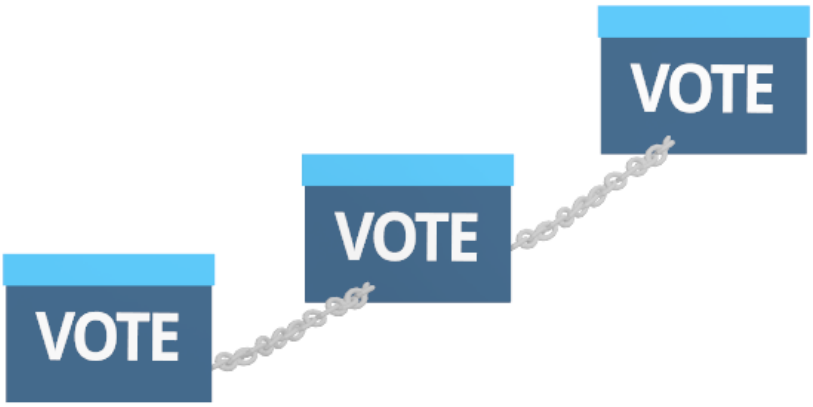


A Decentralized Voting System

Group DATX02-19-85

Bachelor of Science Thesis in Computer Science

Jack Ahlkvist
Anton Gustafsson
Carl Lundborg
Joakim Mattsson Thorell
Aron Sandstedt
Sanjin Slavnic

Department of Computer Science and Engineering
Chalmers University of Technology
Gothenburg, Sweden 2019

Bachelor of Science Thesis

A Decentralized Voting System

DeVote

Jack Ahlkvist
Anton Gustafsson
Carl Lundborg
Joakim Mattsson Thorell
Aron Sandstedt
Sanjin Slavnic

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
Gothenburg, Sweden 2019

A Decentralized Voting System

Jack Ahlkvist
Anton Gustafsson
Carl Lundborg
Joakim Mattsson Thorell
Aron Sandstedt
Sanjin Slavnic

Supervisor: Christos Profentzas , Department of Computer Science and Engineering
Examiner: Miquel Pericas, Department of Computer Science and Engineering

Bachelor of Science Thesis
Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover illustration: By the authors.

Gothenburg, Sweden 2019

Abstract

As the area of use for blockchain technology is rapidly spreading, a digitized democratic voting system might just be the next revolutionary step towards a transparent and trusted electoral system.

This thesis investigates the possibility of a decentralized voting system and explores the possible challenges regarding privacy, correctness and integrity. Using the Ethereum blockchain and various smart contracts, a prototype was implemented as a proof of concept. The thesis also contains studies regarding the social- and environmental aspects of the prototype and electronic voting in general.

Throughout the project, it became clear that the suggested implementation has essential tradeoffs and would not work in practice. The study concretizes what needs to be improved in order to use a voting system built on blockchain technology. This report concludes that there is still work to be done in order to use this technology in crucial fields such as voting. And suggests using a private blockchain in order to implement the specified voting system, DeVote.

Sammandrag

Då användningsområdet för blockkedje-teknologi blir allt bredare, är ett digitaliserat demokratiskt röstningssystem kanske nästa revolutionära steg framåt till ett transparent och pålitligt röstningssystem.

Denna rapport undersöker möjligheterna för ett decentraliserat röstningssystem och utforskar de eventuella problem som integritet, korrekthet och avskildhet bär med sig. Med användning av Ethereums blockkedje-teknologi och olika *smart contracts* implementerades en prototyp som ett *Proof of concept*. Rapporten innehåller även idéer och studier angående hållbar utveckling och sociala aspekter kring den föreslagna prototypen och elektronisk röstning i allmänhet.

Genom projektets gång blev det uppenbart att den föreslagna idéen har fundamentala avvägningar och inte kommer fungera i praktiken. Denna studie konkretiserar vad som måste förbättras för att kunna skapa ett röstningssystem byggt på blockkedje-teknologi. Rapporten konkluderar att det fortfarande krävs mer arbete för att kunna realisera denna teknologi i så pass avgörande områden som just röstning. Och föreslår att använda en privat blockkedja till att implementera det refererade röstningssystemet, DeVote.

General terms and abbreviations

API
Application Programming Interface

Back-end
Server-side, refers to things the user can't see e.g. databases and servers.

Bitcoin
Worlds largest cryptocurrency.

Blockchain
Decentralized data storage.

CI
Continuous Integration, used for automate testing when pushing to master branch.

CLI
Command-line interface.

Consensus mechanism
Protocol used to synchronize all nodes on the blockchain.

DAPP
Decentralized Applications, e.g. the suggested prototype.

Decentralized system
A system with no central authority.

DeVote
The decentralized voting system that was implemented for this thesis

ECDSA
Elliptic Curve Digital Signature Algorithm

ethash
A hashing algorithm used in Ethereum blockchain

Ethereum
Blockchain based platform.

Smart contract
Protocol for digitally verifying negotiations of a contract.

Front-end
Client-side, everything involved with what the user sees.

Genesis block
First block of a blockchain.

GUI
Graphical user interface.

Hash function
Algorithm that converts data in a one way manner.

Keccak
A primitive cryptographic family which hold members of the Secure Hash Algorithm family

Linter
Software for analyzing code for potential errors.

Mining
Process used to create blocks for the blockchain.

Nonce
Essentially a piece of data that is time consuming to compute but easy to verify

npm
Node.js package manager, package manager for JavaScript.

Peer-to-Peer network
Non-hierarchical network where nodes communicate directly between each other.

PoS Proof of stake, compared to the Proof of Work, where the creator of a new block is chosen in a deterministic way, depending on its wealth.

PoW
Proof of Work. Essentially a protocol that by computational power states which block to trust.

Proof of concept
Realization of an idea, used to demonstrate its utility.

secp256k1 curve
Parameters of the elliptic curve used in Bitcoin's public-key cryptography

Solidity
Programming language used for implementing smart contracts and designed to be used on Ethereums Virtual Machine.

TTP
Trusted Third Party

UTXO
Unspent Transaction Outputs is a the output of a bitcoin transaction that a user receives and is able to spend in the future.

Webhook
User-defined HTTP callbacks.

Contents

1 Introduction

The most fundamental aspect of a democracy is the availability for citizens to not only share ideas, opinions, and beliefs but to make their individual voices heard by deciding the collective future by vote. However, for the voting to proceed as intended, there needs to be a transparent and secure process where also the voters knowingly keep their privacy. The challenge is to find a solution that prevents unlawful manipulation of the collected data and achieve desired transparency in the security measures, taken to protect voter privacy and the collected results and therefore democracy itself.

1.1 Background

The challenge with today's common practice of paper-based voting is that the voters in question have to trust both a public authority for publishing the results and their local vote-collecting groups for collecting the votes. In this it is easy to find elements that could compromise democracy; from the human factor of the collectors during vote collection that could range from personal interests to trivial mistakes, or a dishonest authority for publishing false results. Even if the published results were to theoretically differentiate only a little from the actual public opinion, it is still imperative that the individual voters should have their votes recorded without compromise.

In order to ensure correct vote collection, one could propose some kind of e-voting service that would be easy to use and would remove the need for human vote collectors. However, the issue with having to trust a single third party would still stand if the collected data were to only be stored at a central database where some, probably well-known user is responsible for maintenance. What is sought after is a public database, where vote collection is transparent and stored votes immutable. Additional to this, other challenges would be introduced as well: would this system be vulnerable to attacks and would the users keep their privacy?

The problem with a voting system being dependant on some third party is that the election falls under their control and becomes centralized. This creates a single point of failure, which increases not only the probability of abuse but also online vulnerability to attacks and physical vulnerability (e.g. natural disasters, sabotage, etc.) since main servers, and backup systems, are physically accessible [1]. These factors could compromise democracy and jeopardize the privacy of the voters, therefore the database should be distributed but also without anything connecting votes to individuals directly.

The ideal solution would be an easy to use e-voting system that does not depend on humans for vote collection, nor a trusted third party for controlling the collected data, whilst still staying immune to attacks and guarantee user privacy.

When Satoshi Nakamoto released the Bitcoin protocol, the goal was to propose a system for electronic transactions, without needing a financial institution as a trusted third party throughout the payment process; instead, all transactions would be saved on a public, immutable and replicated database known as blockchain [2].

A possible way to reach the previously described ideal voting/election system could be by using the same blockchain technology, like in the Bitcoin protocol proposed by Satoshi Nakamoto. Research is needed about this technology in order to decide whether it is feasible to create a decentralized voting system and successfully exclude the otherwise necessary TTP for transparency,

ensuring security and not jeopardize voter privacy.

1.2 Purpose

The goal of the thesis is to research the possibility to design a decentralized e-voting system and implement a prototype as a proof of concept that assures transparency, privacy, correctness, and integrity.

1.3 Problem definition

Decentralized e-voting is preferably executed online rather than traditional voting. This introduces various, distinctive challenges concerning primarily transparency, privacy, correctness, and integrity. It is crucial to understand that solving these fundamental challenges is an open research question. Nonetheless, for this report, it is assumed that blockchain technology provides transparency and immutability due to its cryptographic properties and can offer insights into some of the challenges. Therefore, it is a feasible foundation for the intended system. However, blockchain alone might not provide the optimal solution for an electronic voting system.

In order to narrow down the subject and define a primary set of questions correlated to the research, one main challenge will consequently be introduced. The main challenge will additionally be systematically broken further down into subtasks, each constrained and defined in order to facilitate a quantitative research and generate a proof of concept which is stated as the following.

How does one implement a minimum valuable decentralized voting system, utilizing blockchain technology and being capable of handling voting, showing votes and verifying correctness?

This challenge consequently harvests several concerns. The central research will nonetheless define three requirements for the intended voting system. The requirements are constructed as subtasks of the central task, namely the challenge of building a decentralized voting system.

The subtasks incorporate the following areas of research, all of which define the area of research with specific questions.

- **Privacy**

  It is essential to keep an individual's vote secret. That is, voter's vote should not be revealed to anyone at any step. The system already leverages cryptographic properties of blockchain to achieve security. However, the non-traceability of a vote is non-trivial.

  Transactions to the blockchain produce metadata. Statistical analysis will reveal sensitive information, even if the data itself is encrypted, making pattern recognition possible [3]. It is therefore essential to assure privacy for the system and assure that votes are untraceable. This naturally generates a question of formulation:

  How does one assure privacy?

- **Integrity**

In computer science, integrity generally relates to the assurance for the accuracy and consistency of data over its entire life-cycle. The voting system should supposedly not allow unauthorized modification of the electoral results. In addition, the system should exclusively assure authorized voters to vote only once. Voters are therefore required to use unique identifiers to assert the right to vote.

This precondition naturally produces additional concerns. However, this thesis will consider a fundamental challenge as a result of this requirement, namely:

How does one assure integrity?

- **Correctness**

  Votes should supposedly not be modified, forged, or deleted. The system should ensure and prove to a voter that all votes were counted correctly. This requirement naturally brings forwards the question:

  How does one assure correctness?

1.4 Delimitations

Limiting the discussion regarding the prototype implementation, introduced in section 1.2, will be of importance since there are many parts of the prototype that lies outside of the scope of the project but are still necessary for it to work. Some details regarding the code for the prototype will not be included since it will be trivial to developers and hard to grasp to non-developers reading the report.

For the blockchain part, there will be no details regarding hash functions and cryptography. The discussion will touch upon these subjects by explaining in short how the principle work, however, the report will not go in depth about it and will not prove that it works since this is a field that is very large, and only a small understanding of how it works is needed for the success of the project. The focus will instead be on privacy regarding the users of the system and not on the security regarding the infrastructure of blockchains. The scope of privacy will namely be about integrity and correctness in regards to how safe it is for the user to use the service without being identified, and additionally how safe and trustworthy it is to use blockchains without it being tampered with. Statistical analysis and other means that can be used to reveal an identity within blockchains are subjects that will be touched upon but will not be something that the discussion will go into great detail about since it belongs more to the part of cryptography which we have chosen not to focus on.

Limitations will also be regarding the implementation and discussion of laws to be considered when it comes to voting. The general focus and the discussion will lie within Swedish elections and the laws regarding the Swedish electoral system. To include other voting systems outside of elections would be unnecessary and to include discussions towards a global solution regarding elections would be too big of a subject. The discussion around laws will not be in professional detail since that is not the main goal. The discussion will instead center around the plausibility of replacing the current voting system in Sweden with a decentralized one. What laws that this will interfere with are of no great importance.

When it comes to issues and questions regarding trust and ethical solutions, these will also be limited to Swedish elections. The focus will additionally be around the Swedish infrastructure and systems. Furthermore, environmental questions will be towards how much an environmental impact that an election in Sweden will have using the proposed solution versus the current impact. Since this is not the main focus, rather a smaller part of the research, the data gathered will be limited and thus making the conclusion regarding the subject limited.

2 Technical Background

This section introduces and establishes the fundamental technical theory of blockchain technology; particularly Ethereum which was the central framework implemented in order to build a minimum valuable decentralized voting system. Basically, all relevant concepts and terms regarding blockchain technology and the necessary surrounding technicalities are explained.

2.1 Blockchain

[Fler källor i blockchain-sektionen om det går] "*A blockchain is a distributed data structure that is replicated and shared among the members of a network. It was introduced with Bitcoin to solve the double-spending problem*" [4].

Blockchains are essentially decentralized, distributed, public ledgers. The ledger is organized into blocks, illustrated in Figure 1, where each block is a digital piece of information about transactions. Each block characteristically contains a cryptographic hash of the previous block to assure there is a standard order to the blocks. This links the blocks and builds symbolically a chain and gives the blockchain two essential properties; a modification of the earlier block will invalidate all the previous, and anyone can verify the whole chain given the first genesis block.

Blocks are created based on the latest block of the most current chain and are processed by nodes in a Peer-to-Peer network. Every creation follows a consensus mechanism, which is essentially a protocol, in order to agree which transactions are legitimate and added to the blockchain. This is feasible with cryptography and necessary to assure correctness. The protocol assures that everyone has the same pieces of information about the transactions.

Information on the blockchain is transparent and anyone can view the content of a blockchain. Transactions on the blockchain are not completely anonymous. However, information about the users is limited to their digital signature.

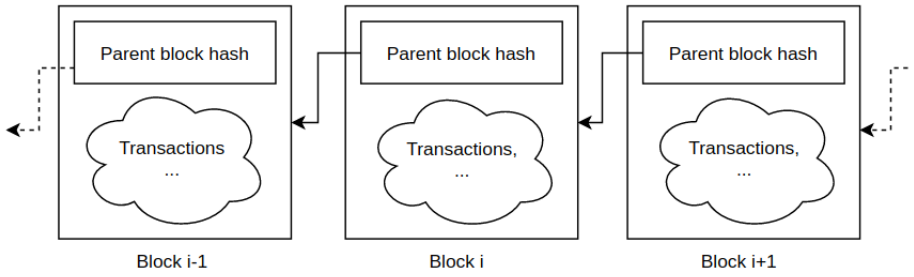


Figure 1: Abstract blockchain visualization, where each cloud represents digital information about transactions. Fundamentally, this information varies and is stored and represented in different forms of data structures. Depending on the blockchain, this might vary from more basic Bitcoin blocks to complex Ethereum blocks.

2.1.1 Transactions

A transaction is a set of instructions for modifying the state of the blockchain. Transaction fees are a part of public blockchain networks in order to have the transactions processed by specialized nodes, or so-called *miners*, and confirmed by the network (See 2.1.4). Miners perform computational work for a financial reward.

In order to fundamentally understand what a state of a blockchain is, it is necessary to shortly introduce Unspent Transaction Outputs (UTXOs for short) for the following reason.

In Bitcoin, the transfer of value is actualized through transactions. Bitcoin's state is represented by its global collection of UTXOs, that is, Bitcoin does not maintain user account balances. A user's account balance is the sum total of each individual UTXO, for which that user holds the corresponding private key. In contrast, there are other blockchains that are able to manage account balances and more, for example Ethereum which is a transaction-based state machine, meaning that all transaction-based state machine concepts may be built on it. This concept is made clear in chapter 2.2.

2.1.2 Digital wallets

Each user that wants to make a transaction is associated with an asymmetric key pair, a private key, and a public key. The public key is associated with a *digital wallet* which serves as an address to the user, with no connection to the user's identity. It is public, available to anyone. The private key, on the other hand, is kept secret to assure only the owner of that wallet can sign valid transactions.

Transactions are considered valid when they are signed with the user's private key, providing a digital signature which makes it unfeasible to forge. Transactions are therefore unique and only the owner can generate the unique digital signature.

2.1.3 Miners

As mentioned, transactions are only valid when they are signed by the sender. They are broadcasted to the network where they are collected and packed into blocks by miners, who then, partially given the public key attached to each transaction, validate the signatures and build the next block. Hence, blocks are built by miners on the network, solving asymmetric, non-deterministic puzzles by means of a consensus mechanism (See 2.1.4). This was originally addressed in the Bitcoin paper and solves the famous double-spending problem, which is basically the risk that a digital currency can be spent twice [5].

Fundamentally, Bitcoin solved the double-spending problem and can as a consequence assure integrity, at least to a certain extent. Hypothetically, a group of miners could potentially control more than 50% of the network computing power and forge the blockchain. This group would then gain a monopoly over the blockchain and be able to prevent new transactions, only allowing certain users to transact or forge transactions. This hypothesis is called *51%-Attack*.

2.1.4 Consensus mechanism

In order to make sure that everyone on the network is recording the same transactions and in the same order, different blockchains provide a specific protocol to reach consensus, intuitively named consensus protocols. Basically, these protocols exist to make sure all nodes in the Peer-to-Peer network are synchronized. In Bitcoin and Ethereum, a block is only considered valid if it has a *Proof Of Work*. This means that the network follows a set of rules which state to trust the block which has the most computational work put into it. This base of trust makes fraud computationally infeasible.

To make this type of protocol possible each block stores a variance called *nonce* (number used once). It differentiates the block from other blocks and makes each block unique when it puts through some hashing algorithm. Therefore, due to the nonce, the hash output of the contents of the block will change.

The nonce is used to find a valid hash, which is difficult, for a new block. This is where computational work comes into play. Miners must find a nonce value that, when plugged into the specific hashing algorithm, generates an output that meets certain requirements, namely a certain number of leading zeros of the hash. Miners have to use brute force as a consequence of the fundamentals of hashing in order to find the exact nonce that will output a valid value. This randomizes which miner ends up solving the block and adding it to the most recent block in the chain. This also gives the blockchain interesting properties, namely that it takes a lot of time to find a valid output but it is easy to verify if the output is correct.

However, different consensus protocols have distinct rules to reach consensus; e.g. Ethereum is going to transition from *Proof Of Work* to *Proof Of Stake* [6] protocol. This protocol is based on the number of coins inside the system instead of computation-based proofing. Stakeholders whose tokens are locked in a specialized wallet receive a dividend based on the amount of staked cryptocurrency.

"In a PoS system, a blockchain appends and agrees on new blocks through a process where anyone who holds coins inside of the system can participate, and the influence an agent has is proportional to the number of coins (or "stake") it holds" [7].

2.1.5 Public and private blockchains

A public blockchain is a blockchain that anyone can read, send transactions to and expect to see them included if they are valid and participate in the consensus process. Public blockchains are secured by crypto economics, that is, the consensus mechanism and cost of transactions. This comes essentially down to that the amount of economic resources miners can have is proportional to the degree of influence they can have in the consensus process.

Besides public blockchain, there are also consortium blockchains, where the consensus process is controlled by a pre-selected set of nodes. They fall in the category of private blockchains where write-permissions are kept centralized to one organization and read permissions may be public or restricted.

2.2 Ethereum

Shortly after blockchain technology was introduced and implemented with Bitcoin, a new era of digital currency was born, giving rise to crypto economics, which Ethereum originates from. However, it is an adaptation of its core properties with the purpose to create a decentralized network with memory, for a whole variety of other applications. Ethereum is a general purpose blockchain that understands a general-purpose programming language. It is able to store data and is capable of enforcing the protocol's correct execution [8].

Ethereum was released in 2015 and became a framework for applications that were in need of decentralization and a concept of shared memory. It allows developers to write *smart contracts* and put them on the blockchain.

2.2.1 Smart contracts

The idea behind a smart contract was originally introduced in 1997 by Nick Szabo [9]. The concept is analogous to the notion of a vending machine. It is a device which implements the conditions of an agreement. Essentially, a user inputs a given amount of currency which yields a specific output; contrary, a user does not input the given amount which yields no output [10]. This is the core idea behind Ethereum. It is basically a software containing rules for negotiating the terms of the contract which govern the behavior of accounts within the Ethereum state. Since the smart contract of Ethereum is implemented on the blockchain, the contract is visible to all the users.

2.2.2 Solidity

In order to build such contracts there is a specialized programming language named Solidity. It is a high-level, object-oriented language for implementing smart contracts that are represented similarly to classes in object-oriented languages. The contracts are a collection of code and data in state variables. These embody its functionality and state [11].

The contracts are deployed at a specific address on the Ethereum blockchain. Contracts obtain specific rules who grant permissions to who and what to read and write on the contract. There are for example four types of visibilities for functions and state variables: external, public, internal or private. They have to be specified and serve a similar function to those in other object-oriented languages, namely public and private variables.

It is important to understand that everything inside a contract is visible to anyone observing the blockchain. A private variable only prevents others from accessing and modifying the information. However, it is still visible to anyone outside of the blockchain.

2.2.3 Transactions

It is also crucial to understand that Ethereum has a set cost per computational step that is executed on the contract. This implies deploying a contract on the Ethereum blockchain or running a function on a contract, both of which are fundamentally transactions. When a contract

is executed, all instructions are executed on every node of the network which ultimately has a cost. The cost is measured in units of gas and paid in Ether, Ethereum's own cryptocurrency [12].

Ethereum transactions are all signed using the ECDSA algorithm, specifically Bitcoin's secp256k1 curve. The encoding of the set of rules that comes with a mechanism and the cryptographic properties keeps it adequately secure without involving a third party. Essentially, it makes it possible to build and execute an application with rules that are immutable once it has been deployed. The contracts run transparently and cannot be manipulated, which provides correctness [13], [14].

2.2.4 Mining

The primary functions behind Ethereum's mining process are the same as Bitcoin. However, Ethereum utilizes the Ethash mining algorithm, which uses the Keccak hash function, rather than the SHA-256 algorithm found in Bitcoin's mining process. Additionally, not all miners will accept every transaction as some might have rules for only accepting transactions with a certain minimum gas price. If a transaction has a set gas price lower than that limit, those miners might just ignore your transaction. The higher the gas price, the more likely the transaction is included in the next block. A block is attached to the Ethereum blockchain every 14-16 seconds. However, a miner can, for instance, configure its node to help the network by mining only low gas transactions [15].

2.2.5 Architecture

Unlike Bitcoin, Ethereum has the property that every block contains something called the state root. It is a special kind of data structure called Merkle tree that stores the entire state of the system. It allows a node, given only the most recent block, to synchronize with the blockchain quickly, without processing any historical transactions. Like the basic Merkle tree, any piece of data inside the tree to be securely authenticated against a root hash [2]. It also has the property that data can be added, removed or modified in the tree quickly, without making changes to the entire structure. The tree is used in Ethereum to store transactions, receipts, accounts and the storage of each account.

3 Social Aspects

This section introduces the social theory of internet-voting and describes how the Swedish electoral system works today. Furthermore, a study of the environmental effects of the current Swedish electoral system is given.

3.1 Swedish Electoral System

In the last Swedish election 2018, 87,18% of the qualified voters voted [16]. According to val.se 3,617,199 of the voters voted on election day [17], which is roughly 84 votes per second distributed over a 12 hours period when voters were allowed to vote.

Swedish Vallagen [18] is the legislation related to the voting process. It includes laws regarding voting ballots, stations and polling booths. The main idea behind the regulation is to ensure correctness, integrity and privacy throughout the process. This suits the purpose of the suggested voting system presented in this report, with the main difference being the ballot system. The ballot system has also been subject for discussion by many parts involved in the voting process [19].

For Sweden, Vallagen states that a citizen is given a vote card and that this card is personal and must be brought to a voting station in order to be able to vote. Vallagen also states that there has to be vote-receivers in the voting station which validates the eligibility of the votes. Vote-receivers are crucial to establish trust and order in the voting station and the voting cards makes it possible to validate the voters as eligible with two-step authentication.

In the Swedish general election the votes are counted twice. First by the vote-receivers in groups of 4-5 and then by the vote counters. The vote counting starts immediately after the voting booths closes. These workers are recruited by the electoral board in each municipality. The vote-receivers job is also to place the voters vote into the ballot box. The common factor between these jobs is that both have to complete a compulsory education where accuracy is of great importance [20].

3.2 Internet-Voting Today

Although it is not a decentralized voting system, internet voting already exists. In Estonia, i-Voting has been available since 2005 and in the last election, 3 March 2019, 43.8% voted over the internet [21]. The government says that its simplicity helps save the country a total of 11,000 working days each election year [22].

The website e-Estonia [23] explains how their system works, "During a designated pre-voting period, the voter logs onto the system using an ID-card or Mobile-ID, and casts a ballot. The voter's identity is removed from the ballot before it reaches the National Electoral Commission for counting, thereby ensuring anonymity".

In 2014, Alex Halderman, a cybersecurity expert and his team visited Estonia to investigate if the U.S should implement a similar voting system [24]. Amongst a series of security problems, Halderman said "Estonia's Internet voting system blindly trusts the election servers and the

voters' computers" [25].
With a decentralized system, trust in the election servers or voters' computers would not be needed. Only trust in the blockchain.

In corrupt countries where rigging is common [26], a decentralized voting system would solve the trust issue. The voters would have to trust the blockchain instead of the government. However, in rural and poor regions of Mexico [27] and Argentina [28] amongst other countries, vote-buying is widespread and common. With i-Voting and mobile-voting, votes would be easier to monitor. Susan Stokes, an American political scientist writes "The more accurately the machine can monitor voters, the greater the potential for vote buying. This accuracy is a function of the technology for monitoring voters' actions and of the machine's organizational structure" [28].

3.3 Environmental Aspects

The environmental effects of a customary paper-based election are the paper, printing and transporting. Mainly voting-cards, ballots and the envelopes. While the environmental effects of a blockchain-based election are the CO_2 emission from the electricity required. In order to see if a blockchain-based election would be sustainable, some calculations will need to be done.

The total energy consumption for the Ethereum blockchain requires (as of writing this, 2019-03-14) roughly the same amount of energy consumed by Ethiopia or Costa Rica, with a current estimated annual electricity consumption of 8.88 TWh; where one transaction costs the same amount of energy as ˜1.49 U.S. households, consuming 44 kWh [29].

The amount of electricity consumed worldwide the year 2014 was 20 591 TWh [30]. The same year 9.8±0.5 GtC (billion metric tons of carbon) was emitted worldwide [31], where 49.04% of which came from electricity and heat production [32]. The reason for using data from 2014 here is because this is the latest year (as writing this) when the percentage of the CO_2 emission from electricity and heat is presented, by the *International Energy Agency*, on *The World Bank* website [32].

The emission percentage from electricity and heat cannot be separated here so the result will be higher than the actual value. But the assumption is that most of that emission is from electricity production. However, if an assumption is made on the percentage difference between electricity and heat, the end result can simply be divided by the proportion.

In order to calculate the total CO_2 cost for a transaction on the Ethereum blockchain we must first calculate the carbon emission from electricity. Then the carbon emission per each kWh, since the energy consumption for a transaction is given.

Using the collected numbers above: GtC (billion metric tons of carbon) from electricity & heat per TWh:

$$\frac{(9.8 \pm 0.5) \cdot 0.4904}{20\ 591} = 233\ 399 \cdot 10^{-9} \pm \frac{0.2452}{20\ 591}$$

Same result transformed to kgC (kilograms of carbon) from electricity & heat per kWh:

$$233\ 399 \cdot 10^{-6} \pm \frac{0.2452}{20\ 591} \cdot 10^{3}$$

Then, since one transaction had the energy cost of 44 kWh, we can calculate the carbon cost for an arbitrary amount of transactions:

$$C(n) = 44n(233\ 399 \cdot 10^{-6} \pm \frac{0.2452}{20\ 591} \cdot 10^{3})$$

where n =Amount of transactions, $C(n)$ = kg Carbon released (multiply by 3.664 for kg CO_2).

4 Method

This section goes on to explain the design of the decentralized voting system. The nature of this project has been investigatory and has followed an iterative process of week-long iterations. The ultimate goal has been to satisfy the requirements that were stated in the problem definition, namely privacy, integrity, and correctness.

The iterative process allowed the group to reflect on the current state of the implementation every week in relation to the specified requirements. It required relatively short iterations due to the broad and subjective problem requirements. Any straightforward technical and theoretical goals are hard to derive without proper knowledge of any available tools, frameworks, and theories that impact the solution, which was obtained throughout the iterative process in during the project.

The project has covered research and implementation in parallel, with the two complementing each other in a circular manner. At the weekly meeting, where the past and coming week are discussed in the group, the requirements were addressed and compared to the state of implementation which lead to new insights on what theory or solution was needed to move forward. When a new solution could be derived from research, the implementation would adapt.

4.1 System Design

The design of the system meets the requirements stated in section 1.3. This section describes the approach to each defined problem and the methodology of the entire system. The design presupposes that a trusted third party (TTP) will handle the election, ideally *Valmyndigheten*.

The abstract system design and its core functionality are illustrating a hypothetical scenario involving two citizens that for illustrative purposes will be called Alice and Bob. Alice is trying to vote with the system, and Bob wants to see the result of the election.

4.1.1 System components of DeVote

As an introduction for what is to come, the project group decided to name the system DeVote to introduce some character to the project. Figure 2 illustrates the methodology of DeVote and how the entire election process proceeds. In order to integrate the system with the user, the system was built in a web application to simulate the system components. It acts as an API between the user and the system.

In short, the system design consists of two abstract components, namely a voter ID generator that will provide eligible voters an anonymous ID, and the smart contract. The contract will provide the back-end functionality to meet the requirements for the system and ultimately allow citizens with an ID to place a vote. The anonymous voter ID generator and the voting booth illustrated in Figure 2 is referred to as a TTP, which is ultimately responsible for the distribution of voter IDs to the citizens and administrating voting booths.

The smart contract acts as a set of rules to accommodate integrity and correctness by providing accuracy of data throughout the election and not allowing unauthorized modification of the electoral results, illustrated in Figure 3.

To complement the step-by-step illustration and system component explanation the next subsubsection 4.1.2 and 4.1.3 will present how DeVote meets the requirements stated in subsection 1.3.

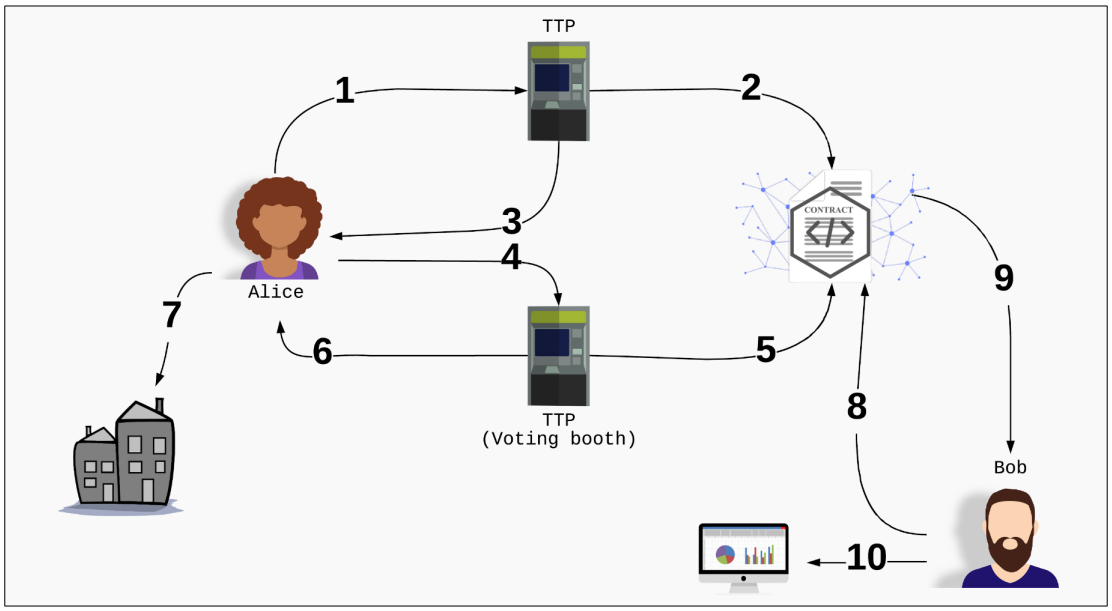


Figure 2: 1. Alice asks for a voter ID. **2.** A TTP generates an anonymous ID and saves it on the smart contract. **3.** Alice receives the ID. **4.** Alice uses the provided ID to place a vote. **5.** The TTP checks if the given ID is eligible to vote and (if so) stores the vote on the contract. **6.** Alice gets a confirmation that her vote was placed. **7.** Alice can go home and check by herself that her vote was placed. **8.** Bob asks to see all the votes. **9.** The smart contract provides Bob with the votes. **10.** Bob can now make sure the result is correct.

4.1.2 Privacy

A citizen has to be eligible to vote in order to place a vote during the election. A TTP will assure they have the right to vote and grant them permission to do so. In this process, DeVote has to assure privacy for individual voters by making votes untraceable. The TTP's purpose is to protect citizens identities by providing voters with an ID that is anonymous and not related to their credentials. The voter should be able to use this ID to submit a vote anonymously.

4.1.3 Integrity and correctness

DeVote has to assure that only a TTP has the permission to manipulate the smart contract. Ultimately, all IDs are generated by the same TTP in order to prevent forgery. Together with the smart contract that acts as a set of rules, accommodate correctness.

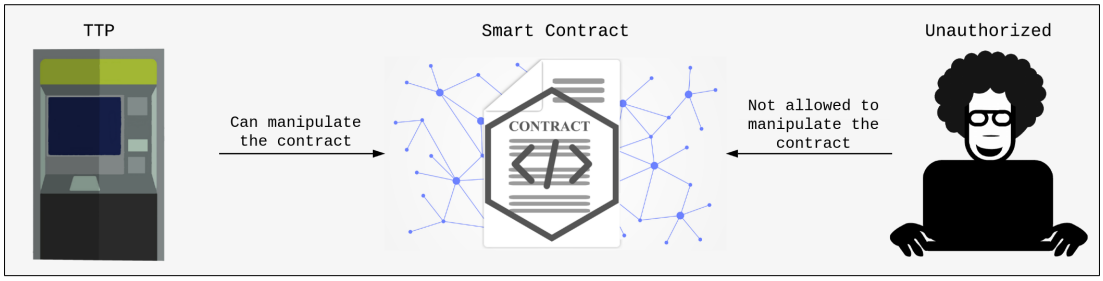


Figure 3: Only a TTP is able to manipulate the smart contract in order to allow eligible voters to place their votes on the contract. This ultimately prevents unauthorized entities to forge the result and manipulate the election.

4.2 Tools

Throughout the process of investigating the possibility of implementing a decentralized voting system a number of tools were used. These are introduced in this subsection.

In order to synchronize all six members of the project and develop code simultaneously, specific tools were used to ease the collaboration and standardize the code output. To be able to parallelize work Github was used. Together with Github, a CI system performed tests on all the code that was developed before it was approved to be added to the main application. To assure code standards, a linter was added to the project together with a webhook, running the linter on each contribution to the main application in order to prevent low quality code.

The project was developed as two separate systems; one for the smart contracts, and the other for the Web application. However, both are saved together in one main project because the Web application depends on the compiled smart contracts. To handle all the dependencies and frameworks used for both systems, *npm* is used. It helps to download all the required files in order to run the entire system. Build- and run-scripts were also made in order to ease the development process. For a full description on how the code was structured and executed read the detailed instructions in the repository on Github (see Appendix A).

4.3 Gathering of opinions

Since a voting process is for the citizens, what is considered as an ideal is a highly subjective matter. To be able to decide on how citizens engage in our voting system, a survey was conducted for gathering opinions.

The survey was built out of three sections gathering opinions on three different subjects. The respondents view of Sweden's current voting system, the respondent's opinion on factors in an ideal voting system and the respondent's opinion on an electronic voting system. The questionnaire can be found in Appendix B.

The Survey was mainly advertised through social media and to the social networks of the authors of this report. The survey was conducted through a Google Forms and gathered the answers anonymously through the internet.

4.4 Environmental Research

In order to compare the difference of the environmental impact between a paper-based and a blockchain-based election on a national level, the focus simply stayed on the CO_2 emission. The blockchain data collected, was for the Ethereum blockchain.

The company, *Lessebo paper*, that was in charge of printing the voting ballots for the Swedish county- and parliament election was contacted via email, regarding CO_2 emission data related to the election.

Data for the total electricity consumption was collected regarding the selected blockchain. Then worldwide data regarding CO_2 emission, electricity consumption, and the percentage of emission that is from electricity production. This was used to create a formula for calculating the total carbon emission, in kilograms of carbon, for an arbitrary amount of transactions.

In order to calculate the possible CO_2 emission the Swedish parliament election would have had if it was held on a smart contract on Ethereum, numbers were collected on how many actually voted in the election.

In the end, a comparison was made between actual emission data of the paper-based election and the calculated possible emission for a blockchain-based election.

5 Implementation

The decentralized voting system described in section 4.1 is built in two parts. This section goes on to explain how the system was implemented. The first part is written in Solidity and consists of two smart contracts utilizing the Ethereum blockchain in order to store votes and make sure a voter is eligible to vote. The second part is a web application that communicates with the contracts and represents the actual voting process. Tests were performed to evaluate the functionality and observe how the system behaves.

5.1 Smart contracts

The purpose of the smart contracts is to act as a back-end for the system. They contain different variables, functions and providing rules for how the voting process works. Due to the limitations of Ethereum, such as private variables that are not truly private [33] and all transactions being public, solutions for keeping data inaccessible had to be made. Specialized tools were used to support the development of smart contracts (see section 5.4).

5.2 Web application

A web application was made to act as a GUI for the election process. This application was developed using React. The application simulates the original design illustrated in 4.1. It generates voter IDs, stores them, retrieves a public key to encrypt the vote on the client side using a cryptographic library. The application then stores encrypted votes over transactions made to the smart contract.

Due to constraints in Solidity, there is no easy way to fetch a whole map or array from the smart contract. This had to be done manually by getting each vote one by one from the contract and decrypt them in the process.

There are several third-party dependencies used for the Web application to make the application more user-friendly. A list of all dependencies can be found in the repository on Github, See Appendix A.

5.3 Prototype

The prototype consists of five parts, visualized in figure 4, each built to simulate how the election would take place in reality. The discussion regarding the protoytpe is discussed in section 7.3. The four parts of the Web application are presented with different GUIs. The entire sequential process of the prototype is illustrated in Figure 5 and Figure 6.

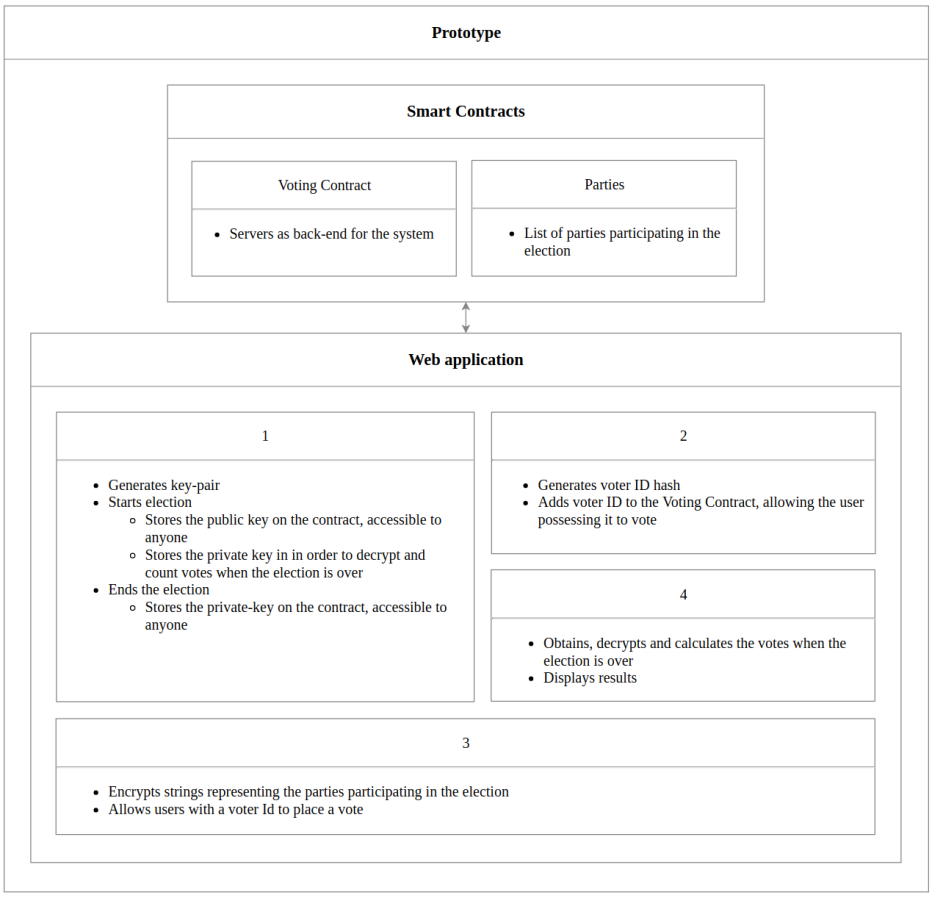


Figure 4: Diagram shows the different parts of the prototype and states their core functionalities.

5.3.1 Smart contracts

The prototype incorporates two smart contracts that are deployed before the application can start. The deployment is done manually, outside the application scope. One contract is implemented for the election process and acts as a back-end for the system. This contract is called *Voting*. It contains variables and functions that ensure the election proceeds as expected, according to a set of rules that accommodate the reports original purpose. The second contract is called *PoliticalParties* and stores the names of the different political parties that an eligible voter is able to vote for when the election is running. This contract is only readable and stores an array of strings.

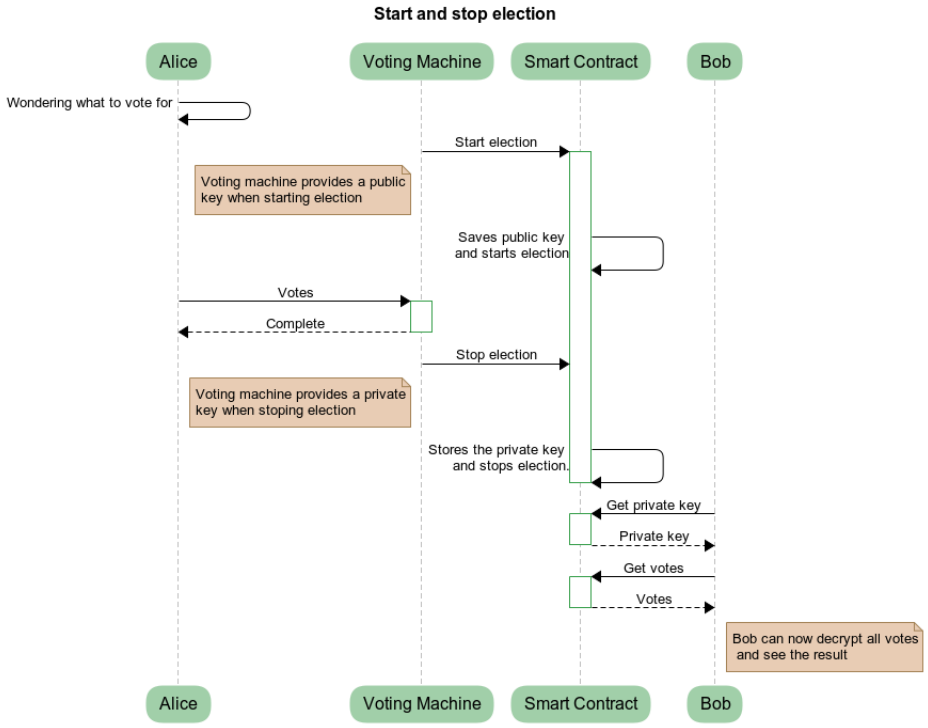


Figure 5: Sequence diagram that demonstrates how the prototype starts and ends an election. It visualizes the steps before Alice votes, what happens when an election ends and how Bob, as a consequence of ending the election, can access the results. The vertical lines represent a timeline that starts at the top of the diagram.

Starting an election requires two parameters, a public key and the maximum number of voters. The owner calls the *startElection* function in order to initialize the election. The *addVoter* and *vote* are the two main functions on the *Voting* contract. *addVoter* accepts the voter ID, a randomly generated hash, and adds it to a map, and an array on the contract. This function is key to ensuring that only eligible voters can place a vote. When a vote is placed the *vote* function encrypts and stores a string of the selected party in the map, mapping the voter ID to the encrypted party. The array serves a purpose for reading the result when the election is over, due to limited properties of Solidity.

5.3.2 Web application

To make sure only eligible voters are allowed to vote, a TTP that deploys the contracts and starts the election has to approve the voters. In the prototype, eligible voters, represented by a randomly generated hash, are added to the contract when the hash is generated. A voter can then exclusively, using the voter ID, select a candidate stored on the contract and place a vote. The voter can verify that their vote was stored on the contract by trying to vote again, or by requiring the votes after the election is over, as Bob does in the sequence diagram (Figure 5).

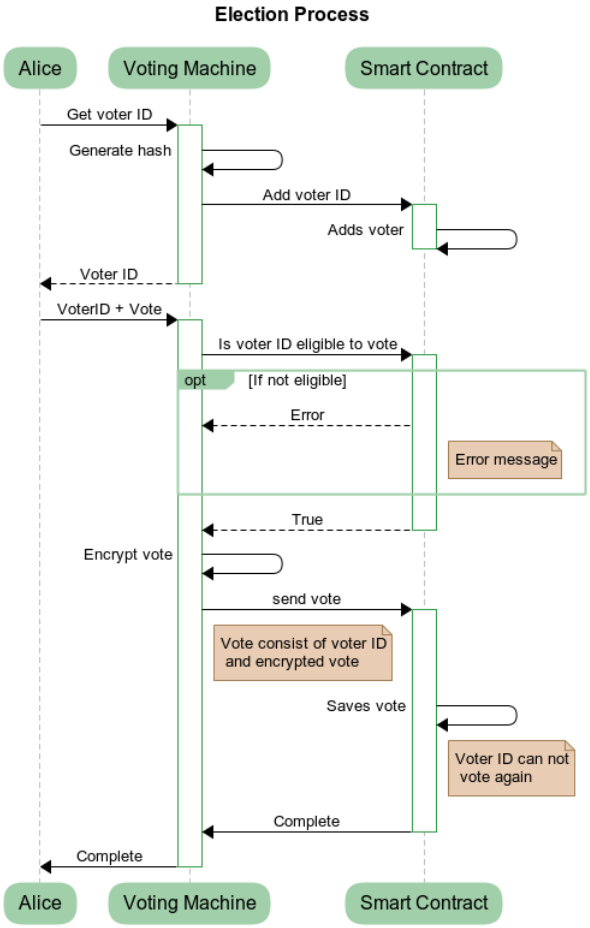


Figure 6: Sequence diagram that demonstrates what happens in the back-end after an election has started and Alice, who is eligible to vote, decides to place a vote. The vertical lines represent a timeline, starting at the top of the diagram. The *Voting Machine* represents the digital wallet that deploys the contract, starts and ends the election and is granted permission to write to the contract; thereby the same digital wallet that provides Alice a voter ID and later encrypts and stores Alice' vote on the contract.

The election is over when the owner ends it. The system was implemented this way due to the limitations of Ganache. Thereafter, the result is presented in a different GUI, showing a graph of the result. When the election has ended the front-end calculates and displays the result by using the published private key on the smart contract.

It is important to understand that whoever deploys the contracts owns a wallet which, by the rules of the *Voting* contract, is the only one who possesses the ability and permission to add voters, vote, start the election and end the election. This means that the same wallet is applied across part 1-3 of the front-end application, as only the owner is granted those functionalities and permissions on the contract.

5.3.3 Encryption

Votes were encrypted to avoid possible biased voting due to transparency on the blockchain. Votes were encrypted with ECDSA encryption, mentioned in 5.4.3, that is originally implemented by the Ethereum blockchain.

In order to perform the encryption, a key-pair was generated. The public key is stored on the contract, accessible to everyone when the elections start. The private key is stored ideally offline and used to end the election, which otherwise keeps running. In the prototype, the private key is stored as a variable in front-end application to serve its purpose for the simulation.

To end the election, the owner has to send the private key to the contract where it is stored and accessible to anyone, or else the election will keep on running. Anyone can thereafter decrypt the votes and count the result.

5.4 Tools

In order to perform tests of Solidity code, Truffle and Ganache were used. Truffle includes a test framework called Mocha which makes it possible to write JavaScript tests. Besides the Solidity tests, there were also tests designed for the web application, mainly to ensure the encryption was done properly. To collect test data, the Ganache CLI version was used and data was later parsed.

5.4.1 Truffle and Ganache

Truffle is a framework that was used throughout the process of developing the prototype. It was helpful by having the ability to act as a package manager, but also came with built-in functionality to compile Solidity code and deploy it to a blockchain. Truffle offered three different commands: *truffle compile*, *truffle migrate* and *truffle test* which were used to compile the contracts, deploy them to a blockchain on the network and perform tests on them.

Ganache was used to simulate an Ethereum blockchain on a local machine. It came in two different representations, GUI and CLI.

5.4.2 Drizzle

Drizzle is a front-end framework for Ethereum that was used to create a GUI for the prototype. Its core functionality is that it connects the smart contract and transaction data to a Redux store by providing synchronization between these; it keeps data in sync with the blockchain. It works well with React, therefore the choice of how to build the front-end came naturally by using React. It also had a set of components that were ready to use in order to make it easy to connect an already existing app to React with Drizzle [34].

5.4.3 Cryptography

In order to avoid visible results while the election is running, votes were encypted using an open source library called *eccrypto*, which in turn is wrapped in an open source Javascript library eth-crypto, specifically built for Ethereum transactions (see Git Repository in Appendix A). It utilizes ECDSA encryption algorithm that is originally implemented by the Ethereum blockchain [35].

6 Results

This section will go through the results that were obtained during and after the implementation.

DeVote, the decentralized voting system that was originally proposed in this thesis applies an updated, digitalized voting process for citizens that participate in the election. Mainly due to the change from ballot voting to an electronic system, but also in order to develop the electoral system further towards our goals to provide: security, integrity, and correctness.

The voting process is similar to traditional ballot voting. Every eligible voter gets a voting card in the mailbox and brings it to their respective voting station. In the voting station, the voter is authorized and checked off a list by a vote-receiver. Another vote-receiver then generates a voter ID for the voter. This ID is created as a physical note which is only known and accessible for the voter and not for the vote-receiver, or anyone else for that matter. This physical note is then taken to the voting machine where the voter decides and marks what party to vote for. After casting a vote, the voter will get a slip which gives instructions on how to access the result and their own vote on the blockchain after the result is posted.

Below are the results obtained from the implementation through the various test, and the conducted survey and environmental calculations.

6.1 Tests

Tests were performed to observe if integrity, correctness, and privacy were achieved. Additionally, two tests collected data to inspect gas cost was used for different kinds of transactions.

6.1.1 Smart contract

A number of tests were performed on the smart contracts. These were built to evaluate if integrity, as described in 1.3, was preserved throughout the prototype. It is important to keep in mind that all of the tests were performed in the prototype environment, meaning using Ganache to simulate a blockchain. The source code is available on a public repository Github.

All tests passed the criteria described in Table 1 except for one, namely *Carry out a large number of votes*. It was an experiment to simulate votes and initially perform an arbitrarily low number of transactions to the smart contract. The results get unstable when the number increases; some trials failed while others passed. When the number reached over 3000 transactions there were no longer any results passed to the smart contract. This test was made in order to observe how the smart contract and Ethereum were acting during a high load. It was an attempt to simulate if it would be possible to perform all votes placed during a real election. However, this result is uncertain; most likely as a result of the test environment and how Ganache deals with transactions and the creation of new blocks. A suggestion for future work is proposed in 8.1.

Tests	
Name	**Description**
Add voter	Ensures a registered voter ID on the smart contract is able to place a vote
Not add voter	Checks if anyone besides the owner of the smart contract is able to add a voter ID
Not able to vote	Control if a non-registered voter ID is able to place a vote
Vote	Tests if an eligible voter's vote is registered
Not able to vote twice	Makes sure a voter ID can only place a vote once
Get number of voters	Retrieves the number of registered voter IDs
Carry out a large number of votes	Executes a test to inspect how many transactions it is possible to perform during a short time period. These transactions are made sequentially and will not wait for a return value.

Table 1: Lists and describes various tests performed on the smart contract

6.1.2 Transactions

During the testing process, two tests were performed to collect data in order to investigate the amount of gas usage in different sized transactions.

The first test observed how the gas usage increased when the length of the string, representing the party, was made longer. It was done by iteratively performing 250 transactions and concatenating one additional character, to the current string starting from one initial character, for each transaction. This test was performed to observe how different kinds of encryption and string compression could affect the price for a vote. Since the hashes generated for voters used in the prototype is of constant length, only the vote string was changed. The result is shown in Figure 7.

The result indicates what is essentially stated in the Solidity documentation, namely that one character represents one byte. Their type can vary depending on the length of the string they create and are implicitly convertible up to 32-byte type.

As observed in the graph shown in Figure 7, the gas cost just after 32 characters increases more than for each increment thereafter, which is more or less constant. This is most likely due to the type variation just mentioned. Data is stored in a data structure after 32 characters. When the number of characters reaches 64 another 32-byte type is stored in the data structure, containing the next 32 characters. This goes on for each additional 32 characters. There are in total 7 increments, representing 7 types, each storing a set of 32 characters.

The second test is performing an identical vote for each transaction. Instead of encrypting the

party name each vote, the same string is used. This string has therefore constant length each iteration. Together with the voter ID hash, also of constant length, votes were placed to observe if the gas-usage was changed when the number of votes saved in the contract increases. See results in Figure 8.

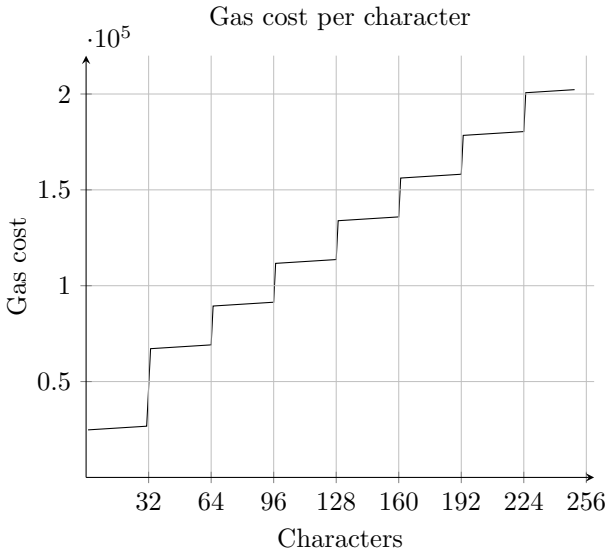


Figure 7: Showing that gas cost per transaction increases when the number of characters in the transaction increases. Note that each transaction represents a vote, and a vote consists of a voter ID and a string that represents a party. The voter ID was kept constant and the party-string was initially one character long and was incremented each transaction to a total length of 250 characters. Therefore, the x-axes (Characters) also shows how many transactions were made.

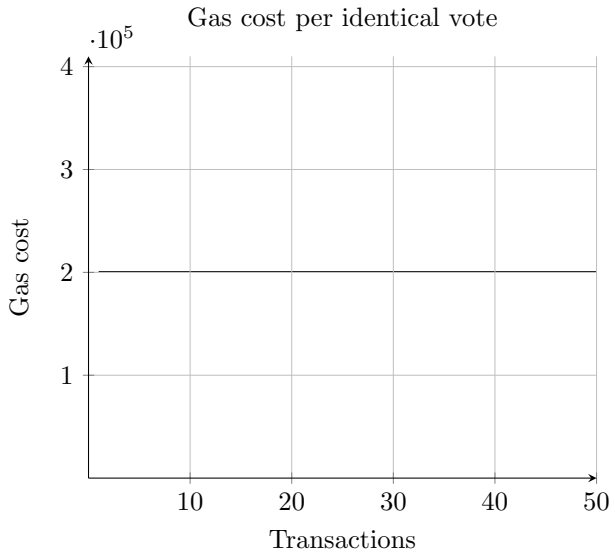


Figure 8: Showing that gas cost per transaction for a total of 50 identical votes is constant. Each transaction is equivalent to one vote.

6.2 Survey

The survey to gather opinions about a decentralized electoral system, among other questions, was conducted. The results gathered from the questions can be found in Appendix C. The survey gathered 183 responses in total. 68% of the participant were Swedish citizens between the age of 20 and 26. Others were Swedish citizens older than 26.

A 95% proportional binomial confidence interval was calculated for each question, with the following results:

- (1) 60% to 74% Yes
- (2) 70% to 83% Yes
- (3) 49% to 63% Yes
- (4) 68% to 80% Yes

From the survey, opinions on other topics were also gathered; some more relevant to the study than others. However, the results that are relevant for this report state that 113 of 124 respondents answered approximately that possible security issues might cause fear or resistance towards electronic systems. Out of the eleven remaining, 9 were either worrying about the system being dependent on electricity or had no fear.

6.3 Environmental impact data

Lessebo Paper, who printed the voting ballots for the county- and parliament election, was contacted regarding their CO_2 emissions. The following answer is from a reply email written by *Anders G Karlsson* (translated from Swedish):

- For the county- and parliament elections 2018 Lessebo produced 650 metric tons of paper.
- The paper is produced by bleached or non-bleached pulp. In many qualities (e.g. newsprint) recycled fiber is used. All our pulp is bleached and comes from *Södras* pulp mills in Mönsterås and Mörrum, which is close to Lessebo.
  Some type of fillers, often chalk ($CaCO_3$), is mixed in to give better printing properties, etc. The mix of fillers stays between 5-25% depending on the usage. The paper contains between 5-7% water in the finished product.
- The ballot papers pulp mix is 10% long fiber and 90% short fiber.
- The total CO_2 emission for the product is 47.9 kg/ton paper. Note: That number includes pulp producers emissions. Total 31 ton CO_2 for what we did.

The question is how many transactions a Swedish election would have. We first have to consider that all of the generated hashes for each vote has to be published first, then used when voting. So each vote consists of two transactions. Then the election has to be published, started and ended. So the number of transactions would be:

$$T(n) = 2n + 3 \text{ where } n = \text{amount of votes.}$$

The number of people who could vote in the Swedish election 2018 was 7 495 936 [36], however only 6 535 271 actually voted [37]. This means that the total amount of transactions needed for the Swedish election 2018 would have been:

$$T(6\ 535\ 271) = 13\ 070\ 545$$

Hence the mean carbon emission would be:

$$C(13\ 070\ 545) = 134\ 228\ 693.82802 \text{ kgC } [\pm 6\ 848\ 404.443494731 \text{ kgC}]$$
(See 3.3 for formula)

We can multiply the result by 3.664 to transform it into kg in CO_2 (see 3.3). So the total CO_2 emission would be 491 813 934.18586528 $kgCO_2$.

As mentioned previously, the county- and parliament election, in 2018, produced 650 metric tons of paper with the emission of 47.9 $kgCO_2$ per ton paper. Which means that the total emission was:

$$650 \cdot 47.9 = 31\ 135 \text{ kgCO}_2$$

Let us assume that half of that was for the parliament election alone, i.e. 15 567.5 $kgCO_2$. This means that if the Swedish election would have been on a smart-contract on the Ethereum

blockchain, the total CO_2 emission difference would have been:

$$\frac{491\ 813\ 934.18586528}{15\ 567.5} = 31\ 593.366363838$$

This means that the Ethereum-based election would have had roughly 31 593 times more CO_2 emission than what the paper-based election actually had, i.e. not ideal.

7 Discussion

This section goes on to discuss the trade-offs and benefits of DeVote; its design, the implementation, results and its underlying technology, namely blockchain and its limiting properties in relation to the system itself.

7.1 Blockchain and Decentralization

The discussion begins with one of the main challenges which are the limitations of the blockchain design.

When blockchain was first introduced, its core purpose was to act as a cryptocurrency, and the technology was used to perform transactions with the currency. When bitcoin gained traction in terms of popularity people started to realize the potential that a decentralized system like blockchain could have. Since then a large number of different blockchains have emerged on the market, most of which still only have the ability of transactions between currency.

Presented here are the trade-offs related to DeVote, and the different dilemmas that these create. The trade-offs mainly revolve around an already known issue, coined by Ethereum's creator Vitalik Buterin, named "The Scalability Trilemma" [38], [39]. The Trilemma is defined as the trade-off where one can only have two out of the three properties, namely decentralization, scalability or security. It is important to point out that these properties are not binary. For example, if you increase the degree of decentralization, you ultimately pay in scalability or security. In the case of the prototype, one of the core properties of blockchain, transparency, had to be affected to a certain degree in order to hide the result of the election until the end. As a result, one could argue that decentralization was cutback due to a TTP that had to be involved in the process to ensure only citizens eligible to vote could participate in the election.

The perhaps most obvious trade-off is the issue with scalability of the transaction throughput in public blockchains, further discussed in 7.1.2. The Ethereum documentation describes the problem as the following:

"Currently, in all blockchain protocols each node stores the entire state (account balances, contract code, and storage, etc.) and processes all transactions. This provides a large amount of security, but greatly limits scalability: a blockchain cannot process more transactions than a single node can." [39].

7.1.1 Benefits and trade-offs of decentralization

As introduced previously in 2.1.5 the blockchain can be divided into private or public based on their properties. Each type of blockchain has certain advantages and disadvantages. The core advantages of private blockchains are that rules of a blockchain can be changed, transactions can be reverted, balances modified, etc. This also implies that faults can be fixed. Transactions are cheaper since they only need to be verified by a few trusted nodes with high computational power. Also, since read-permissions may be restricted, this implies that the level of privacy is high. Finally, any risk of a 51% attack does not apply as the miners are known.

Nonetheless, the core idea behind a public blockchain is to have a public and decentralized

system which in turn creates transparency. There is a fine line between centralization and decentralization in regards to what a private chain can offer. One could argue that if the miners are limited to a certain group of people this would, in turn, create a centralized network. However, downsides with public blockchains are the possibility to trace individual transactions, and thereby individual users, by collecting and analyzing metadata [3].

Public blockchains like Ethereum provide philosophical core values such as freedom, neutrality, and openness, all of which fit into the core values of democracy. It accommodates censorship resistance, protecting the users from the developers and authority. A public blockchain has no restrictions, meaning anyone can send transactions to - or mine the blockchain [40]. This is of great value to the vast and increasing number of users that public blockchains have today [41].

The reason to choose a private chain instead would be to deal with issues such as to eliminate the chance of a 51% attack or to increase the transaction throughput.

7.1.2 Limitations of transaction throughput

Another limitation when it comes to decentralized blockchains is the time it needs to validate a new block of transactions. Ethereum architecture allows it to store and access data quickly due to the nature of its data structures, it takes on average 15 seconds to create a new block [42]. The reason lies within the consensus mechanism that ensures that miners cannot interfere with the security of the network and for example forge transactions.

Basically, block time is the time it takes for a network to find the nonce to generate a valid output and thus the time it takes for a new block to be added to the chain. The result of the cryptographic properties states that the longer the time it takes to add a block, the more secure it is. Hypothetically, if the block time would be very short, the possibility that someone could modify the blockchain would increase. This property creates problems when creating a voting system. The block time for Ethereum would not make it possible to have an election within a reasonable time limit.

During election-day in Sweden 2018, the throughput was approximately 80 votes per second. Perhaps if the election period was longer there might be a possibility for Ethereum to work. However, it might not be feasible for reasons outside the scope of this thesis. Therefore, if a new blockchain would be invented for the purpose of voting, assessing the block time would be a large part of the work. As of today, due to the cryptographic properties of Ethereum that provide immutability, there is no way to avoid the problem that security will be jeopardized if the block time would be decreased which in turn would jeopardize correctness and integrity.

To sum up, in regards to DeVote, it all comes down to the choice of either implementing a decentralized but slow solution which is provided by a public blockchain, or a scalable (meaning faster throughput) but centralized solution which is ultimately provided by a private blockchain.

7.2 Properties of Ethereum

The main reason why Ethereum was used to develop the prototype for this thesis is due to its mobility and widespread use. It provides developers with smart contracts for setting up rules of a decentralized application. But what gave more value and benefit to the thesis was the tools

and documentation provided to make the prototype possible. In this section, we discuss how the Ethereum blockchain fits into a decentralized voting system.

As the world's second largest blockchain in terms of financial value [43], Ethereum provides anyone with a DAPP idea with a powerful platform. Ethereum offers a transparent and pedagogical interface for creating and releasing decentralized applications. Whilst being an open-source project, it does not hide any functionality and strives to be fully transparent. Ethereum aligns with the goals of DeVote because of its transparency as well as its technical aspects. Despite Ethereum's core functionality fitting the purposes of DeVote, there is a fundamental problem that arise from using it.

On the Ethereum blockchain, miners have the ability to alter the size of the blocks in terms of gas. This is to cope with the shifting transaction load while keeping the speed of mining new blocks. While writing this, according to ethstats.io, the average block size 8 003 799 gas. Looking at the results the transaction cost of one vote is equal to 200 000 gas. In a utopia where the Etheruem blockchain only consisted of transactions from DeVote, that would mean that $200\,000 \cdot 10\,000\,000$, the total gas cost of all votes, divided in blocks of 8 million gas with the aforementioned average block mining speed of 15 seconds, would need more than 43 days to process the whole election.

Nonetheless, Ethereum is transitioning to an updated version; utilizing a different consensus mechanism that allegedly will introduce beneficial properties to the blockchain's life-cycle. Buterin states that Ethereum's transactions per second can get a major lift in the coming years. In an interview with Abra CEO Bill Barhydt, Buterin said:

"There are two major kinds of strategies that we're working on for scalability. One is layer-one scaling and the other is layer-two scaling... And our solution to this, called sharding, basically means that you split up the different transactions to randomly selected, different groups of computers... And that can increase scalability by maybe a factor of 1,000 or so, but then potentially even more, much later down the road." [44].

7.3 Prototype

After discussing various limitations and aspects of blockchain and Ethereum, this subsection goes on to discuss the main issues and benefits related to the prototype. The central theme comes down to that the prototype depends on a centralized TTP in order to handle a set of actions for the election to work. These are holding the private key for the result decryption, therefore also starting and ending the election as well as authorizing the citizens as eligible voters. The last is not a constraint in the actual prototype but will be further discussed in 7.4.

7.3.1 Hiding information from miners

It is not possible to store private data on a smart contract. A smart contract is stored in memory on mining computers and all information on the contract is therefore accessible if one were to decode those bytes where the variables are stored. Therefore storing the result in these fields as clear text would expose the result for everyone that wants to mine on the chain and have the technical knowledge to obtain the data, which would jeopardize the correctness of the result

since voters can get biased to vote tactically. This would imply voting for a specific party which is near the parliament entrance limit, 4 percent in Sweden, thus obtaining more power.

The solution for the transparent election result was to run the parties through the aforementioned hashing algorithm, which can only be unlocked by a private key. This creates a centralized process in handling the keys but keeps the result from being spread from the miners. However, this harvests a need for a TTP which, in order to provide the function on encryption, has to ultimately deploy the smart contracts and keep the private key secret until the election has ended; but the TTP does not have to be trusted regarding the result.

7.3.2 Starting and ending the election

A decentralized solution for ending an election would be to have a set time for when the election would end automatically instead of having to end it manually. The problem with ending it automatically lies in that it limits testing on the Ganache side. Time on Ethereum exists on a time-stamp on each block that indicates when the block was mined. However, this is only an approximation that depends on the precision of the clock of the machine it was mined on. Consequently, another issue is that miners can manipulate this time-stamp, which makes it a not secure measure.

The time is calculated in seconds. While there is no problem to set how many seconds it would take until a certain time to stop the election, the block times in Ethereum makes it difficult to perform exact tests on Ganache. Furthermore, the Solidity documentation does not recommend relying on exact time-stamps [33].

The insecurities regarding tests that rely on time is another reason the prototype expects that a TTP deploys the contract (which would ideally be Valmyndigheten) and would thus be the owner.

7.3.3 Storing data and practical limitations

As observed in the data that was obtained, shown in Figure 7, there is essentially an increasing gas cost for larger transactions. It was not possible to compress the encrypted parties in order to decrease the number of characters due to the the encryption that was used. It generated long strings. And the way characters are encoded in Solidity also prevented additional compression of the data.

Moreover, for the visual representation of the prototype, some simplifications had to be made in order to make the prototype simulate a real election in an easy and user-friendly way. With DeVote, every polling station would use the same wallet where voters with an already generated voter ID, registered on the voting contract, placed their encrypted votes. However, in the prototype, all of this is built in into the same application, separated only by different views.

The back-end system incorporated into the smart contracts also comes with limitations due to Solidity. The only data structures that are available in Solidity are specific maps, acting as hash tables with key-value pairs, and arrays. This affected how data was stored on and obtained from the contracts. There are more optimal ways of doing this by using other data structures. However, those were not provided given the tools that were used to build the prototype.

7.4 DeVote in the Swedish elections

The ideal voting process is a major subject of discussion. Therefore, the capabilities and possible dangers of implementing an electronic voting system will be discussed, and further, how DeVote manages to cope with these.

7.4.1 Assuring privacy in the voting system

One of the important points of the voting system was to assure the privacy of the voters. However, privacy is subjective and is hard to define for every individual citizen. The goal with DeVote was meant to make it impossible to trace back a vote to the person actually casting the vote. This will, in turn, create problems for another fundamental point of interest, to assure that only eligible voters vote and that their right to vote is only used once.

These two are in many ways conflicting, and in the system used in Sweden today, referred to as ballot voting, it is unclear whether they are solved to the extent we are aiming for. Voting ballots are traceable back to their respective voter, but the ballots indicating the vote are sealed inside an envelope. Ballot voting is not foolproof but the outcome is satisfactory since there is simply no time for the vote receivers to open all of the envelopes and look. In DeVote, the votes are not connected to a person, but only a vote-receiver can give you access to vote. Once again, the trust is upon the receiver of the vote to follow the law and only hand out as many voter IDs as there are voters. However, it is possible with DeVote to compare the lists of eligible voters checked off with the number of voters registered, calling for reelection if the amount differs.

7.4.2 Assuring correctness in the voting system

In ballot voting, the votes are recounted to ensure that the envelopes match the ticks in the voting attendance list and in extension also the correctness of the result. The voters depend on the vote receivers to count the votes correctly and do not change the ballots. DeVote is dependent on the public citizens to actually check that the result is correct and the amount of correctness is directly correlated to how many actually check the result since the result is not owned by anyone, but for the public eye to view. The voters can easily check that the code that they used to vote correlates to the actual vote that they cast after the result has been posted, which mimics the fact that you can gather your ballot vote from Valmyndigheten after an election.

7.4.3 Assuring the integrity of the voting system

One part of implementing an electronic voting system is its mobility. Anyone with an internet connection could technically reach the place where the votes are stored, which in the case of DeVote is the smart contract. This leaves voters with possibilities and concerns which are clear in the survey results. The possibility of voting from any place the voter pleases is intriguing and could lead to great things such as higher voting attendance and accessibility for those with mobility issues. DeVote considered this opportunity but did not implement it for the sake of correctness and privacy.

The main problem is the identification of the voter, which has to be verified remotely. This

could compromise the privacy of the voter since the identification would need to be stored somewhere and could, therefore, be accessed and connected to the vote posted. As said before, the ballot system already allows this but the physical nature of the ballots makes it logistically impossible to compromise the privacy of the voters, which is not the case with digital voting. Another problem with remote identification is that there is no identification system available today that ensures that the vote cast is actually the democratic opinion of the identified voter, i.e. voters could actually be forced, be paid or intoxicated while voting. Therefore DeVote uses voting stations and polling booths in the same way as ballot voting.

The mobility of an electronic voting system also opens up for the biggest concern presented by the answers from the survey, hacking and tampering with the election results. As discussed and argued in this report, the blockchain implementation is immutable when correctly executed, which means that the issue lies in the gathering of trust from the people. When developing DeVote it has been of great concern to pick the good things from the ballot system but to expand on the possibilities of being electronic.

7.4.4 Gathering trust for the voting system

Looking at the results from the survey handed out for this thesis, the trend is that the citizens want to see their votes being counted and have a central authority that counts the votes. DeVote, as well as ballot voting, makes this possible in different ways. For ballot voting, the closed envelope with your ballot is put in the ballot box and then counted by those appointed by Valmyndigheten. After leaving the election hall you have to trust that the vote-receivers will actually count your vote. DeVote offers the possibility to see your vote being used in the actual election, as well as making it possible for the central authority to check that the result is correct in terms of the number of votes.

Results from the survey also show ambivalence in the question regarding seeing the live results of votes being cast. Ballot voting does not offer this feature for the sake of correctness. For DeVote, it is possible to implement both the ability to see the votes as they are cast and not to. The solution implemented in the prototype is a hybrid, the results are in the smart contract however they hidden under compressed hash codes and therefore it is impossible to trace it back to the party before the key is provided. Valmyndigheten has to keep the key from start to end of the election. Nonetheless, this makes the citizens dependent on those who have access, not to bias themselves or others with their knowledge.

7.5 Social and ethical aspects

To navigate the report towards a conclusion and an ending, a final discussion regarding DeVote to certain social and ethical questions is raised.

From a social standpoint the question arises: does Sweden need a decentralized voting system? Some argue that voting systems using blockchains are still too vulnerable to be used today [45]. It was concluded that the social and ethical effects of the project needed to be considered. Therefore the survey was conducted, see Appendix B and C. Among the respondents of the survey, young adults 18-25 were over represented. Since this study was conducted on social networks closely related to the authors of this report, it can be assumed that a big portion of the respon-

dents are students. This can of course bias the result since students are used to taking on new concepts. However, since the questions regarding a perfect electoral system was designed not to be biased by the idea of an electronic voting system, the final interpretation of the result is that the results are usable for the purpose of this report. It can also be noted that younger people would use the proposed system for a longer period of time which strengthens the interpretation.

The result of the interpretation made is that DeVote tries to resemble ballot voting in many ways, since this system has trust from the citizens. DeVote also follows the specifications of a perfect electoral system that respondents supported the most.

Throughout the project the following questions arose:

- Will more people vote?
  - The thesis does not answer this. The group reasons that if DeVote improves the voting system (from the voter's point of view), more people will vote. It is not clear from the survey if DeVote would be more beneficial to the people than the current system. Since 74.3% want a central authority counting the votes, which we remove. But on the other hand, 67.2% would want to see their vote being counted, which DeVote supports.

- Should citizens trust the blockchain instead of the government?
  - The group believes that in terms of security you most definitely can trust the blockchain since it is transparent and immutable.

- Will a decentralized voting system be socially accepted?
  - Based on the projects survey and the fact that i-Voting already is socially accepted in Estonia for example, the group believes that the system could eventually be socially accepted. Given that the system is proven secure.

- Is DeVote better for the environment?
  - From the results, we can conclude that from an environmental point of view, a voting system using Ethereum's blockchain is much worse than a paper-based one. However, the group believes that using a private blockchain, instead of a public, will decrease the environmental impact considerably. The results may also be quite different for future solutions, depending on how fast energy production worldwide will reduce greenhouse gas emission, and if future blockchains will handle transactions more efficiently.

One of the problems with the system is that it is vulnerable to vote-buying. As Susan Stokes writes in an article "The more accurately the machine can monitor voters, the greater the potential for vote buying..." [28]. If every voter is given a personal voter ID and the ability to afterward confirm that the vote has been counted for the correct party. Then it would be easy for the vote buyers to monitor this.

8 Conclusion

To summarize the report we can conclude that creating a decentralized e-voting system using the Ethereum blockchain is at the moment not possible. This is mainly due to the fact that the decentralization indirectly makes the verification process of the blockchain too slow to process the number of votes which will be cast per second on the voting day. This confirms the known trilemma problem regarding the scalability of public blockchains. We can also conclude that voting on an Ethereum-based blockchain would be far worse than paper-based voting in terms of environmental impact.

Regarding implementing DeVote as an electoral system in Sweden, we can first conclude that the system in place today works well in relation to our purpose. However, as argued, DeVote offers more transparency than ballot voting and in combination with its immutability and implementation it provides correctness, integrity and privacy. The concept would be an improvement to existing systems. The issue of gaining the Swedish citizens' trust in DeVote, was only answered with logical arguments. DeVote is after all a proof of concept.

Furthermore, we can conclude that DeVote is not entirely decentralized and there exist problems to assess when it comes to removing the TTP from political elections. This is mainly due to privacy and correctness issues. The fact that a voting station is still needed in DeVote results in that the impact on the voting process in Sweden is relatively small, but the technology and implementation is a great tool for eliminating corruption.

8.1 Future work

For the future, it will be interesting to see how Ethereum's transition to Proof Of Stake could benefit a system such as DeVote. And if other blockchain implementations apart from Ethereum could handle the requirements for this particular system. It would surely be relevant, and a suggested next step could be to implement this specific voting system on a private blockchain.

The prototype also remains to be tested on the live Ethereum blockchain. There are also a number of dedicated test networks in Ethereum named testnets, which are supported by various clients. For future work, the prototype could be deployed on a testnet in order to observe how it functions on a live blockchain. An interesting test might be to carry out a large number of votes.

Nonetheless, it is crucial to understand that there are still challenges left to possibly be resolved concerning blockchain technology. It remains to see what the future holds for this technology.

8.2 Acknowledgments

We want to thank Christos Profentzas for his guidance and shared knowledge as well as Anders G Karlsson for answering our questions.

References

[1] G. H. Baker, "A vulnerability assessment methodology for critical infrastructure sites," in *DHS symposium: R and D partnerships in homeland security*, 2005.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," http://bitcoin.org/bitcoin.pdf, 2008.

[3] P. Reynolds and A. S. Irwin, "Tracking digital footprints: anonymity within the bitcoin system," *Journal of Money Laundering Control*, vol. 20, no. 2, pp. 172–189, 2017.

[4] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *Ieee Access*, vol. 4, pp. 2292–2303, 2016.

[5] Jake Frankenfield. Double-Spending. [Online; accessed 13-May-2019]. [Online]. Available: https://www.investopedia.com/terms/d/doublespending.asp

[6] E. community. Ethereum Mining. [Online; accessed 2-May-2019]. [Online]. Available: https://github.com/ethereum/wiki/wiki/Mining

[7] V. Buterin and V. Griffith, "Casper the friendly finality gadget," *arXiv preprint arXiv:1710.09437*, 2017.

[8] Ethereum community. Ethereum Introduction. [Online; accessed 7-May-2019]. [Online]. Available: https://github.com/ethereum/wiki/wiki/Ethereum-introduction

[9] N. Szabo, "The idea of smart contracts," *Nick Szabo's Papers and Concise Tutorials*, vol. 6, 1997.

[10] C. Dannen, *Smart Contracts and Tokens*. Berkeley, CA: Apress, 2017, pp. 89–110. [Online]. Available: https://doi.org/10.1007/978-1-4842-2535-6_5

[11] ——, *Solidity Programming*. Berkeley, CA: Apress, 2017, pp. 69–88. [Online]. Available: https://doi.org/10.1007/978-1-4842-2535-6_4

[12] Ethereum community. What is gas. [Online; accessed 9-May-2019]. [Online]. Available: http://www.ethdocs.org/en/latest/contracts-and-transactions/account-types-gas-and-transactions.html#what-is-gas

[13] A. K. Koç, U. C. Çabuk, E. Yavuz, and G. Dalkilic, "Towards secure e-voting using ethereum blockchain," in *Towards Secure E-Voting Using Ethereum Blockchain*. IEEE, 2018.

[14] E. Foundation, "Information," 2019. [Online]. Available: https://www.ethereum.org/

[15] C. Dannen, *Mining Ether*. Berkeley, CA: Apress, 2017, pp. 111–137. [Online]. Available: https://doi.org/10.1007/978-1-4842-2535-6_6

[16] "Röster - val 2018," https://data.val.se/val/val2018/slutresultat/R/rike/index.html, 2018, [Online; accessed 05-Feb-2019].

[17] Valmyndigheten. valresultat. [accessed 25-April-2019]. [Online]. Available: https://www.val.se/valresultat/riksdag-landsting-och-kommun/2018/valresultat.html

[18] "2005:837 vallag." [Online]. Available: http://rkrattsbaser.gov.se/sfst?bet=2005:837

[19] E. Assarsson, E. Debels, T. Hansson, J. Hellsten, C. Holm, L. Liedberg, I. Lindqvist, D. Olsson, S.Olsson, and J. Storm, "Dn debatt. "förändra valsedelssystemet för att öka trovärdigheten"," *DN*, 2019. [Online]. Available: https://www.dn.se/debatt/forandra-valsedelssystemet-for-att-oka-trovardigheten/

[20] Valmyndigheten. Arbeta som röstmottagare. [Accessed 28-April-2019]. [Online]. Available: https://www.val.se/om-oss/valadministrationen/arbeta-som-rostmottagare.html

[21] Valimised, "Statistics about internet voting in estonia," https://www.valimised.ee/en/archive/statistics-about-internet-voting-estonia, 2019, [Online; accessed 21-April-2019].

[22] B. Perrigo. (2019) What the U.S. Can Learn About Electronic Voting From This Tiny Eastern European Nation. [Online]. Available: http://time.com/5541876/estonia-elections-electronic-voting

[23] Valimised. I-Voting. [Online; accessed 22-April-2019]. [Online]. Available: https://e-estonia.com/solutions/e-governance/i-voting/

[24] D. Springall, T. Finkenauer, Z. Durumeric, J. Kitcat, H. Hursti, M. MacAlpine, and J. A. Halderman, "Security analysis of the estonian internet voting system," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2014, pp. 703–715.

[25] T. Finkenauer and D. Springall, "Estoniavoting.org press-release," https://estoniaevoting.org/press-release/, 2014, [Online; accessed 22-April-2019].

[26] S. Nichter, "Vote buying or turnout buying? machine politics and the secret ballot," *American political science review*, vol. 102, no. 1, pp. 19–31, 2008.

[27] C. Vilalta, "Vote-buying crime reports in mexico: magnitude and correlates," *Crime, law and social change*, vol. 54, no. 5, pp. 325–337, 2010.

[28] S. C. Stokes, "Perverse accountability: A formal model of machine politics with evidence from argentina," *American political science review*, vol. 99, no. 3, pp. 315–325, 2005.

[29] Digiconomist, "Ethereum energy consumption index," https://digiconomist.net/ethereum-energy-consumption, 2019, [Online; accessed 13-Mars-2019].

[30] Enerdata, "Electricity domestic consumption," https://yearbook.enerdata.net/electricity/electricity-domestic-consumption-data.html, 2017, [Online; accessed 26-April-2019].

[31] C. Le Quéré, R. Moriarty, R. M. Andrew, J. G. Canadell, S. Sitch, J. I. Korsbakken, P. Friedlingstein, G. P. Peters, R. J. Andres, T. A. Boden *et al.*, "Global carbon budget 2015," *Earth System Science Data*, vol. 7, pp. 349–396, 2015.

[32] OECD/IEA, "Co2 emissions from electricity and heat production, total (% of total fuel combustion)," https://data.worldbank.org/indicator/EN.CO2.ETOT.ZS, 2014, [Online; accessed 26-April-2019].

[33] Contracts - Solidity 0.5.5 documentation. [Accessed 30-April-2019]. [Online]. Available: https://solidity.readthedocs.io/en/v0.5.5/contracts.html

[34] T. suite, "Drizzle," 2019. [Online]. Available: https://trufflframework.com/drizzle

[35] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ecdsa)," *International journal of information security*, vol. 1, no. 1, pp. 36–63, 2001.

[36] Valmyndigheten, "Val till riksdagen - Ålder och kön," https://data.val.se/val/val2018/alkon/R/rike/alderkon.html#rostberalkon, 2018, [Online; accessed 26-April-2019].

[37] ——, "Val till riksdagen röster," https://data.val.se/val/val2018/slutresultat/R/rike/index.html, 2018, [Online; accessed 26-April-2019].

[38] S. Viswanathan and A. Shah. The Scalability Trilemma. [Online; accessed 16-May-2019]. [Online]. Available: https://medium.com/@aakash_13214/the-scalability-trilemma-in-blockchain-75fb57f646df

[39] E. community. On sharding blockchains. [Online; accessed 25-April-2019]. [Online]. Available: https://github.com/ethereum/wiki/wiki/Sharding-FAQ

[40] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *2017 IEEE international congress on big data (BigData congress)*. IEEE, 2017, pp. 557–564.

[41] Statista, "Number of blockchain wallet users worldwide from 1st quarter 2016 to 1st quarter 2019," https://www.statista.com/statistics/647374/worldwide-blockchain-wallet-users/, 2019, [Online; accessed 2-May-2019].

[42] Etherscan, "Ethereum block time history," https://etherscan.io/chart/blocktime, 2019, [Online; accessed 26-April-2019].

[43] coinmarketcap. Cryptocurrency Market Capitalization. [Online; accessed 7-May-2019]. [Online]. Available: https://coinmarketcap.com/

[44] B. Barhydt. (2019) Crypto bites: Chat with ethereum founder vitalik buterin. Youtube. [Online]. Available: https://www.youtube.com/watch?time_continue=346&v=u-i_mTwL-FI

[45] S. Shankland, "No, blockchain isn't the answer to our voting system woes," https://www.cnet.com/news/blockchain-isnt-answer-to-voting-system-woes/, 2019, [Online; accessed 05-Feb-2019].

Appendices

A Source code

The prototype's source code is available at `https://github.com/datx02-19-85/project`

B Survey

First Section:

- Where are you from?
  - Sweden
  - Other
- How old are you?
  - 16-19
  - 20-26
  - 27-35
  - 35-50
  - 50-65
  - 65+
- How likely do you think that the result in the latest Swedish political election was incorrect to some extent?
  - 1 (Not likely)
  - 2
  - 3 (Very likely)
- How efficient do you think that the current Swedish electoral system is?
  - 1 (Not efficient at all)
  - 2
  - 3
  - 4 (Very efficient)

Second Section:
"Which of these factors do you think is present in a perfect electoral system?"

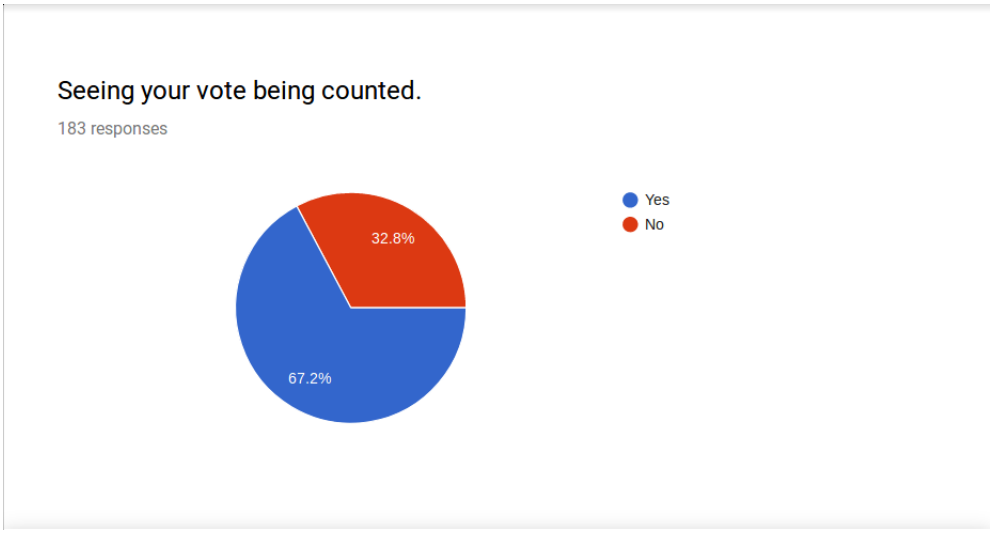
- Seeing your vote being counted.

  - Yes
  - No
- Casting vote in a private polling booth.
  - Yes
  - No
- Seeing live result as votes being casted.
  - Yes
  - No
- Central authority counting votes.
  - Yes
  - No

Third Section:
"Our ultimate idea is to discuss and implement a decentralized electronic voting system."

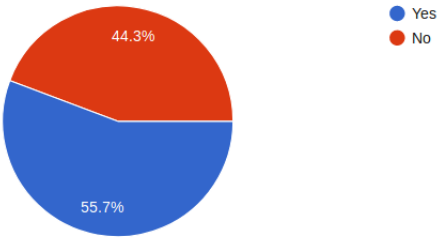
- What are your greatest hopes for an electronic voting system? **(Free text)**
- What are your greatest fears for an electronic voting system? **(Free text)**

C Survey results



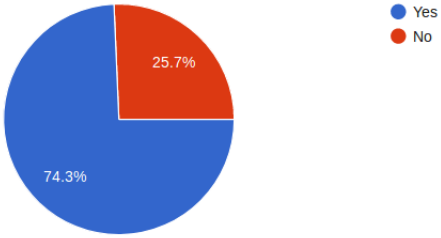
Seeing live result as votes being casted.

183 responses



Central authority counting votes.

183 responses



Seeing live result as votes being casted.

183 responses