum Accounts for eligible voters
Write account addresses to smart contract
Addresses are whitelisted
Distribute private keys to polling website
Invites electorate for election day by sending a reminder
Visits polling websites & logs in using DigiD
Hands over a private key tot the browser if voter is eligible
Casts vote
Writes hash to the smart contract using private key
Only save transaction if voter did not vote before during this round
Waits until second voting round begins
Reveals vote on website using same private key
Writes revealed vote and used salt to smart contract.
Calculates election result when polling day is over

Figure 10 DigiD/Ethereum Election Activity Diagram

4.3.1 Wrap up

The table below (table 6) provides an overview of compatibility of Ethereum based elections over the internet with the election requirements.

Election requirement	**Requirement compliant?**	**Elaboration**
Equal suffrage	✓	• DigiD used to authenticate • Voting key given after authentication
Ballot secrecy	-	• Separation between voter and vote • But: people can question this separation, because the government can secretly link these domains.
Citizen right to vote	✓	• Only citizens authenticated by the system can vote
Access to information	✗	• The application services built on top of the blockchain are not transparent. There is no way of checking if the government violates ballot secrecy.
Uncertain election results	✓	• Hashing techniques used to prevent real-time election status
Access to polling station & ballot box	-	• Electorate must vote twice, but do not have to go to a physical polling station. This could be a problem for less digital savvy people.
No tampering, manipulation or destructions of ballots	✗	• Blockchain is write-only ledger • But: software on the surface can still manipulate the vote before adding it to the blockchain.
Legend: ✓ Compliant with requirement − disputable ✗ Not compliant with requirement		

Table 6: Overview of compatibility with electoral requirements

5 Conclusions and recommendations

This chapter describes the findings of this study.

5.1 Discussion

None of the designed models are fully compliant with requirements for free and fair elections described in table 1.

(1) The election method based on the public blockchain gives away the voting results during election day. Whereas the nature of public blockchains is that they are fully transparent, the same transparency causes incompatibility with the 'uncertain election result' requirement.

(2) A slightly modified version of the same model solves this problem. Hashes are introduced to preserve the uncertainty of the election results. However, applying hashing techniques introduce another problem. Hashing functions are one-way only and therefore not reversible. Thus, this altered model introduces a second voting round. The first round to cast the vote, the second round to reveal the vote. The uncertainty of the election result is guaranteed using this technique, because the electorate can no longer change their vote. The consequence of this change is that citizens have to visit the polling stations twice. Since people may suffer from immobility or lack of time, this model affects the accessibility.

(3) Experts suggest altering the public blockchain model in such a way that the voting machines keep the votes off-chain and temporarily store them on the voting machine. Once polling day is over, the machines should write the data to the blockchain. This ensures that the uncertainty of the election result is guaranteed. To prevent malicious actors from tampering with the votes, the electorate should have the possibility to verify their vote. Verifying is achievable by comparing the hash of the vote cast with the vote stored on the blockchain. Experts suggested to allow this in polling stations only, to prevent coercion. However, this still violates the criteria related to transparency. The voting machine can display a "vote confirmed" message on the screen even though the hashes do not match.

(4) Elections based on a private blockchain mitigate the compatibility problems with accessibility and uncertain election results. The electorate only needs to visit a polling station once, because the data of the blockchain is kept private during election day. Whereas this looks like a good idea at first sight, it introduces another compatibility

problem. The electorate does not know what happens with the data as long as it is kept private. Some people working at the organisation hosting the private blockchain can still access the data, or leak it. Furthermore, organisations may have an inventive to alter the election result. For example, the European Union could favour pro-EU parties. Because of the lack of transparency, it is harder to counter these claims.

(5) A possible way to vote is over the internet. This model is based on a public blockchain. Therefore, similar hashing techniques as describer earlier must be used, implying that two voting rounds are required: one round to a cast the vote and one to reveal the vote. It should cause less accessibility problems than model 1, because the electorate does not have to physically visit a polling station twice. Yet, critics are likely to argue that the system is not accessible enough because it still requires two voting rounds. Moreover, computer illiterate people may not understand the digital aspects of the system.

    Besides the accessibility issues, internet based voting introduces transparency issues as well. There is no way to check if the government does not store a link between the voter and their vote. Even if the government is in good faith, people can claim the government violates ballot secrecy. Therefore, this model is unusable because of the lack of transparency.

Blockchain based elections seem to introduce conflicting aspects. Public blockchains are transparent, because citizens can see what happens. Yet, the election results should remain unknown until election day is over. Countermeasures such as hashing techniques are applicable to counter this problem, but this limits the accessibility of the elections. Private blockchains eliminate the need of hashing techniques, but eliminate transparency as well.

This reminds of a *devil's triangle* model, where only two out of three variables can be chosen. The third variable is severely limited or missing entirely. The triangle is illustrated in figure 11.

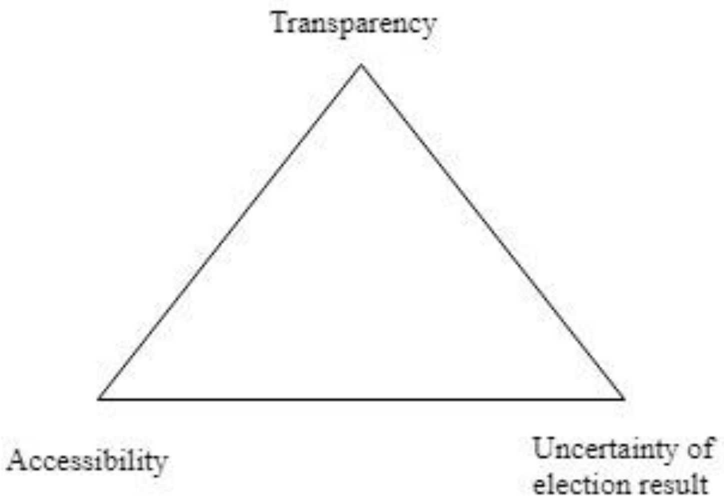

Figure 11 The Devil's Triangle

All the election scenarios based on blockchain suffered from the trade-off depicted in the devil's triangle. People designing other models are also likely forced to make a choice between these electoral requirements.

The problems found with the election scenarios are not limited to the devil's triangle. The potential vulnerabilities detected in the client program used during the Estonian elections are a threat as well. A voter can cast a vote on the European People Party, but the blockchain client can write a vote for another party to the blockchain. This violates the election requirement that no tampering, manipulation or destructions of ballots should occur. Therefore, whereas a blockchain is rather safe and secure, the surface of a blockchain remains vulnerable. This poses a larger threat in comparison with voting on paper. Manipulating paper based elections demands the assistance of multiple people across the country. On the other hand, manipulating computer based elections only requires one rogue entity. The vendor of the voting software could have built in a cheating component. A malware attack can also achieve this. Gonggrijp and Hengeveld (2007) suggested parallel testing to detect inconsistencies with voting software, but openly questioned whether such tests are sufficient enough. The Volkswagen emission scandal suggests that it is indeed possible to cheat during such tests.

Several authors therefore suggest the use of a voter verified audit trail. This implies that the voting machine must print the cast vote. The voter should then store the printed vote in a ballot box. Such auditing trails render e-voting methods where the electorate does not have to visit a physical polling station impossible. Moreover, the main question is what would happen if the blockchain proclaims a winner and the paper trails proclaims another. There is no point in organising computer based elections if parliament decides that the paper vote is leading and vice versa. Furthermore, a difference in the results gives stakeholders the opportunity to challenge the legitimacy.

5.2 Limitations

At the time of writing, blockchain ballots are also constrained by the scalability of the blockchain. Well known blockchain networks can only achieve a theoretical limit of 66,7 transactions per second. This implies that it takes more than two days to register the votes of all the eligible voters in the Netherlands. It takes more than four days if two voting rounds are needed to guarantee the uncertainty of the election result. Luckily, blockchain develops fast. Experts believe the scalability problem of blockchains are solved in the foreseeable future. Another limitation is the small number of interviewed experts. However, the findings were confirmed by experts in writings, therefore it is unlikely that more interviews would yield different findings.

5.3 Recommendation & future research

The findings of this study imply that a blockchain ballot is not suitable as backbone for vote counting during free and fair elections. It is also likely that voting machines in general are not suitable for electoral usage, because these are prone to malicious software as well. Therefore, future research could focus on a better voting procedure. For instance, the errors made during the last Dutch elections might have been caused because the ballot papers are rather large.

5.4 Conclusion

It is not possible to organize blockchain based elections without violating at least one of the criteria for free and fair elections. Furthermore, malicious actors can compromise the result of the election, because client software running on the voting machine is able to alter votes. A blockchain ballot does not necessarily eliminate the ‘Byzantine Generals’ problem; it merely moves it to another layer. Since the attack surface to manipulate digital ballots is much wider, it is probably less secure than the paper based ballots. Similarly, the introduction of a blockchain ballot would create tensions between several criteria for free and fair elections, such as transparency of the ballot and uncertain election results. The purpose of this study is to figure out whether a blockchain ballot is an electoral enhancement or a danger to democracy. The findings suggest the former is not applicable, since the electoral criteria are not met using blockchain.

6 References

Al Hasib, A., & Haque, A. A. M. M. (2008, November). A comparative study of the performance and security issues of AES and RSA cryptography. In *Convergence and Hybrid Information Technology, 2008. ICCIT'08. Third International Conference on* (Vol. 2, pp. 505-510). IEEE.

Alvarez, R. M., Hall, T. E., & Trechsel, A. H. (2009). Internet voting in comparative perspective: the case of Estonia. *PS: Political Science & Politics*, *42*(3), 497-505.

Anthes, G. (2015). Estonia: a model for e-government. *Communications of the ACM*, *58*(6), 18-20.

Bazhanov, B. G. (1980). Vospominaniia byvshego sekretaria Stalina. Tretia volna.

Badev, A. I., & Chen, M. (2014). Bitcoin: Technical background and data analysis.

Bentov, I., Gabizon, A., & Mizrahi, A. (2016). Cryptocurrencies without proof of work. In *International Conference on Financial Cryptography and Data Security* (pp. 142-157). Springer Berlin Heidelberg.

Bishop, S., & Hoeffler, A. (2016). Free and fair elections: A new database. *Journal Of Peace Research*, *53*(4), 608-616.

Bratus, S., Lembree, A., & Shubina, A. (2010, June). Software on the witness stand: what should it take for us to trust it?. In *International Conference on Trust and Trustworthy Computing* (pp. 396-416). Springer, Berlin, Heidelberg.

Buterin, V. (2014). A next-generation smart contract and decentralized application platform. *white paper*.

Blackwelder, B., Coleman, K., Colunga-Santoyo, S., Harrison, J. S., & Wozniak, D. (2016). The Volkswagen Scandal.

Boonkrong, S., & Somboonpattanakit, C. (2015). Dynamic Salt Generation and Placement for Secure Password Storing.

Cachin, C. (2016). Architecture of the Hyperledger blockchain fabric. In *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*.

Cachin, C., Schubert, S., & Vukolić, M. (2016). Non-determinism in byzantine fault-tolerant replication. *arXiv preprint arXiv:1603.07351.*

Cheibub, J. A., Gandhi, J., & Vreeland, J. R. (2010). Democracy and dictatorship revisited. *Public choice, 143*(1-2), 67-101.

Chen, T. (2010). Stuxnet, the real start of cyber warfare? *IEEE Network, 24*(6), 2-3.

Christidis, K., & Devetsikiotis, M. (2016). Blockchains and smart contracts for the internet of things. *IEEE Access, 4*, 2292-2303.

Croman, K., Decker, C., Eyal, I., Gencer, A. E., Juels, A., Kosba, A., & Song, D. (2016, February). On scaling decentralized blockchains. In *International Conference on Financial Cryptography and Data Security* (pp. 106-125). Springer Berlin Heidelberg.

Davis-Roberts, A., & Carroll, D. J. (2010). Using international law to assess elections. *Democratization, 17*(3), 416-441.

Decker, C., & Wattenhofer, R. (2013). Information propagation in the bitcoin network. In *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on* (pp. 1-10). IEEE.

The Economist. (2015). *Democracy in an age of anxiety.* Retrieved from https://www.yabiladi.com/img/content/EIU-Democracy-Index-2015.pdf

Delfs, H., & Knebl, H. (2007). Symmetric-key encryption. *Introduction to Cryptography*, 11-31.

Delmolino, K., Arnett, M., Kosba, A., Miller, A., & Shi, E. (2015). A programmer's guide to ethereum and serpent. Retrieved 22 October 2017, from*: https://mc2-umd. github. io/ethereumlab/docs/serpent_tutorial. pdf.*

DigiD. (n.d.). Over DigiD. Retrieved October 31, 2017, from https://www.digid.nl/over-digid/

Dimitrova, E. (2016). Token-weighted voting implementation. Retrieved October 25, 2017, from https://blog.colony.io/token-weighted-voting-implementation-part-2-13e490fe1b8a

Document of the Copenhagen Meeting of the Conference on the Human Dimension of the CSCE. (1990). *Osce.org*. Retrieved 6 April 2017, from http://www.osce.org/odihr/elections/14304

Douceur, J. R. (2002, March). The sybil attack. In *International Workshop on Peer-to-Peer Systems* (pp. 251-260). Springer, Berlin, Heidelberg.

Elklit, J., & Svensson, P. (1997). What Makes Elections Free and Fair?. *Journal Of Democracy*, *8*(3), 32-46.

Engberts, M. (2011). *Van stembus naar uitslag: gegevensintegriteit verkiezingsproces* (pp. 16-18). Retrieved from http://www.ru.nl/publish/pages/769526/m_engberts_scriptie.pdf

Eyal, I., & Sirer, E. G. (2014). Majority is not enough: Bitcoin mining is vulnerable. In *International conference on financial cryptography and data security* (pp. 436-454). Springer, Berlin, Heidelberg.

Faulconbridge, G. (2017). *Venezuelan election turnout figures manipulated by one million votes: election company*. *Reuters*. Retrieved 17 August 2017, from https://www.reuters.com/article/us-venezuela-politics-vote-smartmatic-idUSKBN1AI1KZ

Gerber, A. S., Huber, G. A., Doherty, D., Dowling, C. M., & Hill, S. J. (2013). Do perceptions of ballot secrecy influence turnout? Results from a field experiment. *American Journal of Political Science*, *57*(3), 537-551.

Gervais, A., Karame, G. O., Wüst, K., Glykantzis, V., Ritzdorf, H., & Capkun, S. (2016). On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (pp. 3-16).* ACM.

Gringer, D. (2008). Why the national popular vote plan is the wrong way to abolish the Electoral College. *Columbia Law Review*, 182-230.

Gonggrijp, R., & Hengeveld, W. J. (2007). Studying the Nedap/Groenendaal ES3B voting computer.

Goodwin-Gill, G. S. (2006). *Free and fair elections*. Inter-Parliamentary Union.

Gujrathi, S. (2014). Heartbleed bug: AnOpenSSL heartbeat vulnerability. *International Journal of Computer Science and Engine ter Science and Engineering*, *2*(5), 61-64.

Gulhane, I., & Hoyt, K. (2017, May 5). Create a web-based to-do list application using Hyperledger Fabric V1.0. Retrieved October 31, 2017, from https://developer.ibm.com/code/patterns/create-a-to-do-list-app-using-blockchain/

Heckelman, J. C. (1995). The effect of the secret ballot on voter turnout rates. *Public Choice, 82*(1), 107-124.

Huijgen, P. E. M. (2006). Ook deugdelijke stemmachines deugen niet.

Hevner, A., & March, S. (2004). Design Science in Information Systems Research. *MIS Quarterly*, *28*(1), 75-105.

Introduction to Smart Contracts. (n.d.). Retrieved October 23, 2017, from http://solidity.readthedocs.io/en/develop/introduction-to-smart-contracts.html

Jacobs, B., & Pieters, W. (2009). Electronic Voting in the Netherlands: From Early Adoption to Early Abolishment. *Foundations Of Security Analysis And Design V,* 121-144.

JavaScript API. (n.d.). Retrieved November 1, 2017, from https://github.com/ethereum/wiki/wiki/JavaScript-API

Kamerbrief over fout bij invoeren uitslagen Tweede Kamerverkiezing. (2017). *Rijksoverheid.nl*. Retrieved 18 June 2017, from https://www.rijksoverheid.nl/ministeries/ministerie-van-binnenlandse-zaken-en-koninkrijksrelaties/documenten/kamerstukken/2017/06/06/kamerbrief-over-fout-bij-invoeren-uitslagen-tweede-kamerverkiezing

Kaushal, R. (2016). Bitcoin: Vulnerabilities and Attacks. *Imperial Journal of Interdisciplinary Research, 2*(7).

Kiayias, A., Russell, A., David, B., & Oliynykov, R. (2017, August). Ouroboros: A provably secure proof-of-stake blockchain protocol. In *Annual International Cryptology Conference* (pp. 357-388). Springer, Cham.

Kiesraad. (n.d.). Instructiefilms programma 4 en 5. Retrieved from https://www.kiesraad.nl/verkiezingen/osv-en-eml/ondersteunende-software-verkiezingen-osv/instructiefilms-programma-4-en-5

King, S., & Nadal, S. (2012). Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *self-published paper, August, 19.*

Kumar, H., Kumar, S., Joseph, R., Kumar, D., Singh, S. K. S., & Kumar, P. (2013, April). Rainbow table to crack password using MD5 hashing algorithm. In *Information & Communication Technologies (ICT), 2013 IEEE Conference on* (pp. 433-439). IEEE.

Lamport, L., Shostak, R., & Pease, M. (1982). The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS), 4*(3), 382-401.

Lauer, T. W. (2004). The risk of e-voting. *Electronic Journal of E-government, 2*(3), 177-186.

Lin, I. C., & Liao, T. C. (2017). A Survey of Blockchain Security Issues and Challenges. *IJ Network Security, 19*(5), 653-659.

Li, W., Andreina, S., Bohli, J. M., & Karame, G. (2017). Securing Proof-of-Stake Blockchain Protocols. *In Data Privacy Management, Cryptocurrencies and Blockchain Technology* (pp. 297-315). Springer, Cham.

Li, W., Sforzin, A., Fedorov, S., & Karame, G. O. (2017). Towards scalable and private industrial blockchains. In *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts* (pp. 9-14). ACM.

Lee, K., James, J. I., Ejeta, T. G., & Kim, H. (2016). Electronic Voting Service Using Block-Chain. *The Journal of Digital Forensics, Security and Law: JDFSL, 11*(2), 123.

López, V., & Brodzinsky, S. (2017). Venezuela to vote amid crisis: all you need to know. the Guardian. Retrieved 17 August 2017, from https://www.theguardian.com/world/2017/jul/25/venezuela-elections-all-you-need-to-know

McCorry, P., Shahandashti, S. F., & Hao, F. (2017). A Smart Contract for Boardroom Voting with Maximum Voter Privacy. *IACR Cryptology ePrint Archive, 2017*, 110.

Mercuri, R. (2002). A better ballot box?. *IEEE spectrum, 39*(10), 46-50.

Met potlood stemmen onveilig: verkiezingsuitslag eenvoudig te hacken. (2017). *RTL Nieuws*. Retrieved 12 April 2017, from https://www.rtlnieuws.nl/nederland/politiek/met-potlood-stemmen-onveilig-verkiezingsuitslag-eenvoudig-te-hacken

Naor, M., & Yung, M. (1989). Universal one-way hash functions and their cryptographic applications. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing* (pp. 33-43). ACM.

O'Dwyer, K. J., & Malone, D. (2014). Bitcoin mining and its energy footprint.

Oostveen, A. M., & Van den Besselaar, P. (2004). Security as belief: user’s perceptions on the security of electronic voting systems. *Electronic voting in Europe: Technology, law, politics and society*, 47, 73-82.

OSV en EML. (2017). *Kiesraad.nl*. Retrieved 6 April 2017, from https://www.kiesraad.nl/verkiezingen/inhoud/osv-en-eml

Otte, P., de Vos, M., & Pouwelse, J. (2017). TrustChain: A Sybil-resistant scalable blockchain. *Future Generation Computer Systems*.

Park, J. H., & Park, J. H. (2017). Blockchain Security in Cloud Computing: Use Cases, Challenges, and Solutions. *Symmetry*, *9*(8), 164.

Pass, R., Seeman, L., & Shelat, A. (2017). Analysis of the blockchain protocol in asynchronous networks. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques* (pp. 643-673). Springer, Cham.

Rampton, R. & Volz, D. (2016, November 27). *Trump, without evidence, says illegal voting cost him U.S. popular vote*. *Reuters*. Retrieved 18 July 2017, from http://www.reuters.com/article/us-usa-trump-votes-idUSKBN13M0XZ

Peters, G. W., & Panayi, E. (2015). Understanding modern banking ledgers through blockchain technologies: Future of transaction processing and smart contracts on the internet of money. In *Banking Beyond Banks and Money* (pp. 239-278). Springer International Publishing.

Regeling - Kieswet. (2017). Wetten.overheid.nl. Retrieved 29 August 2017, from http://wetten.overheid.nl/BWBR0004627/2017-06-10#AfdelingII_HoofdstukE_Paragraaf1_ArtikelE1

Reid, F., & Harrigan, M. (2013). An analysis of anonymity in the bitcoin system. In *Security and privacy in social networks* (pp. 197-223). Springer New York.

Ritzen, G. (2017). *Kabinet: stemmen worden bij verkiezingen met de hand geteld*. *nrc.nl*. Retrieved 6 April 2017, from https://www.nrc.nl/nieuws/2017/02/01/kabinet-stemmen-worden-bij-verkiezingen-met-de-hand-geteld-a1544017

Rössler, T. (2009). Electronic Voting Using Identity Domain Separation and Hardware Security Modules. *Software Services for e-Business and e-Society*, 1-12.

Sako, K. (2005). Public Key Cryptography. *Encyclopedia of Cryptography and Security*, 487-488.

Schryen, G., & Rich, E. (2009). Security in large-scale internet elections: a retrospective analysis of elections in Estonia, the Netherlands, and Switzerland. *IEEE Transactions on Information Forensics and Security, 4*(4), 729-744.

Sekaran, U., & Bougie, R. (2016). *Research methods for business: A skill building approach*. John Wiley & Sons.

Singh, K., Dhindsa, K. S., & Bhushan, B. (2017). Distributed Defense: An Edge over Centralized Defense against DDos Attacks. *International Journal of Computer Network and Information Security*, *9*(3), 36.

Sompolinsky, Y., & Zohar, A. (2015). Secure high-rate transaction processing in bitcoin. In *International Conference on Financial Cryptography and Data Security* (pp. 507-527). Springer, Berlin, Heidelberg.

Springall, D., Finkenauer, T., Durumeric, Z., Kitcat, J., Hursti, H., MacAlpine, M., & Halderman, J. A. (2014). Security analysis of the Estonian internet voting system. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* (pp. 703-715). ACM.

Stembureau-instructie voor de dag van de stemming. (2017). Retrieved from https://www.stembureauinstructie.nl/Handout_Stembureau-instructie_voor_de_dag_van_de_stemming.pdf

Sullivan, C., & Burger, E. (2017). E-residency and blockchain. *Computer Law & Security Review.*

Swan, M. (2015). *Blockchain* (1st ed.). Sebastopol, Calif: O'Reilly.

Tattersall, N. (2017). *Turkish electoral board head says 'yes' campaign won referendum. Reuters.* Retrieved 19 July 2017, from http://www.reuters.com/article/us-turkey-referendum-board-idUSKBN17I0TE

Tikhomirov, S. (2017). Ethereum: state of knowledge and research perspectives.

Tomuschat, C. (2008). International covenant on civil and political rights. *United Nations.*

Van der Parre, H. (2017). *Gemeenten willen graag terug naar computerstemmen*. *Nos.nl*. Retrieved 6 April 2017, from http://nos.nl/artikel/2165112-gemeenten-willen-graag-terug-naar-computerstemmen.html

Van Heese, R. (2007, September 28). Het rode potlood is voor even terug. Retrieved October 26, 2017, from https://www.trouw.nl/home/het-rode-potlood-is-voor-even-terug~a27f803f/

Vasin, P. (2014). Blackcoin’s proof-of-stake protocol v2.

Vranken, H. (2017). Sustainability of bitcoin and blockchains. *Current Opinion in Environmental Sustainability*, 28, 1-9.

Vrees voor hackers: kabinet schrapt software, stemmen tellen volledig met de hand. (2017). *RTL Nieuws*. Retrieved 6 April 2017, from https://www.rtlnieuws.nl/nederland/politiek/vrees-voor-hackers-kabinet-schrapt-software-stemmen-tellen-volledig-met-de-hand

Wang, X., & Yu, H. (2005). How to break MD5 and other hash functions. In *Eurocrypt* (Vol. 3494, pp. 19-35).

Wij vertrouwen stemcomputers niet. (2009, June 4). Vraag en Antwoord. Retrieved October 26, 2017, from http://wijvertrouwenstemcomputersniet.nl/Vraag_en_Antwoord

Wood, G. (2014). Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 151.

Writing Your First Application. (n.d.). Retrieved October 31, 2017, from http://hyperledger-fabric.readthedocs.io/en/latest/write_first_app.html

Wüst, K., & Gervais, A. (2017). Do you need a Blockchain? *IACR Cryptology ePrint Archive, 201*7, 375.

Zhang, F., Cecchetti, E., Croman, K., Juels, A., & Shi, E. (2016). Town crier: An authenticated data feed for smart contracts. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (pp. 270-282). ACM.

Appendix A

Interview with Jan Willem Barnhoorn (translated from Dutch)

Q: What you think of blockchain-powered elections?

A: Blockchain can indeed support the ballot on election day. The models described are an example. The security really depends on the consensus model used. There are technologies to enforce these security models on hardware level. The company Intel has tools for this. This implies that you must trust Intel though.

Q: Would you prefer public or permissioned blockchains if used for electoral purposes?

A: Seeing that two voting rounds are needed to guarantee an uncertain election result and to prevent bias, a public blockchain doesn't seem like a viable solution. You can host a private blockchain where, for instance, all the countries in the world host a validating node. If you still want to stick with a public blockchain you can temporary store the votes off-chain and write them to the blockchain once election day is over. To prevent people from tampering with the votes the electorate should somehow be able to check whether their vote has been processed correctly. Yet, you want to avoid coercion. Therefore, I think it should only be possible to check your vote in the polling station.

Q: Blockchain is known for its hardened security. Are the blockchain clients as equally secure?

A: No, they are not. I'd argue that the client is not part of the blockchain protocol. Therefore, it is not a blockchain problem. Yet, you cannot use the blockchain without a client so the problem does relate to blockchain. Testing the clients might help to prevent this.

Q: How you think that the scalability constraints of blockchain can be resolved to make the technology applicable for national scale elections?

A: Private blockchains do not really suffer from this. Public blockchains do but they are working on techniques to improve this. For instance, Ethereum has an off-chain solution to improve scalability. It's called Raiden. Temporary storing the votes off-chain and writing them to the blockchain at once as I just said also helps.

Q: Most public blockchain clients have a limited transaction per second throughput. Are there any other limitations which may hinder national scale elections?

A: I can’t think of any.

Q: The literature describes different requirements elections have to meet. Accessibility, transparency and an uncertain result of the elections seems to conflict if blockchain is applied to the election process. What do you think about this trade-off?

A: It seems that these criteria affect each other indeed. Temporary storing the votes on the machines and writing them to the blockchain at once might help. It should be possible to verify the votes without coercion though.